Title
Leveraging latent structure in high-dimensional data: causality, neuroscience, and nonparametrics

Author
Bloniarz, Adam

Publication Date
2016

Peer reviewed|Thesis/dissertation

Leveraging latent structure in high-dimensional data: causality, neuroscience, and nonparametrics

by

Adam Edward Bloniarz

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Statistics

and the Designated Emphasis

in

Computational Data Science and Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Bin Yu, Chair
Professor Jasjeet Sekhon
Professor Jack L. Gallant

Spring 2016

Leveraging latent structure in high-dimensional data: causality, neuroscience, and nonparametrics

Adam Edward Bloniarz

Abstract

Leveraging latent structure in high-dimensional data: causality, neuroscience, and nonparametrics

by

Adam Edward Bloniarz

Doctor of Philosophy in Statistics
and the Designated Emphasis in
Computational Data Science and Engineering

University of California, Berkeley

Professor Bin Yu, Chair

Many scientific fields have been changed by rapid technological progress in data collection, storage, and processing. This has greatly expanded the role of statistics in scientific research. The three chapters of this thesis examine core challenges faced by statisticians engaged in scientific collaborations, where the complexity of the data require use of high-dimensional or nonparametric methods, and statistical methods need to leverage lower dimensional structure that exists in the data.

The first chapter concerns the promise and challenge of using large datasets to uncover causal mechanisms. Randomized trials remain the gold-standard for inferring causal effects of treatment a century after their introduction by Fisher and Neyman. In this chapter, we examine whether large numbers of auxilary covariates in a randomized experiment can be leveraged to help improve estimates of the treatment effect and increase power. In particular, we investigate Lasso-based adjustments of treatment effects through theory, simulation, and a case study of a randomized trial of the pulmonary artery cathether. In our investigation, we avoid imposing a linear model, and examine the robustness of Lasso to violations of traditional assumptions.

The second chapter examines the use of predictive models to elucidate functional properties of the mammalian visual cortex. We investigate the activity of single neurons in area MT when stimulated with natural video. One way to investigate single-neuron activity is to build encoding models that predict spike rate given an arbitrary natural stimulus. In this work, we develop encoding models that combine a nonlinear feature extraction step with a linear model. The feature extraction step is unsupervised, and is based on the principle of sparse coding. We compare this model to one that applies relatively simple, fixed nonlinearities to the outputs of V1-like spatiotemporal filters. We find evidence that some MT cells may be tuned to more complex video features than previously thought.

The third chapter examines a computational challenge inherent in nonparametric modeling of large datasets. Large datasets are often stored across many machines in a computer cluster, where communication between machines is slow. Hence, nonparametric regression methods should avoid communication of data as much as possible. Random forests, among the most popular nonparametric methods for regression, are not well-suited to distributed architectures. We develop a modification of random forests that leverage ideas in nonparametric regression by local modeling. Our method allows for training of random forests completely in parallel, without synchronization between machines, with communication of sufficient statistics at test-time only. We show that this method can improve the predictive performance of standard random forests even in the single-machine case, and that performance remains strong when data is distributed.

To my family

Contents

List of Figures

List of Tables

Acknowledgments

In the half decade I have spent at Berkeley, I have been fortunate to work and learn among an extraordinary group of scholars. I am grateful for the generosity of many people in mentoring me and guiding the research presented in this thesis.

I owe a great debt of gratitude to my principal advisor, Bin Yu. She has provided me with a wealth of inspiration for research problems, has helped me to my clarify my thinking and writing, and has greatly contributed to my intellectual and personal growth. I hope that I have absorbed some of her good taste in research and commitment to the advancement of science. I am also grateful for the mentoring of Jasjeet Sekhon. I've benefited from many stimulating conversations and late-night email chains, his very inspiring lectures on causal inference, and his seemingly encyclopedic knowledge of the statistics literature of the last 50 years.

The projects presented in this thesis have all been collaborative efforts. Chapter 2 is adapted from a forthcoming article in the *Proceedings of the National Academy of Sciences*, with Hanzhong Liu, Jasjeet Sekhon, Cun-Hui Zhang, and Bin Yu. I thank Hanzhong, who is co-first author, for his technical help in high-dimensional statistics, his extremely clean and clear way of thinking, and for his not losing patience with me as I knocked on his door again and again to talk about some part of the proof. I thank Cun-Hui Zhang for sharing numerous mathematical insights and clever ways of strengthening our results. I gained much from watching such a sharp and creative mind at work.

In the work in chapter 3, I received much hands-on guidance from Julien Mairal, who was a very patient and generous mentor. In weekly meetings in the first few years of my Ph.D., he shared a great deal of his extensive knowledge of computer vision, sparse coding, and optimization. Additionally, he helped me ramp up my programming and computing skills, such an essential part of being a statistician. I thank him for hosting me for a very enjoyable and productive month-long visit to Inria-Grenoble in the summer of 2013, and again in the spring of 2016. Yuval Benjamini was an excellent mentor for the time that we overlapped at Berkeley, and he helped me better understand the model for V4 [59] that was the basis for the work in this thesis. I also thank Shinji Nishimoto and Jack Gallant for providing the data from their paper on area MT [69], and I particularly thank Jack Gallant for his openness in having me attend his weekly lab meetings. He and his students always had a wealth of interesting statistical problems and practical insights, which could inspire many dissertations worth of material. Shinji, Michael Oliver, Natalia Bilenko, and Mark Lescroart have all been very generous in answering general questions about neuroscience, data collection, and pre-processing. I particularly thank Mark for his careful thinking and many interesting discussions about model comparisons and uncertainty quantification.

I have been very fortunate to collaborate with Ameet Talwalkar on several projects, one of which is the work on distributed random forests presented in chapter 4. Christopher Wu and Bin also contributed substantially to this work, and it will appear at the 2016 AISTATS conference. Ameet has been a helpful guide into the world of big data and distributed computation, and has always been very generous in sharing research ideas, connecting me

to other computer scientists, and giving life advice. I've thorougly enjoyed our research meetings and he has always helped me make progress when I found myself stuck in the weeds.

I am very grateful for the community of students in statistics and EECS at Berkeley. There are too many people to mention, but I want to thank everyone for being great classmates and office-mates, for the late-night sessions working through problem sets, for the efforts with the SGSA, and for all the fun experiences exploring the Bay Area and cooking up decadent feasts.

It has been a pleasure to interact with the administrative and technical staff of the Berkeley Statistics Department, as they have consistenly gone above and beyond to make sure that I had everything I need to progress at Berkeley and carry out my research. La Shana Polaris helped me navigate the administrative ins-and-outs of academic life and I thank her for her extensive knowledge and efficiency. Ryan Lovett and Chris Paciorek were very generous and responsive in helping me out with the department computing facilities, and they have developed a great platform for performing computing-intensive research. Mary Melinn was very generous in providing assitance with department resources, and was always dependable for a bit of Irish wit.

In my final year, I spent most of my time as a guest at the Department of Computer Science at CU Boulder, and I thank Xiao-Chuan Cai for hosting me in his lab and introducing me to the Boulder CS community. The public library in Lafayette, Colorado unknowingly provided free WiFi and a beatiful view of the Rocky Mountains as I typed up much of this document and Googled many cryptic LaTeX errors. My mother-in-law, Marilyn Nims, greatly assisted with child care during her numerous visits to Berkeley and Boulder, and this document would not exist without her help.

I have been very fortunate to be supported financially for the last three years by the National Defense Science & Engineering Graduate Fellowship Program (NDSEG). Prior to that, I was supported by The Center for Science of Information (CSoI), an US NSF Science and Technology Center under grant agreement CCF-0939370, which has also hosted me for many interesting site visits and workshops at Purdue. My visits to Inria were supported by the LEAR team and by the France-Berkeley Fund. I thank the UC Berkeley AMPlab for bringing me on as a GSR for a very enjoyable summer of 2014.

Finally I would like to thank my family—Abby, Ewan, mom, dad, Brian, Kate, Marilyn and Bob—whose support, love, and patience continue to be a great blessing. Abby deserves great credit for putting up with the inevitable twists and turns of Ph.D. life, and even became my wife and mother to our son during this time. I don't know how I got so lucky, but I am grateful for all the random events that eventually led to us finding each other.

Chapter 1

Overview

Lord Ernest Rutherford is famously quoted as saying "if your experiment needs statistics, you ought to have done a better experiment." Given the increasingly central role of statistics in many areas of scientific inquiry, one might worry that Lord Rutherford is rolling in his grave. However, this trend is a result of advances in technology that have resulted in data sources that are high-dimensional and high-throughput, where modern statistical tools are necessary for even the most basic data exploration tasks. Examples include whole-genome sequencing in biology, whole-brain fMRI in neuroscience, and high-resolution sky surveys in astronomy. Even at a more fundamental level, statistics has become a key tool for the the discovery of causal mechanisms in systems that are too complex to be fully characterized with the hard sciences. Examples include agriculture, medical practice, and social impacts of public policy. One of the greatest impacts of statistics in the 20th century has been the recognition that deliberate randomization is a powerful tool that can separate true causal relationships from incidental correlations even when rigorous theoretical models are not conceivable.

It is indeed a golden age to be an applied statistician, as statistics is central to scientific discovery in so many domains. However, Lord Rutherford's sentiment should be taken seriously by statisticians, as it emphasizes the importance of experimental design and domain knowledge in data-driven science. The last 20 years of research in statistics and machine learning have yielded remarkable methods that can learn complex prediction rules with efficiency that matches the limits imposed by information theory. However, there is a danger that such methods obviate the need for real scientific understanding of data: in many downstream applications, what need is there for scientific theory when a statistical procedure can learn a model that makes very accurate predictions, perhaps even better than a model derived from current scientific understanding? Indeed, many of the most successful nonlinear learning algorithms come at the cost of apparent inscrutability of the learned models. Similarly, we can be thankful that in current times, data-driven medical trials have become the gold standard for guiding medical practice, rather than imprecise theories of human biology. However, we don't want our doctors prescribing treatments without some understanding their physical action, and we wouldn't want them depending on randomized trials to determine, for example, whether parachutes are effective tools for preventing gravity-induced

mortality [85].

This thesis explores problems where modern statistical methods, particularly high-dimensional methods and nonparametrics, help to enable real progress in scientific and causal understanding. The methods presented in this thesis all confront the complexity and high-dimensionality of the data by exploiting the presence of latent low-dimensional structure. Uncovering this structure is essential for scientific understanding of the problem at hand. In chapter 2, we consider the analysis of a randomized trial of a treatment in which high-dimensional covariate information is recorded about each patient in the trial. In principle, the covariate information can help reduce variance of the treatment effect estimate, though the high-dimensionality of the covariates renders classical methods like OLS either ill-posed or very unstable, and would yield dense, uninterpretable models. We propose using the least-absolute selection and shrinkage operator (Lasso) [90], which is well-suited to studies where a only sparse subset of covariates are related to the outcomes. We analyze this procedure under the Neyman-Rubin model for a randomized experiment, which makes very weak assumptions on the data generating process. Because the Lasso selects a small subset of covariates for adjustment, it can help guide downstream scientific studies into the interactions of treatment with covariates. Chapter 3 examines the problem of understanding single neuron behavior in area MT, an important mid-level visual cortex region. Early visual areas (eg. lateral geniculate nucleus and V1) can be described fairly accurately with encoding models that combine linear Gabor filters with simple nonlinearities. However, neurons in mid-level visual regions have responses which are more complex nonlinear functions of the visual input. End-to-end nonparametric statistical methods would require impractically large training datasets given the complexity of visual stimuli, and may yield models that are difficult to interpret. Instead, progress in understanding these areas has been guided by a combination of biological understanding of brain function in combination with statistical methods. We focus on motion processing in area MT, and develop encoding models of single-neuron behavior based on the principle of sparse coding introduced in Olshausen [71], and which use trace-norm regularized regression to leverage the approximate time-space separability of neuronal receptive fields demonstrated in Mazer et al. [62].

Another major challenge to statisticians is the necessity of working with the constraints of computer architectures that house and process large datasets. A major recent trend has been a move towards parallelism in both data storage and computation. Large datasets are generally stored in a distributed manner across many machines in a cluster. Communication between machines is extremely slow compared to communication between hard disk and RAM within a machine. Single machines contain more and more processor cores that can independently execute instructions. Hence, for statistics to be useful in analyzing large scientific datasets, statisticians must design methods do not rely on communication and synchronization. Chapter 4 proposes a method for adapting random forests to these criteria, augmenting the standard random forest procedure with local modeling methods from nonparametric statistics. This procedure allows for purely divide-and-conquer, communication-free training of random forests, where statisticial efficiency is not sacrificed even when the data is distributed across many workers. The key to this method is an interpretation of

random forests as an adaptive nearest neighbor method, where the neighborhoods of a point stretch out to ignore dimensions that are irrelevant to the regression problem.

The following sections describe the contributions of this thesis in more detail.

Lasso-adjusted treatment effect estimates

It has been known since the early 20th century that, with a randomized trial, one can estimate the causal effect of a treatment without bias by simply reporting the difference in means between treatment and control groups [86]. Furthermore, one can form confidence intervals that are valid under very weak assumptions. This is the major reason for the prominence of randomized trials in settings where it is impossible to identify all possible confounders. However, when covariate data is present, investigators often use statistical modeling in the hopes of getting a more precise treatment effect estimate. This practice has generated some controversy. As was pointed out in a series of elegant papers by David Freedman, adjustment through regression modeling may inadvertently increase the variance relative to the simple unadjusted estimator [34, 35]. These issues were examined in further detail in the work of Winston Lin [54], who showed that regression with interaction terms prevents many of the pitfalls pointed out by Freedman. Remarkably, Lin's results hold without assuming any underlying linear generative model; instead, only moment and stability conditions on potential outcomes and covariates are required. The variance reducing properties of regression with interactions follow from a geometric argument about projections.

In practical work, outcome measurements in randomized trials are now often accompanied with a very large number of covariates. For example, one may have demographic and genomic information about patients in a medical trial, survey responses for individuals in a policy experiment, or behavioral data about users in an A/B experiment. This motivates a natural followup to Lin's work, which is to consider Lasso-based regression adjustment of the treatment effect estimate. This is the focus of this chapter. I collaborated on this project with a group of researchers who became interested in this problem from different angles: Bin Yu and Hanzhong Liu from Berkeley Statistics, Cun-Hui Zhang from Rutgers, and Jasjeet Sekhon from Berkeley Political Science and Statistics. We approached this problem from the standpoint of Freedman and Lin, using the Neyman-Rubin model as the basis for analysis, as it avoids imposing any explicit generative model on the potential outcomes. This is a major feature of our analysis which differentiates it from earlier theoretical work on the Lasso, which usually assumes an exact generative linear model or misspecified linear model with exogenous errors. Using the Neyman-Rubin model as a baseline, we add conditions that guarantee Lasso reduces the variance of the treatment effect estimator, and show that one can construct asymptotically valid confidence intervals that are generally shorter than the unadjusted confidence intervals. We carry out extensive simulations where we test the robustness of Lasso-based adjustments to violations of assumptions. We also present an application to a medical trial of the pulmonary artery catheter (PAC) to show Lasso-based adjustment in action, and to discuss the important practical problem of selection of regularization parameter. We also compare Lasso adjustment to Lasso+OLS, which uses Lasso for

selection and then OLS for estimation of coefficient values. This procedure produces very parsimonious and interpretable models compared to cross-validated Lasso, and the resulting estimators have similar accuracy.

Modeling area MT with sparse coding principles

Starting in the mid 20th century, neuroscientists began to uncover the structural properties of the mammalian visual cortex, and found that it is hierarchically organized into functionally distinct cortical regions that coordinate to perform the tasks of visual cognition [33, 94]. One way of characterizing the activity of these regions is to describe the behavior of individual cells in response to visual stimuli. This problem is known as receptive field estimation. In its most general sense, the receptive field of a neuron in the visual cortex is a description of how its expected spiking rate varies depending on the contents of the visual field. This is also referred to as an encoding model.

Chapter 3 tackles the problem of building encoding models of neurons in area MT (middle temporal), a major extrastriate cortical region along the dorsal stream, which is involved in motion cognition. Particularly in higher order regions such as MT, estimation of encoding models is challenging problem due to the nonlinearity of neuron responses, the high amount of noise, and the sparsity of neuron spiking. Furthermore, neuroscientists have come to recognize that behavior of neurons under naturalistic conditions can be more complex then when they are probed using only controlled artificial stimuli [91]. Hence, full understanding of single neuron receptive fields, particularly in extrastriate regions, requires building encoding models that are accurate over a wide range of natural video stimuli.

When expressed in raw pixel intensities, natural images and videos are very high-dimensional objects, where low-dimensional substructure, such as object classes, positions, texture descriptions, is not clearly represented. One important approach to estimating encoding models is to use theoretical models of vision to transform the image into a representation where image structure is more clearly represented, and modeling of brain responses becomes more tractable. In this chapter, we follow this approach, and build encoding models that combine a nonlinear feature representation of video stimuli with a linear model that predicts the neuron spike rate. The initial nonlinear step is built purely using biological principles and unsupervised learning techniques. Recordings of neuron spike counts are used as training data to build the top-level linear model only.

Our feature representation for natural videos is inspired by the principle of sparse coding, which has proven to be a widely applicable theoretical model for single-neuron tuning in a variety of sensory areas [72]. Our representation, at a high level, has three layers. The first layer acts as a caricature of V1 neurons, and filters the stimulus video with a set of spatiotemporal wavelets. This layer reflects the extensive innervation of MT by axonal projections from V1. The second layer transforms the output of the first layer by re-coding image patches in a sparse dictionary. The third layer performs pools of patch representations within spatial regions, which greatly reduces dimension and makes the video representation more spatially invariant. This approach was previously developed for modeling neurons in

area V4 in [58], where it was shown to yield much more accurate encoding models compared to those based on Gabor wavelet features. The major contribution in this chapter is to extend this feature extraction pipeline to natural video stimuli and to show that it is effective for modeling neurons in area MT.

In this work I collaborated with Bin Yu, Julien Mairal, and members of the Gallant lab at UC Berkeley. The dataset analyzed in this chapter consists of high-quality recordings of 52 MT neurons from *Macacca mulatta*, each stimulated with 20,000-40,000 frames of natural video. This dataset, previously analyzed in Nishimoto and Gallant [69], provides a rich source for the understanding of MT's role in motion estimation. We compare the performance of our sparse-coding model with the spatiotemporal Gabor wavelet model presented in Nishimoto and Gallant [69]. The prediction accuracy of our model is higher, on average, than that of the Gabor wavelet model. While many neurons are equally well modeled by both pipelines, we find that the sparse coding model encodes complex video features that are useful for modeling a subset of MT neurons. Canonical correlation analysis reveals that the two representations are nearly identical along several directions, explaining why many MT neurons are equally well-modeled with or without the sparse coding map. These findings suggest that receptive fields in area MT are heterogenous, and that a subset of neurons have senstivity to features more complex than spatiotemporal Gabor wavelets.

Supervised neighborhoods for distributed nonparametric regression

Large datasets hold the promise of allowing statisticians to make few assumptions when building models. In the context of regression problems, the statistician may want to only assume that the true generative function comes from a broad class of functions with some degree of smoothness, but otherwise not constrain the structure of the function any further (e.g. with assumptions of linearity, additivity, convexity, etc.). This is known as the problem of nonparametric regression. The information theoretic challenges of this problem were shown in a series of papers of Stone [87, 88], who derived minimax lower bounds for pointwise and integrated mean square risk for estimation of nonlinear functions in Hölder smoothness classes. These results were pessimistic, in a sense, reflecting the fundamental difficulty of nonlinear regression due to the curse of dimensionality. Their minimax rates imply that sample size must grow exponentially in the dimension of the problem to attain a fixed error. In problems of even moderate dimension by today's standards (> 20), this would seem to imply that nonparametric regression is intractable even with huge training sets.

However, the ensuing three decades have seen an explosion in successful applications for nonparametric regression and classification even in complex, high-dimensional problems. This reflects the fact that this minimax thinking is too conservative, and there is often very useful lower-dimensional structure in the problem which can be exploited by statistical methods. Though it is known that local polynomial methods like LOESS [20] attain the minimax optimal rate over Hölder classes, these methods are very rarely applied to problems

in high dimensions, as they do not adapt to lower dimensional substructure. Instead, tree-based methods (CART, random forests), neural networks, and support vector machines have been much more popular in practical statistical work. Tree-based methods, in particular, adapt very well to regression problems where there are many irrelevant predictors. Variable selection is usually built in to the node splitting criteria, so that the irrelevant predictors are usually ignored.

The work in chapter 4 is a result of collaboration with Ameet Talwalkar, Christopher Wu, and Bin Yu. We develop a new procedure that attempts to merge the advantages of local modeling with the power of random forests to adapt to the shape of the response. Leveraging the interpretation of random forests as providing supervised neighborhoods, we fit local models using a weight function derived from random forests. We show that the procedure is consistent under some assumptions on the random forest training procedure. Empirically, we see that it reduces the bias relative to standard random forest predictions. Furthermore, we find that this local modeling method is very effective in enabling distributed nonparametric estimation with random forests. With distributed datasets, as stated above, communication is very slow, and it is desirable that workers can process the data as independently as possible. At one extreme, the most simple method is for a master node to average predictions from random forests fit independently on each worker. Averaging in such a way will yield a low-variance estimate due to cancelling of random error; however, individual workers' biases do not cancel, and the overall estimator will have the same bias as the estimator fit on the data of a single machine. Performing a local regression on the master node, however, shifts the tradeoff in a favorable way if the function is smooth: the local regression increases variance but yields a comparatively large reduction in bias. We examine the performance of this method on several simulated nonlinear functions, as well as a real regression task from the million song dataset. We implement it in Apache Spark™, a popular framework for distributed computation, and find that it improves on the industry standard distributed random forest implementation in terms of both statistical accuracy and computational speed. Because our method is purely divide-and-conquer, the training time does not increase with the number of workers in the cluster. We do perform extra communication and computation at test time, but are still able to make predictions in less than 20 ms per test point.

Chapter 2

Lasso adjustments of treatment effect estimates in randomized experiments

2.1 Introduction

Randomized experiments are widely used to measure the efficacy of treatments. Randomization ensures that treatment assignment is not influenced by any potential confounding factors, both observed and unobserved. Experiments are particularly useful when there is no rigorous theory of a system's dynamics, and full identification of confounders would be impossible. This advantage was cast elegantly in mathematical terms in the early 20th century by Jerzy Neyman, who introduced a simple model for randomized experiments, which showed that the difference of average outcomes in the treatment and control groups is statistically unbiased for the Average Treatment Effect (ATE) over the experimental sample [86].

However, no experiment occurs in a vacuum of scientific knowledge. Often, baseline covariate information is collected about individuals in an experiment. Even when treatment assignment is not related to these covariates, analyses of experimental outcomes often take them into account with the goal of improving the accuracy of treatment effect estimates. In modern randomized experiments, the number of covariates can be very large—sometimes even larger than the number of individuals in the study. In clinical trials overseen by regulatory bodies like the Food and Drug Administration and Medicines and Healthcare products Regulatory Agency, demographic and genetic information may be recorded about each patient. In applications in the tech industry, where randomization is often called A/B testing, there is often a huge amount of behavioral data collected on each user. However, in this 'big data' setting, much of this data may be irrelevant to the outcome being studied or there may be more potential covariates than observations, especially once interactions are taken into account. In these cases, selection of important covariates or some form of regularization is necessary for effective regression adjustment.

To ground our discussion, we examine a randomized trial of the Pulmonary Artery Catheter (PAC) that was carried out in 65 intensive care units in the UK between 2001 and 2004,

called PAC-man [40]. The PAC is a monitoring device commonly inserted into critically ill patients after admission to intensive care, and it provides a continuous measurement of several indicators of cardiac activity. However, insertion of PAC is an invasive procedure that carries some risk of complications (including death), and it involves significant expenditure both in equipment costs and personnel [24]. Controversy over its use came to a head when an observational study found that PAC had an adverse effect on patient survival and led to increased cost of care [21]. This led to several large-scale randomized trials, including PAC-man.

In the PAC-man trial, randomization of treatment was largely successful, and a number of covariates were measured about each patient in the study. If covariate interactions are included, the number of covariates exceeds the number of individuals in the study; however, few of them are predictive of the patient's outcome. As it turned out, the (pre-treatment) estimated probability of death was imbalanced between the treatment and control groups ($p = 0.005$, Wilcoxon rank sum test). Because the control group had, on average, a slightly higher risk of death, the unadjusted difference-in-means estimator may overestimate the benefits of receiving a PAC. Adjustment for this imbalance seems advantageous in this case, since the pre-treatment probability of death is clearly predictive of health outcomes post-treatment.

In this chapter, we study regression-based adjustment, using the Lasso to select relevant covariates. Standard linear regression based on ordinary least squares suffers from overfitting if a large number of covariates and interaction terms are included in the model. In such cases, researchers sometimes perform model selection based on observing which covariates are unbalanced given the realized randomization. This generally leads to misleading inferences because of incorrect test levels [77]. The Lasso [90] provides researchers with an alternative that can mitigate these problems and still perform model selection. We define an estimator, $\widehat{ATE}_{\text{Lasso}}$, which is based on running an l_1-penalized linear regression of the outcome on treatment, covariates and, following the method introduced in Lin [54], treatment $\times$ covariate interactions. Because of the geometry of the l_1 penalty, the Lasso will usually set many regression coefficients to 0, and is well defined even if the number of covariates is larger than the number of observations. The Lasso's theoretical properties under the standard linear model have been widely studied in the last decade; consistency properties for coefficient estimation, model selection, and out-of-sample prediction are well understood (see Bühlmann and van de Geer [17] for an overview).

In the theoretical analysis in this chapter, instead of assuming that the standard linear model is the true data-generating mechanism, we work under the aforementioned nonparametric model of randomization introduced by Neyman [86] and popularized by Donald Rubin [81]. In this model, the outcomes and covariates are fixed quantities, and the treatment group is assumed to be sampled without replacement from a finite population. The treatment indicator, rather than an error term, is the source of randomness, and it determines which of two potential outcomes is revealed to the experimenter. Unlike the standard linear model, the Neyman-Rubin model makes few assumptions not guaranteed by the randomiza-

tion itself. The setup of the model does rely on the stable unit treatment value assumption (SUTVA), which states that there is only one version of treatment, and that the potential outcome of one unit should be unaffected by the particular assignment of treatments to the other units; however it makes no assumptions of linearity or exogeneity of error terms. Ordinary Least Squares (OLS) [35][34][54], logistic regression [36], and post-stratification [66] are among the adjustment methods that have been studied under this model.

To be useful to practitioners, the Lasso-based treatment effect estimator must be consistent and yield a method to construct valid confidence intervals. We outline conditions on the covariates and potential outcomes that will guarantee these properties. We show that an upper bound for the asymptotic variance can be estimated from the model residuals, yielding asymptotically conservative confidence intervals for the average treatment effect which can be substantially narrower than the unadjusted confidence intervals. Simulation studies are provided to show the advantage of the Lasso adjusted estimator and to show situations where it breaks down. We apply the estimator to the PAC-man data, and compare the estimates and confidence intervals derived from the unadjusted, OLS-adjusted, and Lasso-adjusted methods. We also compare different methods of selecting the Lasso tuning parameter on this data.

2.2 Framework and definitions

We give a brief outline of the Neyman-Rubin model for a randomized experiment; the reader is urged to consult Splawa-Neyman et al. [86], Rubin [81], and Holland [44] for more details. We follow the notation introduced in Freedman [35] and Lin [54]. For concreteness, we illustrate the model in the context of the PAC-man trial.

For each individual in the study, the model assumes that there exists a pair of quantities representing his/her health outcomes under the possibilities of receiving and not receiving the catheter. These are called the potential outcomes under treatment and control, and are denoted as a_i and b_i, respectively. In the course of the study, the experimenter observes only one of these quantities for each individual, since the catheter is either inserted or not. The causal effect of the treatment on individual i is defined, in theory, to be $a_i - b_i$, but this is unobservable. Instead of trying to infer individual-level effects, we will assume that the intention is to estimate the average causal effect over the whole population, as outlined in the next section.

In the mathematical specification of this model we consider the potential outcomes to be fixed, non-random quantities, even though they are not all observable. The only randomness in the model comes from the assignment of treatment, which is controlled by the experimenter. We define random treatment indicators T_i, which take on a value 1 for a treated individual, or 0 for an untreated individual. We will assume that the set of treated individuals is sampled without replacement from the full population, where the size of the treatment group is fixed beforehand; thus the T_i are identically distributed but not independent.

The model for the observed outcome for individual i, defined as Y_i, is thus

$$Y_i = T_i a_i + (1 - T_i) b_i.$$

This equation simply formalizes the idea that the experimenter observes the potential outcome under treatment for those who receive the treatment, and the potential outcome under control for those who do not.

Note that the model does not incorporate any covariate information about the individuals in the study, such as physiological characteristics or health history. However, we will assume we have measured a vector of baseline, pre-experimental covariates for each individual i. These might include, for example, age, gender, and genetic makeup. We denote the covariates for individual i as the column vector $\mathbf{x}_i = (x_{i1}, ..., x_{ip})^T \in \mathbb{R}^p$ and the full design matrix of the experiment as $X = (\mathbf{x}_1, ..., \mathbf{x}_n)^T$. In the theoretical results, we will assume that there is a correlational relationship between an individual's potential outcomes and covariates, but we will not assume a generative statistical model.

Define the set of treated individuals as $A = \{i \in \{1, ..., n\} : T_i = 1\}$, and similarly define the set of control individuals as B. Define the number of treated and control individuals as $n_A = |A|$ and $n_B = |B|$, respectively, so that $n_A + n_B = n$. We add a line on top of a quantity to indicate its average and a subscript A or B to label the treatment or control group. Thus, for example, the average of the potential outcomes and the covariates in the treatment group are

$$\bar{a}_A = n_A^{-1} \sum_{i \in A} a_i, \ \bar{\mathbf{x}}_A = n_A^{-1} \sum_{i \in A} \mathbf{x}_i,$$

respectively. Note that these are random quantities in this model, since the set A is determined by the random treatment assignment. Averages over the whole population are denoted as

$$\bar{a} = n^{-1} \sum_{i=1}^{n} a_i, \ \bar{b} = n^{-1} \sum_{i=1}^{n} b_i, \ \bar{\mathbf{x}} = n^{-1} \sum_{i=1}^{n} \mathbf{x}_i.$$

Note that the averages of potential outcomes over the whole population are not considered random, but are unobservable.

2.3 Treatment effect estimation

Our main inferential goal will be average effect of the treatment over the whole population in the study. In a trial such as PAC-man, this represents the difference between the average outcome if everyone had received the catheter, and the average outcome if no one had received it. This is defined as

$$ATE = \bar{a} - \bar{b}.$$

The most natural estimator arises by replacing the population averages with the sample averages:

$$\widehat{ATE}_{\text{unadj}} \stackrel{\text{def}}{=} \bar{a}_A - \bar{b}_B,$$

The subscript "unadj" indicates an estimator without regression adjustment. The foundational work in Splawa-Neyman et al. [86] points out that, under a randomized assignment of treatment, $\widehat{ATE}_{\text{unadj}}$ is unbiased for ATE, and derives a conservative procedure for estimating its variance.

While $\widehat{ATE}_{\text{unadj}}$ is an attractive estimator, covariate information can be used to make adjustments in the hope of reducing variance. A commonly used estimator is

$$\widehat{ATE}_{\text{adj}} = \left[\bar{a}_A - (\bar{\mathbf{x}}_A - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(a)}\right] - \left[\bar{b}_B - (\bar{\mathbf{x}}_B - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(b)}\right]$$

where $\hat{\boldsymbol{\beta}}^{(a)}, \hat{\boldsymbol{\beta}}^{(b)} \in \mathbb{R}^p$ are adjustment vectors for the treatment and control groups, respectively, as indicated by the superscripts. The terms $\bar{\mathbf{x}}_A - \bar{\mathbf{x}}$ and $\bar{\mathbf{x}}_B - \bar{\mathbf{x}}$ represent the fluctuation of the covariates in the subsample relative to the full sample, and the adjustment vectors fit the linear relationships between the covariates and potential outcomes under treatment and control. For example, in the PAC-man trial, this would help alleviate the imbalance in the pre-treatment estimated probability of death: the corresponding element of $\bar{\mathbf{x}}_B - \bar{\mathbf{x}}$ would be positive (due to the higher average probability of death in the control group), the corresponding element of $\hat{\boldsymbol{\beta}}^{(b)}$ would be negative (a higher probability of death correlates with worse health outcomes), so the overall treatment effect estimate would be adjusted downwards. This procedure is equivalent to imputing the unobserved potential outcomes; if we define

$$\hat{\bar{a}}_B = \bar{a}_A + (\bar{\mathbf{x}}_B - \bar{\mathbf{x}}_A)^T \hat{\boldsymbol{\beta}}^{(a)}, \;\; \hat{\bar{b}}_A = \bar{b}_B + (\bar{\mathbf{x}}_A - \bar{\mathbf{x}}_B)^T \hat{\boldsymbol{\beta}}^{(b)},$$

we can form the equivalent estimator

$$\widehat{ATE}_{\text{adj}} = n^{-1}\left(n_A \bar{a}_A + n_B \hat{\bar{a}}_B\right) - n^{-1}\left(n_B \bar{b}_B + n_A \hat{\bar{b}}_A\right).$$

If we consider these adjustment vectors to be fixed (non-random), or if they are derived from an independent data source, then this estimator is still unbiased, and may have substantially smaller asymptotic and finite-sample variance than the unadjusted estimator. This allows for construction of tighter confidence intervals for the true treatment effect.

In practice, the "ideal" linear adjustment vectors, leading to a minimum-variance estimator of the form of $\widehat{ATE}_{\text{adj}}$, cannot be computed from the observed data. However, they can be estimated, possibly at the expense of introducing modest finite-sample bias into the treatment effect estimate. In the classical setup, when the number of covariates is relatively small, ordinary least squares (OLS) regression can be used. The asymptotic properties of this kind of estimator are explored under the Neyman-Rubin model in Freedman [34, 36] and Lin [54]. We will follow a particular scheme which is studied in Lin [54] and shown to have favorable properties: we regress the outcome on treatment indicators, covariates, and treatment $\times$ covariate interactions. This is equivalent to running separate regressions in the treatment and control groups of outcome against an intercept and covariates. If we define

$\hat{\boldsymbol{\beta}}^{(a)}_{\text{OLS}}$ and $\hat{\boldsymbol{\beta}}^{(b)}_{\text{OLS}}$ as the coefficients from the separate regressions, then the estimator is

$$\widehat{ATE}_{\text{OLS}} = \left[\bar{a}_A - (\bar{\mathbf{x}}_A - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(a)}_{\text{OLS}}\right] - \left[\bar{b}_B - (\bar{\mathbf{x}}_B - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(b)}_{\text{OLS}}\right]$$

This has some finite-sample bias, but Lin [54] shows that it vanishes quickly at the rate of $1/n$ under moment conditions on the potential outcomes and covariates. Moreover, for a fixed p, under regularity conditions, the inclusion of interaction terms guarantees that it never has higher asymptotic variance than the unadjusted estimator, and asymptotically conservative confidence intervals for the true parameter can be constructed.

In modern randomized trials, where a large number of covariates are recorded for each individual, p may be comparable to or even larger than n. In this case OLS regression can overfit the data badly, or may even be ill-posed, leading to estimators with large finite-sample variance. To remedy this, we propose estimating the adjustment vectors using the Lasso [90]. The adjustment vectors would take the form

$$\hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}} = \arg\min_{\boldsymbol{\beta}} \left[\frac{1}{2n_A} \sum_{i \in A} \left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \boldsymbol{\beta}\right)^2 + \lambda_a \sum_{j=1}^{p} |\beta_j|\right] \tag{2.1}$$

$$\hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}} = \arg\min_{\boldsymbol{\beta}} \left[\frac{1}{2n_B} \sum_{i \in B} \left(b_i - \bar{b}_B - (\mathbf{x}_i - \bar{\mathbf{x}}_B)^T \boldsymbol{\beta}\right)^2 + \lambda_b \sum_{j=1}^{p} |\beta_j|\right] \tag{2.2}$$

and the proposed Lasso adjusted ATE estimator is[1]

$$\widehat{ATE}_{\text{Lasso}} = \left[\bar{a}_A - (\bar{\mathbf{x}}_A - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}}\right] - \left[\bar{b}_B - (\bar{\mathbf{x}}_B - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}}\right].$$

Here λ_a and λ_b are regularization parameters for the Lasso which must be chosen by the experimenter; simulations show that cross-validation works well. In the next section, we study this estimator under the Neyman-Rubin model, and provide conditions on the potential outcomes, the covariates and the regularization parameters under which $\widehat{ATE}_{\text{Lasso}}$ enjoys similar asymptotic and finite-sample advantages as $\widehat{ATE}_{\text{OLS}}$.

It is worth noting that when two different adjustments are made for the treatment and control groups as in Lin [54] and here, the covariates do not have to be the same for the two groups. However, when they are not the same, the Lasso or OLS adjusted estimators are no longer guaranteed to have smaller or equal asymptotic variance than the unadjusted one, even in the case of fixed p. In practice, one may still choose between the adjusted and unadjusted estimators based on the widths of the corresponding confidence intervals.

[1] To simplify the notation, we omit the dependence of $\hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}}$, $\hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}}$, λ_a and λ_b on the population size n.

2.4 Theoretical results

Notation

For a vector $\boldsymbol{\beta} \in R^p$ and a subset $S \subset \{1, ..., p\}$, let β_j be the j-th component of $\boldsymbol{\beta}$, $\boldsymbol{\beta}_S = (\beta_j : j \in S)^T$, S^c be the complement of S, and $|S|$ the cardinality of the set S. For any column vector $\mathbf{u} = (u_1, ..., u_m)^T$, let $\|\mathbf{u}\|_2^2 = \sum_{i=1}^m u_i^2$, $\|\mathbf{u}\|_1 = \sum_{i=1}^m |u_i|$, $\|\mathbf{u}\|_\infty = \max_{i=1,...,m} |u_i|$ and $\|\mathbf{u}\|_0 = |\{j : u_j \neq 0\}|$. For a given $m \times m$ matrix D, let $\lambda_{\min}(D)$ and $\lambda_{\max}(D)$ be the smallest and largest eigenvalues of D respectively, and D^{-1} the inverse of the matrix D. Let $\xrightarrow{d}$ and $\xrightarrow{p}$ denote convergence in distribution and in probability, respectively.

Decomposition of the potential outcomes

The Neyman-Rubin model does not assume a linear relationship between the potential outcomes and the covariates. In order to study the properties of adjustment under this model, we decompose the potential outcomes into a term linear in the covariates and an error term. Given vectors of coefficients $\boldsymbol{\beta}^{(a)}, \boldsymbol{\beta}^{(b)} \in \mathbb{R}^p$, we write[2] for $i = 1, ..., n$,

$$a_i = \bar{a} + (\mathbf{x}_i - \bar{\mathbf{x}})^T \boldsymbol{\beta}^{(a)} + e_i^{(a)}, \tag{2.3}$$

$$b_i = \bar{b} + (\mathbf{x}_i - \bar{\mathbf{x}})^T \boldsymbol{\beta}^{(b)} + e_i^{(b)}. \tag{2.4}$$

Note that we have not added any assumptions to the model; we have simply defined unit-level residuals, $e_i^{(a)}$ and $e_i^{(b)}$, given the vectors $\boldsymbol{\beta}^{(a)}, \boldsymbol{\beta}^{(b)}$. All the quantities in (2.3) and (2.4) are fixed, deterministic numbers. It is easy to verify that $\bar{e}^{(a)} = \bar{e}^{(b)} = 0$. In order to pursue a theory for the Lasso, we will add assumptions on the populations of a_i's, b_i's, and $\mathbf{x}_i$'s, and we will assume the existence of $\boldsymbol{\beta}^{(a)}, \boldsymbol{\beta}^{(b)}$ such that the error terms satisfy certain assumptions.

Conditions

We will need the following to hold for both the treatment and control potential outcomes. The first set of assumptions (conditions 1–3) are similar to those found in Lin [54].

Condition 1. Stability of treatment assignment probability.

$$n_A/n \to p_A, \text{ as } n \to \infty \tag{2.5}$$

for some $p_A \in (0, 1)$.

Condition 2. The centered moment conditions. There exists a fixed constant $L > 0$ such that, for all $n = 1, 2, ...$ and $j = 1, ..., p$,

$$n^{-1} \sum_{i=1}^n (x_{ij} - (\bar{\mathbf{x}})_j)^4 \leq L; \tag{2.6}$$

[2] Again, we omit the dependence of $\boldsymbol{\beta}^{(a)}$, $\boldsymbol{\beta}^{(b)}$, λ_a, λ_b, $e^{(a)}$ and $e^{(b)}$ on n.

$$n^{-1}\sum_{i=1}^{n}(e_i^{(a)})^4 \leq L; \quad n^{-1}\sum_{i=1}^{n}(e_i^{(b)})^4 \leq L. \tag{2.7}$$

Condition 3. The means $n^{-1}\sum_{i=1}^{n}(e_i^{(a)})^2$, $n^{-1}\sum_{i=1}^{n}(e_i^{(b)})^2$ and $n^{-1}\sum_{i=1}^{n}e_i^{(a)}e_i^{(b)}$ converge to finite limits.

Since we consider the high-dimensional setting where p is allowed to be much larger than n, we need additional assumptions to ensure that the Lasso is consistent for estimating $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$. Before stating them, we define several quantities.

Definition 1. Given $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$, the sparsity measures for treatment and control groups, $s^{(a)}$ and $s^{(b)}$, are defined as the number of nonzero elements of $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$, i.e.,

$$s^{(a)} = |\{j : \beta_j^{(a)} \neq 0\}|, \;\; s^{(b)} = |\{j : \beta_j^{(b)} \neq 0\}|, \tag{2.8}$$

respectively. We will allow $s^{(a)}$ and $s^{(b)}$ to grow with n, though the notation does not explicitly show this.

Definition 2. Define δ_n to be the maximum covariance between the error terms and the covariates.

$$\delta_n = \max_{\omega=a,b}\left\{\max_j \left|\frac{1}{n}\sum_{i=1}^{n}\left(x_{ij} - (\bar{\mathbf{x}})_j\right)\left(e_i^{(\omega)} - \bar{e}^{(\omega)}\right)\right|\right\}. \tag{2.9}$$

The following conditions will guarantee that the Lasso consistently estimates the adjustment vectors $\boldsymbol{\beta}^{(a)}, \boldsymbol{\beta}^{(b)}$ at a fast enough rate to ensure asymptotic normality of $\widehat{ATE}_{\text{Lasso}}$. It is an open question whether a weaker form of consistency would be sufficient for our results to hold.

Condition 4. Decay and scaling. Let $s = \max\left\{s^{(a)}, s^{(b)}\right\}$.

$$\delta_n = o\left(\frac{1}{s\sqrt{\log p}}\right). \tag{2.10}$$

$$(s\log p)/\sqrt{n} = o(1). \tag{2.11}$$

Condition 5. Cone invertibility factor. Define the Gram matrix as $\Sigma = n^{-1}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$: There exist constants $C > 0$ and $\xi > 1$ not depending on n, such that

$$\|\mathbf{h}_S\|_1 \leq Cs\|\Sigma\mathbf{h}\|_\infty, \;\; \forall \mathbf{h} \in \mathcal{C}, \tag{2.12}$$

with $\mathcal{C} = \{\mathbf{h} : \|\mathbf{h}_{S^c}\|_1 \leq \xi\|\mathbf{h}_S\|_1\}$, and

$$S = \{j : \boldsymbol{\beta}_j^{(a)} \neq 0 \text{ or } \boldsymbol{\beta}_j^{(b)} \neq 0\}. \tag{2.13}$$

Condition 6. Let $\tau = \min\left\{1/70, (3p_A)^2/70, (3-3p_A)^2/70\right\}$. For constants $0 < \eta < \frac{\xi-1}{\xi+1}$ and $\frac{1}{\eta} < M < \infty$, assume the regularization parameters of the Lasso belong to the sets

$$\lambda_a \in (\frac{1}{\eta}, M] \times \left(\frac{2(1+\tau)L^{1/2}}{p_A} \sqrt{\frac{2\log p}{n}} + \delta_n \right), \tag{2.14}$$

$$\lambda_b \in (\frac{1}{\eta}, M] \times \left(\frac{2(1+\tau)L^{1/2}}{p_B} \sqrt{\frac{2\log p}{n}} + \delta_n \right). \tag{2.15}$$

Denote respectively the population variances of $e^{(a)}$ and $e^{(b)}$ and the population covariance between them by

$$\sigma^2_{e^{(a)}} = n^{-1}\sum_{i=1}^n (e_i^{(a)})^2, \quad \sigma^2_{e^{(b)}} = n^{-1}\sum_{i=1}^n (e_i^{(b)})^2,$$
$$\sigma_{e^{(a)}e^{(b)}} = n^{-1}\sum_{i=1}^n e_i^{(a)} e_i^{(b)}.$$

Theorem 1. *Assume conditions 1–6 hold for some $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$. Then*

$$\sqrt{n}\left(\widehat{ATE}_{Lasso} - ATE\right) \xrightarrow{d} \mathcal{N}\left(0, \sigma^2\right) \tag{2.16}$$

where

$$\sigma^2 = \lim_{n\to\infty} \left[\frac{1-p_A}{p_A}\sigma^2_{e^{(a)}} + \frac{p_A}{1-p_A}\sigma^2_{e^{(b)}} + 2\sigma_{e^{(a)}e^{(b)}} \right]. \tag{2.17}$$

The proof of Theorem 1 is given in section 2.A. It is easy to show, as in the following corollary, that the asymptotic variance of $\widehat{ATE}_{\text{Lasso}}$ is no worse than $\widehat{ATE}_{\text{unadj}}$ when $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$ are defined as coefficients of regressing potential outcomes on a subset of covariates. More specifically, suppose there exists a subset $J \subset \{1, ..., p\}$, such that

$$\boldsymbol{\beta}^{(a)} = ((\boldsymbol{\beta}^{(a)}_J)^T, \mathbf{0})^T, \ \boldsymbol{\beta}^{(b)} = ((\boldsymbol{\beta}^{(b)}_J)^T, \mathbf{0})^T, \tag{2.18}$$

where $\boldsymbol{\beta}^{(a)}_J$ and $\boldsymbol{\beta}^{(b)}_J$ are the population level OLS coefficients for regressing the potential outcomes a and b on the covariates in the subset J with intercept, respectively.

Corollary 1. *For $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$ defined in (2.18) and some λ_a and λ_b, assume conditions 1–6 hold. Then the asymptotic variance of $\sqrt{n}\ \widehat{ATE}_{\text{Lasso}}$ is no greater than that of the $\sqrt{n}\ \widehat{ATE}_{\text{unadj}}$. The difference is $\frac{1}{p_A(1-p_A)}\Delta$, where*

$$\Delta = -\lim_{n\to\infty} \|X\boldsymbol{\beta}_E\|_2^2 \le 0, \quad \boldsymbol{\beta}_E = (1-p_A)\boldsymbol{\beta}^{(a)} + p_A\boldsymbol{\beta}^{(b)}. \tag{2.19}$$

The proof of Corollary 1 is given in section 2.B.

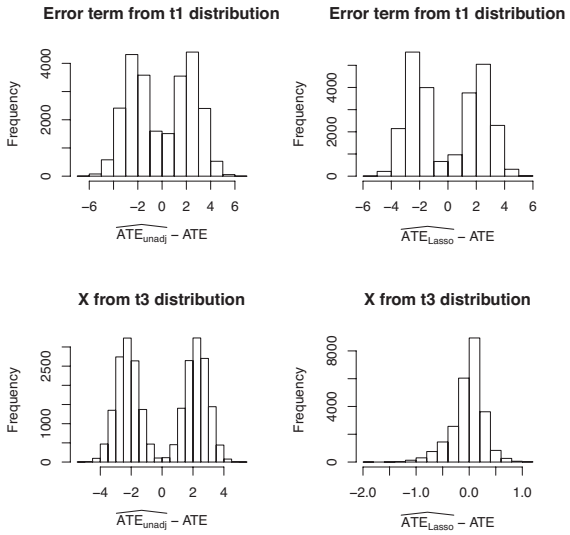

FIGURE 2.1: **Histograms of the unadjusted estimator and the Lasso adjusted estimator when the moment conditions do not hold.** We select the tuning parameters for Lasso using 10-fold cross validation. The potential outcomes are simulated from a linear regression model and then kept fixed. See section 2.E for details. For the upper two subplots, the error terms are generated from t distribution with one degree of freedom and therefore do not satisfy second moment condition; while for the lower two subplots, the covariates are generated from t distribution with there degrees of freedom and thus violate fourth moment condition.

Remark 1. If, instead of eq. (2.6), we assume that the covariates are uniformly bounded, i.e., $\max_{i,j} |x_{ij}| \leq L$, then the fourth moment condition on the error terms, given in eq. (2.7), can be weakened to a second moment condition. While we do not prove the necessity of any of our conditions, our simulation studies show that the distributions of the unadjusted and the Lasso adjusted estimator may be non-normal when: (1) The covariates are generated from Gaussian distributions and the error terms do not satisfy second moment condition, e.g., being generated from a t distribution with one degree of freedom; or (2) The covariates do not have bounded fourth moments, e.g., being generated from a t distribution with three degrees of freedom. See the histograms in fig. 2.1 where the corresponding p-values of Kolmogorov–Smirnov testing for normality are less than $2.2e-16$. These findings indicate that our moment conditions cannot be dramatically weakened for asymptotic normality. However, we also find that the Lasso adjusted estimator still has smaller variance and mean squared error than the unadjusted estimator, even when these moment conditions do not hold. In practice, when the covariates do not have bounded fourth moments, one may perform some transformation—e.g., a logarithm transformation—to ensure that the transformed covariates have bounded fourth moments while having a sufficiently large variance so as to retain useful information. We leave it as future work to explore the properties of different transformations.

Remark 2. The rate given in eq. (2.11), typically required in de-biasing the Lasso [103], is stronger by a factor of $\sqrt{\log p}$ than the usual requirement for l_1 consistency of the Lasso.

Remark 3. Condition 5 is slightly weaker than the typical restricted eigenvalue condition for analyzing the Lasso.

Remark 4. If we assume $\delta_n = O\left(\sqrt{\frac{\log p}{n}}\right)$ which satisfies eq. (2.10), then condition 6 requires that the tuning parameters are proportional to $\sqrt{\frac{\log p}{n}}$ which is typically assumed for the Lasso in the high-dimensional linear regression model.

Remark 5. For fixed p, $\delta_n = 0$ in eq. (2.9), condition 4 holds automatically, and condition 5 holds when the smallest eigenvalue of Σ is uniformly bounded away from 0. In this case, corollary 1 reverts to corollary 1.1. in Lin [54]. When these conditions are not satisfied, we should set λ_a and λ_b to be large enough to cause the Lasso adjusted estimator to revert to the unadjusted one.

2.5 Neyman-type conservative variance estimate

We note that the asymptotic variance in Theorem 1 involves the cross-product term $\sigma_{e^{(a)}e^{(b)}}$ which is not consistently estimable in the Neyman-Rubin model as a_i and b_i are never simultaneously observed. However, we can give a Neyman-type conservative estimate of the variance. Let

$$\begin{aligned}\hat{\sigma}^2_{e^{(a)}} &= \frac{1}{n_A - df^{(a)}} \sum_{i \in A} \left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}}\right)^2 \\ \hat{\sigma}^2_{e^{(b)}} &= \frac{1}{n_B - df^{(b)}} \sum_{i \in B} \left(b_i - \bar{b}_B - (\mathbf{x}_i - \bar{\mathbf{x}}_B)^T \hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}}\right)^2\end{aligned} \tag{2.20}$$

where $df^{(a)}$ and $df^{(b)}$ are degrees of freedom defined by

$$df^{(a)} = \hat{s}^{(a)} + 1 = ||\hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}}||_0 + 1; \; df^{(b)} = \hat{s}^{(b)} + 1 = ||\hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}}||_0 + 1.$$

Define the variance estimate of $\sqrt{n}(\widehat{ATE}_{\text{Lasso}} - ATE)$ as follows:

$$\hat{\sigma}^2_{\text{Lasso}} = \frac{n}{n_A}\hat{\sigma}^2_{e^{(a)}} + \frac{n}{n_B}\hat{\sigma}^2_{e^{(b)}}. \tag{2.21}$$

We show in Theorem 2 that the limit of $\hat{\sigma}^2_{\text{Lasso}}$ is greater than or equal to the asymptotic variance of $\sqrt{n}(\widehat{ATE}_{\text{Lasso}} - ATE)$, and therefore can be used to construct a conservative confidence interval for the ATE. We will require one additional condition for this theorem.

Condition 7. For the Gram matrix Σ defined in condition 5, the largest eigenvalue is bounded away from ∞, that is, there exists a constant $\Lambda_{max} < \infty$ such that

$$\lambda_{max}(\Sigma) \leq \Lambda_{max}.$$

Theorem 2. *Assume conditions in Theorem 1 and condition 7 hold. Then* $\hat{\sigma}^2_{\text{Lasso}}$ *converges in probability to*

$$\frac{1}{p_A}\lim_{n\to\infty}\sigma^2_{e^{(a)}}+\frac{1}{1-p_A}\lim_{n\to\infty}\sigma^2_{e^{(b)}},$$

which is greater than or equal to the asymptotic variance of $\sqrt{n}(\widehat{ATE}_{\text{Lasso}}-ATE)$*. The difference is*

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\left[a_i-b_i-ATE-(\mathbf{x}_i-\bar{\mathbf{x}})^T(\boldsymbol{\beta}^{(a)}-\boldsymbol{\beta}^{(b)})\right]^2.$$

The proof of Theorem 2 can be found in section 2.C.

Remark 6. The Neyman-type conservative variance estimate for the unadjusted estimator is given by

$$\hat{\sigma}^2_{\text{unadj}}=\frac{n}{n_A}\frac{1}{n_A-1}\sum_{i\in A}(a_i-\bar{a}_A)^2+\frac{n}{n_B}\frac{1}{n_B-1}\sum_{i\in B}\left(b_i-\bar{b}_B\right)^2,$$

which, under second moment conditions of potential outcomes a and b, converges in probability to

$$\frac{1}{p_A}\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}(a_i-\bar{a})^2+\frac{1}{1-p_A}\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}(b_i-\bar{b})^2.$$

Therefore, for the $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$ defined in eq. (2.18), the limit of $\hat{\sigma}^2_{\text{Lasso}}$ is no greater than that of $\hat{\sigma}^2_{\text{unadj}}$ and the difference is

$$-\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\frac{1}{p_A}\left[(\mathbf{x}_i-\bar{\mathbf{x}})^T(\boldsymbol{\beta}^{(a)})\right]^2+\frac{1}{1-p_A}\left[(\mathbf{x}_i-\bar{\mathbf{x}})^T(\boldsymbol{\beta}^{(b)})\right]^2.$$

Remark 7. With the conservative variance estimate in Theorem 2, the Lasso adjusted confidence interval is also valid for the PATE (Population Average Treatment Effect) if there is a super population of size N with $N>n$.

Remark 8. The extra condition 7 is used to obtain the following bounds for the number of selected covariates by the Lasso: $\max(\hat{s}^{(a)},\hat{s}^{(b)})=o_p(\min(n_A,n_B))$. Condition 7 can be removed from Theorem 2 if we redefine $\hat{\sigma}^2_{e^{(a)}}$ and $\hat{\sigma}^2_{e^{(b)}}$ without adjusting the degrees of freedom, i.e.,

$$(\hat{\sigma}^*)^2_{e^{(a)}}=\frac{1}{n_A}\sum_{i\in A}\left(a_i-\bar{a}_A-(\mathbf{x}_i-\bar{\mathbf{x}}_A)^T\hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}}\right)^2,$$

$$(\hat{\sigma}^*)^2_{e^{(b)}}=\frac{1}{n_B}\sum_{i\in B}\left(b_i-\bar{b}_B-(\mathbf{x}_i-\bar{\mathbf{x}}_B)^T\hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}}\right)^2,$$

and define $(\hat{\sigma}^*)^2_{\text{Lasso}}=\frac{n}{n_A}(\hat{\sigma}^*)^2_{e^{(a)}}+\frac{n}{n_B}(\hat{\sigma}^*)^2_{e^{(b)}}$. It follows from the bounds for $\max(\hat{s}^{(a)},\hat{s}^{(b)})$ that $(\hat{\sigma}^2_{e^{(a)}},\hat{\sigma}^2_{e^{(b)}})$ and $((\hat{\sigma}^*)^2_{e^{(a)}},(\hat{\sigma}^*)^2_{e^{(b)}})$ have the same asymptotic property.

Theorem 3. *Assume the conditions in Theorem 1 hold. Then* $(\hat{\sigma}^*)^2_{\text{Lasso}}$ *converges in probability to*

$$\frac{1}{p_A}\lim_{n\to\infty}\sigma^2_{e^{(a)}}+\frac{1}{1-p_A}\lim_{n\to\infty}\sigma^2_{e^{(b)}}.$$

The proof of Theorem 3 can be found in section 2.C.

Remark 9. Though $(\hat{\sigma}^*)^2_{\text{Lasso}}$ has the same limit as $\hat{\sigma}^2_{\text{Lasso}}$, our simulation experience shows that, in finite samples, the confidence intervals based on $(\hat{\sigma}^*)^2_{\text{Lasso}}$ may yield low coverage probabilities (e.g., the coverage probability for 95% confidence interval can be only 80%). Hence, we recommend readers to use $\hat{\sigma}^2_{\text{Lasso}}$ in practice.

2.6 Related work

The Lasso has already made several appearances in the literature on treatment effect estimation. In the context of observational studies, Zhang and Zhang [103] constructs confidence intervals for preconceived effects or their contrasts by de-biasing the Lasso adjusted regression, Belloni et al. [7] employs the Lasso as a formal method for selecting adjustment variables via a two-stage procedure which concatenates features from models for treatment and outcome, and similarly, Belloni et al. [5] gives very general results for estimating a wide range of treatment effect parameters, including the case of instrumental variables estimation.

In addition to the Lasso, Li et al. [53] considers nonparametric adjustments in the estimation of ATE. In works such as these, which deal with observational studies, confounding is the major issue. With confounding, the naive difference-in-means estimator is biased for the true treatment effect, and adjustment is used to form an unbiased estimator. However, in our work, which focuses on a randomized trial, the difference-in-means estimator is already unbiased; adjustment reduces the variance while, in fact, introducing a small amount of finite-sample bias. Another major difference between this prior work and ours is the sampling framework: we operate within the Neyman-Rubin model with fixed potential outcomes for a finite population, where the treatment group is sampled without replacement, while these papers assume independent sampling from a probability distribution with random error terms.

Our work is related to the estimation of heterogeneous or subgroup-specific treatment effects; including interaction terms to allow the imputed individual-level treatment effects to vary according to some linear combination of covariates. This is pursued in the high-dimensional setting in Tian et al. [89]; this work advocates solving the Lasso on a reduced set of modified covariates, rather than the full set of covariate $\times$ treatment interactions, and includes extensions to binary outcomes and survival data. The recent work in Rosenblum et al. [80] considers the problem of designing multiple-testing procedures for detecting subgroup-specific treatment effects; they pose this as an optimization over testing procedures where constraints are added to enforce guarantees on type-I error rate and power to detect effects. Again, the sampling framework in these works is distinct from ours; they do not use the Neyman-Rubin model as a basis for designing the methods or investigating their properties.

2.7 PAC data illustration and simulations

We now return to the PAC-man study introduced earlier. We examine the data in more detail and explore the results of several adjustment procedures. There were 1013 patients in the PAC-man study: 506 treated (managed with PAC) and 507 control (managed without PAC, but retaining the option of using alternative devices). The outcome variable is quality-adjusted life years (QALYs). One QALY represents one year of life in full health; in-hospital death corresponds to a QALY of zero. We have 59 covariates about each individual in the study; we include all main effects as well as 1113 two-way interactions, and form a design matrix $\mathbf{X}$ with 1172 columns and 1013 rows. See section 2.F for more details on the design matrix.

The assumptions that underpin the theoretical guarantees of the $\widehat{ATE}_{\text{Lasso}}$ estimator are, in practice, not explicitly checkable, but we attempt to inspect the quantities that are involved in the conditions to help readers make their own judgement. The uniform bounds on the fourth moments refer to a hypothetical sequence of populations; these cannot be verified given that the investigator has a single dataset. However, as an approximation, the fourth moments of the data can be inspected to ensure that they are not too large. In this data set, the maximum fourth moment of the covariates is 37.3, which is indicative of a heavy-tailed and potentially destabilizing covariate; however, it occurs in an interaction term not selected by the lasso, and thus does not influence the estimate[3]. Checking the conditions for high-dimensional consistency of the Lasso would require knowledge of the unknown active set S, and moreover, even if it were known, calculating the cone invertibility factor would involve an infeasible optimization. This is a general issue in the theory of sparse linear high-dimensional estimation. To approximate these conditions, we use the bootstrap to estimate the active set of covariates S and the error terms $e^{(a)}$ and $e^{(b)}$. See section 2.G for more details. Our estimated S contains 16 covariates and the estimated second moments of $e^{(a)}$ and $e^{(b)}$ are 11.8 and 12.0, respectively. The estimated maximal covariance δ_n equals 0.34 and the scaling $(s \log p)/\sqrt{n}$ is 3.55. While this is not close to zero, we should mention that the estimation of δ_n and $(s \log p)/\sqrt{n}$ can be unstable and less accurate since it is based on a subsample of the population. As an approximation to condition 5, we examine the largest and smallest eigenvalues of the sub-Gram matrix $(1/n)\mathbf{X}_S^T\mathbf{X}_S$, which are 2.09 and 0.18 respectively. Thus the quantity in condition 5 seems reasonably bounded away from zero.

We now estimate the ATE using the unadjusted estimator, the Lasso adjusted estimator and the OLS adjusted estimator which is computed based on a sub-design matrix containing only the 59 main effects. We also present results for the two-step estimator $\widehat{ATE}_{\text{Lasso+OLS}}$ which adopts the Lasso to select covariates and then uses OLS to refit the regression coefficients. In the next paragraph and in algorithm 1, we show how we adapt the cross-validation

[3]The fourth moments of the covariates are shown in fig. 2.G.1. The covariates with the largest two fourth moments (37.3 and 34.9 respectively) are quadratic term $interactnew^2$ and interaction term $IMscorerct : systemnew$. Neither of them are selected by the Lasso to do the adjustment.

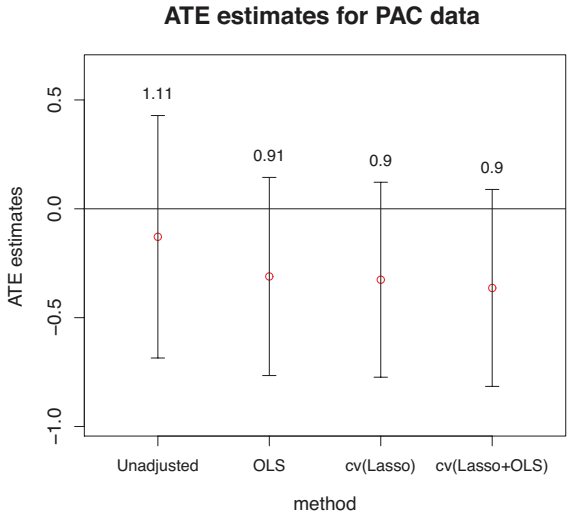

FIGURE 2.2: **ATE estimates (red circles) and 95% confidence intervals (bars) for the PAC data.** The numbers above each bar are the corresponding interval lengths.

procedure to select the tuning parameter for $\widehat{ATE}_{\text{Lasso+OLS}}$ based on a combined performance of Lasso and OLS, or cv(Lasso+OLS).

We use the R package "glmnet" to compute the Lasso solution path and select the tuning parameters λ_a and λ_b by 10-fold Cross Validation (CV). To indicate the method of selecting tuning parameters, we denote the corresponding estimators as cv(Lasso) and cv(Lasso+OLS) respectively. We should mention that for the cv(Lasso+OLS) adjusted estimator, we compute the CV error for a given value of λ_a (or λ_b) based on the whole Lasso+OLS procedure instead of just the Lasso estimator (see algorithm 1). Therefore, the cv(Lasso+OLS) and the cv(Lasso) may select different covariates to do the adjustment. This type of cross validation requires more computation than the cross validation based on just the Lasso estimator since it needs to compute the OLS estimator for each fold and each given λ_a (or λ_b), but it can give better prediction and model selection performance.

Figure 2.2 presents the ATE estimates along with 95% confidence intervals (CI). The interval lengths are shown on top of each interval bar. All the methods give confidence intervals containing 0; hence, this experiment failed to provide sufficient evidence to reject the hypothesis that PAC did not have an effect on patient QALYs (either positive or negative). Since the caretakers of patients managed without PAC retained the option of using less invasive cardiac output monitoring devices, such an effect may have been particularly hard to detect in this experiment.

However, it is interesting to note that, compared with the unadjusted estimator, the OLS adjusted estimator causes the ATE estimate to decrease (from -0.13 to -0.31), and shortens the confidence interval by about 20%. This is due mainly to the imbalance in the pre-treatment probability of death, which was highly predictive of the post-treatment QALYs. The cv(Lasso) adjusted estimator yields a comparable ATE estimate and confidence interval,

TABLE 2.1: **Selected covariates for adjustment.**

Method	**Treatment**	**Covariates**
cv(Lasso+OLS)	treated	age, p_death, age^2, age:p_death
cv(Lasso+OLS)	control	age, p_death, age^2, age:p_death, p_death:mech_vent
cv(Lasso)	treated	pac_rate, age, p_death, age^2, p_death^2, region:im_score, region:systemnew, pac_rate:age, pac_rate:p_death, pac_rate:systemnew, im_score:interactnew, age:p_death, age:glasgow, age:systemnew, interactnew:systemnew, pac_rate:creatinine, age:mech_vent, age:respiratory, age:creatinine, interactnew:mech_vent, interactnew:male, glasgow:organ_failure, p_death:mech_vent, systemnew:male
cv(Lasso)	control	age, p_death, age^2, unitsize:p_death, pac_rate:systemnew, age:p_death, interactnew:mech_vent, p_death:mech_vent*

* Covariate definitions: age (patient's age); p_death (baseline probability of death); mech_vent (mechanical ventilation at admission); region (geographic region); pac_rate (PAC rate in unit); creatinine, respiratory, glasgow, interactnew, organ_failure, systemnew, im_score (various physiological indicators).

but the fitted model is more interpretable and parsimonious than the OLS model: it selects 24 and 8 covariates for treated and control, respectively. The cv(Lasso+OLS) estimator selects even fewer covariates: 4 and 5 for treated and control, respectively, but performs a similar adjustment as the cv(Lasso). We also note that these adjustments agree with the one performed in Miratrix et al. [66], where the treatment effect was adjusted downwards to -0.27 after stratifying into 4 groups based on predicted probability of death.

The covariates selected by Lasso for adjustment are shown in table 2.1, where "A^2" denotes the quadratic term of the covariate A and "A:B" denotes a two way interaction between covariates A and B. Among them, patient's age and estimated probability of death (p_death), together with the quadratic term "age^2" and interactions "age:p_death" and "p_death:mech_vent" (mechanical ventilation at admission), are the most important covariates for the adjustment. The patients in control group are slightly older and have slightly higher risk of death. These covariates are important predictors of the outcome. Therefore, the unadjusted estimator may overestimate the benefits of receiving PAC.

Since not all the potential outcomes are observed, we cannot know the true gains of adjustment methods. However, we can estimate the gains via building a simulated set of potential outcomes by matching treated units to control units on observed covariates. We use the matching method described in Diamond and Sekhon [29] which gives 1013 observations with all potential outcomes imputed. We match on the 59 main effects only. The ATE is -0.29. We then use this synthetic data set to calculate the biases, standard deviations (SD)

TABLE 2.2: **Statistics for the PAC synthetic data set.**

						No. of selected covariates	
	Bias	SD	$\sqrt{\text{MSE}}$	Covg. (%)	Length	treated	control
unadjusted	0.001(0)*	0.20(0.02)	0.20(0.02)	99	1.06	-	-
OLS	0.002(0)	0.18(0.02)	0.18(0.02)	99	0.95	-	-
cv(Lasso)	0.001(0)	**0.17(0.02)**	**0.17(0.02)**	99	**0.94**	25(23)	15(14)
cv(Lasso+OLS)	**0.000(0)**	**0.17(0.02)**	**0.17(0.02)**	99	0.95	**6(6)**	**4(3)**

* The numbers in parentheses are the corresponding standard errors estimated by using the bootstrap with $B = 500$ resamplings of the ATE estimates.

and root-mean square errors ($\sqrt{\text{MSE}}$) of different ATE estimators based on 25000 replicates of completely randomized experiment which assigns 506 subjects to the treated group and the remainders to the control group.

Table 2.2 shows the results. For all the methods, the bias is substantially smaller (by a factor of 100) than the SD. The SD and $\sqrt{\text{MSE}}$ of the OLS adjusted estimator are both 10.2% smaller than those of the unadjusted estimator, while the cv(Lasso) and cv(Lasso+OLS) adjusted estimators further improve the SD and $\sqrt{\text{MSE}}$ of the OLS adjusted estimator by approximately 4.7%. Moreover, all these methods provide conservative confidence intervals with coverage probabilities higher than 99%. However, the interval lengths of the OLS, cv(Lasso) and cv(Lasso+OLS) adjusted estimator are comparable and are approximately 10% shorter than that of the unadjusted estimator. The cv(Lasso+OLS) adjusted estimator is similar to the cv(Lasso) adjusted estimator in terms of mean squared error, confidence interval length and coverage probability, but outperforms the latter with much fewer and more stable covariates in the adjustment (see fig. 2.3 for the selection frequency of each covariate for treatment group and control group). We show in fig. 2.4 that the sampling distribution of the estimates is very close to Normal.

We conduct additional simulation studies to evaluate the finite sample performance of $\widehat{ATE}_{\text{Lasso}}$ and compare it with that of the OLS adjusted estimator and the unadjusted estimator. A qualitative analysis of these simulations yields the same conclusions as presented above; see section 2.E for details.

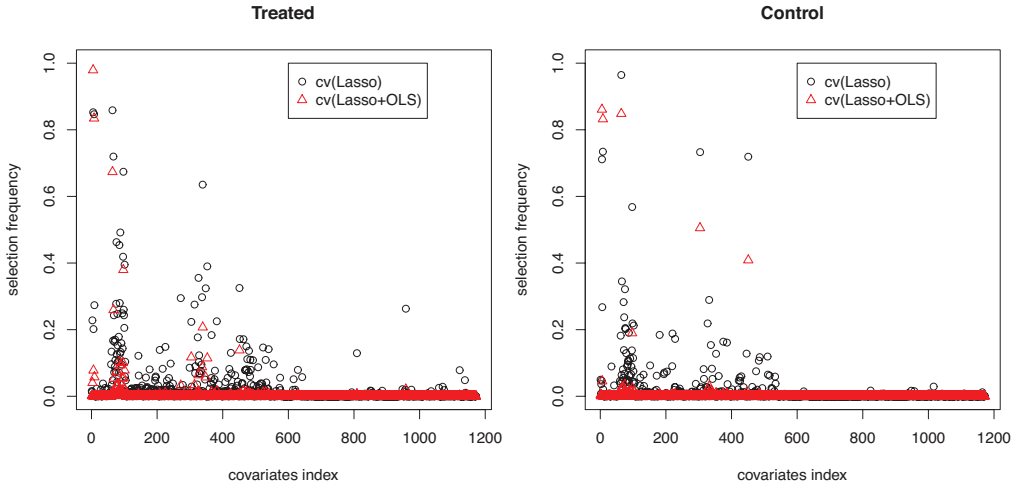

FIGURE 2.3: **Selection stability comparison of cv(Lasso) and cv(Lasso+OLS)** The left plot shows the treatment group, and the right plot shows the control group.

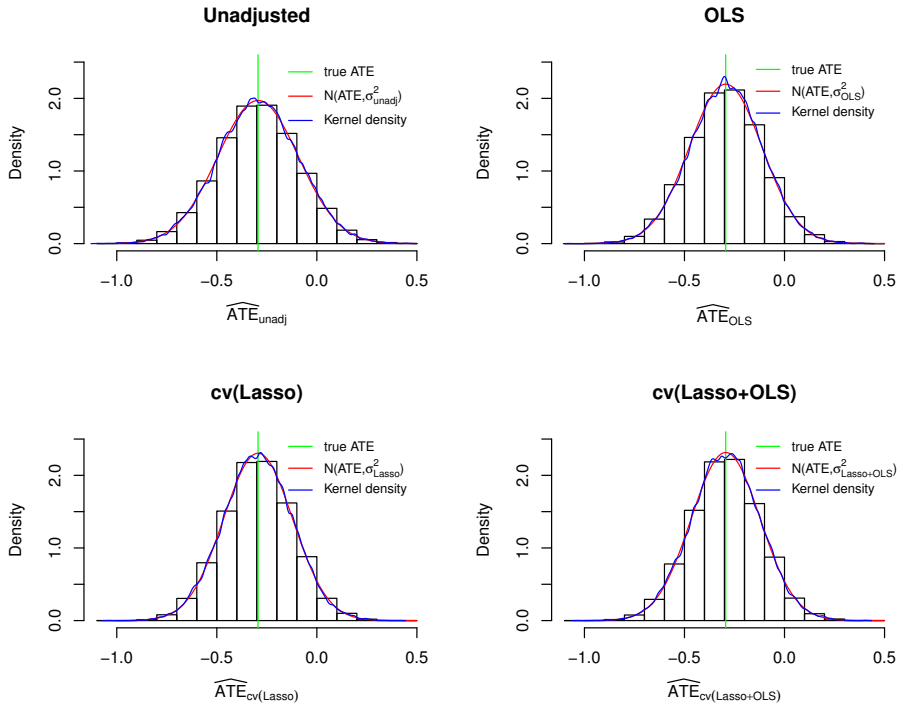

FIGURE 2.4: **Histograms of ATE estimates.** The green vertical lines are the true ATE; the red curves are the densities of a normal distribution; the blue curves are the kernel density estimate. The blue curves are very close to the red ones meaning that all the ATE estimates follow normal distribution.

2.8 Discussion

We study the Lasso adjusted average treatment effect (ATE) estimate under the Neyman-Rubin model for randomization. Our purpose in using the Neyman-Rubin model is to investigate the performance of the Lasso under a realistic sampling framework which does not impose strong assumptions on the data. We provide conditions that ensure asymptotic normality, and provide a Neyman-type estimate of the asymptotic variance which can be used to construct a conservative confidence interval for the ATE. While we do not require an explicit generative linear model to hold, our theoretical analysis requires the existence of latent 'adjustment vectors' such that moment conditions of the error terms are satisfied, and that the cone invertibility condition of the sample covariance matrix is satisfied in addition to moment conditions for OLS adjustment as in Lin [54]. Both assumptions are difficult to check in practice. In our theory, we do not address whether these assumptions are necessary for our results to hold, though simulations indicate that the moment conditions cannot be substantially weakened. As a by-product of our analysis, we extend Massart's concentration inequality for sampling without replacement, which is useful for theoretical analysis under the Neyman-Rubin model. Simulation studies and the real data illustration show the advantage of the Lasso-adjusted estimator in terms of estimation accuracy and model interpretation. In practice, we recommend a variant of Lasso, cv(Lasso+OLS), to select covariates and perform the adjustment, since it gives similar coverage probability and confidence interval length when compared with cv(Lasso), but with far fewer covariates selected. In future work, we plan to extend our analysis to other popular methods in high-dimensional statistics such as Elastic-Net and ridge regression, which may be more appropriate for estimating adjusted ATE under different assumptions.

The main goal of using Lasso in this chapter is to reduce the variance (and overall mean squared error) of ATE estimation. Another important task is to estimate heterogenous treatment effects and provide conditional treatment effect estimates for subpopulations. When the Lasso models of treatment and control outcomes are different, both in variables selected and coefficient values, this could be interpreted as modeling treatment effect heterogeneity in terms of covariates. However, reducing variance of the ATE estimate and estimating heterogenous treatment effects have completely different targets. Targeting heterogenous treatment effects may result in more variable ATE estimates. Moreover, our simulations show that the set of covariates selected by the Lasso is unstable and this may cause problems when interpreting them as evidence of heterogenous treatment effects. How best to estimate such effects is an open question that we would like to study in future research.

Acknowledgements

We thank David Goldberg for helpful discussions, Rebecca Barter for copyediting and suggestions for clarifying the text, and Winston Lin for comments. We thank Richard Grieve (LSHTM), Sheila Harvey (LSHTM), David Harrison (ICNARC) and Kathy Rowan (ICNARC) for access to data from the PAC-Man CEA and the ICNARC CMP database.

Appendix

2.A Proof of Theorem 1

In this section, we will prove Theorem 1 under a weaker sparsity condition than that given in section 2.4.

Definition 3. We define an approximate sparsity measure. Given the regularization parameter λ_a, λ_b and $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$, the sparsity measures for treatment and control groups, $s^{(a)}_{\lambda_a}$ and $s^{(b)}_{\lambda_b}$ are defined as

$$s^{(a)}_{\lambda_a} = \sum_{j=1}^{p} \min\left\{\frac{|\beta^{(a)}_j|}{\lambda_a}, 1\right\}, \; s^{(b)}_{\lambda_b} = \sum_{j=1}^{p} \min\left\{\frac{|\beta^{(b)}_j|}{\lambda_b}, 1\right\}, \tag{2.22}$$

respectively. We will allow $s^{(a)}_{\lambda_a}$ and $s^{(b)}_{\lambda_b}$ to grow with n, though the notation does not explicitly show this. Note that this is weaker than strict sparsity, as it allows $\boldsymbol{\beta}^{(a)}$ and $\boldsymbol{\beta}^{(b)}$ to have many small non-zero entries.

First, we restate conditions 4–6 from above, substituting this approximate sparsity measure.

Condition 8. Decay and scaling. Let $s_\lambda = \max\left\{s^{(a)}_{\lambda_a}, s^{(b)}_{\lambda_b}\right\}$,

$$\delta_n = o\left(\frac{1}{s_\lambda \sqrt{\log p}}\right), \tag{2.23}$$

$$(s_\lambda \log p)/\sqrt{n} = o(1). \tag{2.24}$$

Condition 9. Cone invertibility factor. Define the Gram matrix as $\Sigma = n^{-1}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$. There exist constants $C > 0$ and $\xi > 1$ not depending on n, such that

$$\|\mathbf{h}_S\|_1 \leq C s_\lambda \|\Sigma \mathbf{h}\|_\infty, \; \forall \mathbf{h} \in \mathcal{C}, \tag{2.25}$$

with $\mathcal{C} \stackrel{\text{def}}{=} \{\mathbf{h} : \|\mathbf{h}_{S^c}\|_1 \leq \xi \|\mathbf{h}_S\|_1\}$, and

$$S \stackrel{\text{def}}{=} \{j : |\beta^{(a)}_j| > \lambda_a \text{ or } |\beta^{(b)}_j| > \lambda_b\}. \tag{2.26}$$

Condition 10. Let $\tau = \min\left\{1/70, (3p_A)^2/70, (3-3p_A)^2/70\right\}$. For constants $0 < \eta < \frac{\xi-1}{\xi+1}$ and $0 < M < \infty$, assume the regularization parameters of the Lasso belong to the sets

$$\lambda_a \in (\frac{1}{\eta}, M] \times \left(\frac{2(1+\tau)L^{1/2}}{p_A} \sqrt{\frac{2\log p}{n}} + \delta_n \right), \tag{2.27}$$

$$\lambda_b \in (\frac{1}{\eta}, M] \times \left(\frac{2(1+\tau)L^{1/2}}{p_B} \sqrt{\frac{2\log p}{n}} + \delta_n \right). \tag{2.28}$$

Condition 11. Suppose there exist $\boldsymbol{\beta}^{(a)}$, $\boldsymbol{\beta}^{(b)}$, λ_a and λ_b such that conditions 1–3 and 8–10 hold simultaneously.

It is easy to verify that conditions 8–11 are implied by conditions 4–6 and the assumption stated in Theorem 1. We will prove Theorem 1 under the weaker conditions 8–11. For ease of notation, we will omit the subscripts in $\hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso}}$, $\hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso}}$, s_λ, $s^{(a)}_{\lambda_a}$ and $s^{(b)}_{\lambda_b}$. Note that we can assume, without loss of generality, that

$$\bar{a} = 0,\ \bar{b} = 0,\ \bar{\mathbf{x}} = \mathbf{0}. \tag{2.29}$$

Otherwise, we can consider $\breve{a}_i = a_i - \bar{a}$, $\breve{b}_i = b_i - \bar{b}$ and $\breve{\mathbf{x}}_i = \mathbf{x}_i - \bar{\mathbf{x}}$. Thus, we assume $\text{ATE} = \bar{a} - \bar{b} = 0$ and the definition of $\widehat{ATE}_{\text{Lasso}}$ becomes

$$\widehat{ATE}_{\text{Lasso}} = \left[\bar{a}_A - (\bar{\mathbf{x}}_A)^T \hat{\boldsymbol{\beta}}^{(a)}\right] - \left[\bar{b}_B - (\bar{\mathbf{x}}_B)^T \hat{\boldsymbol{\beta}}^{(b)}\right]. \tag{2.30}$$

We now proceed with the proof of Theorem 1.

Proof. Recall the decompositions of the potential outcomes given in eqs. (2.3) and (2.4) If we define $\mathbf{h}^{(a)} = \hat{\boldsymbol{\beta}}^{(a)} - \boldsymbol{\beta}^{(a)}$, $\mathbf{h}^{(b)} = \hat{\boldsymbol{\beta}}^{(b)} - \boldsymbol{\beta}^{(b)}$, by substitution, we have

$$\sqrt{n}(\widehat{ATE}_{\text{Lasso}} - ATE) = \underbrace{\sqrt{n}\left[\bar{e}^{(a)}_A - \bar{e}^{(b)}_B\right]}_{*} - \underbrace{\sqrt{n}\left[(\bar{\mathbf{x}}_A)^T \mathbf{h}^{(a)} - (\bar{\mathbf{x}}_B)^T \mathbf{h}^{(b)}\right]}_{**}.$$

We will analyze these two terms separately, showing that $(*)$ is asymptotically normal with mean 0 and variance given by (2.17), and that $(**)$ is $o_p(1)$.

Asymptotic normality of $(*)$ follows from the Theorem 1 in Freedman [34] with a and b replaced by $e^{(a)}$ and $e^{(b)}$ respectively. To bound $(**)$, we will apply Hölder's inequality to each of the terms. We will focus on the term involving the treatment group A, but exact same analysis is applied to the control group B. We have the bound

$$\left|(\bar{\mathbf{x}}_A)^T \mathbf{h}^{(a)}\right| \leq \|\bar{\mathbf{x}}_A\|_\infty \|\mathbf{h}^{(a)}\|_1. \tag{2.31}$$

We bound the two terms on the right hand side of (2.31) by the following Lemmas 1 and 2, respectively. Using these two Lemmas, it is easy to show that $(**) = \sqrt{n} \cdot O_p\left(\sqrt{\frac{\log p}{n}}\right) \cdot o_p\left(\frac{1}{\sqrt{\log p}}\right) = o_p(1)$. □

Lemma 1. *Under the moment condition of eq. (2.6), if we let* $c_n = \frac{(1+\tau)L^{1/4}}{p_A}\sqrt{\frac{2\log p}{n}}$, *then as* $n \to \infty$,

$$P\left(\|\bar{\mathbf{x}}_A\|_\infty > c_n\right) \to 0$$

Thus, $\|\bar{\mathbf{x}}_A\|_\infty = O_p\left(\sqrt{\frac{\log p}{n}}\right)$.

Proof. Let $c_n = \frac{(1+\tau)L^{1/4}}{p_A}\sqrt{\frac{2\log p}{n}}$. By the union bound,

$$P\left(\|\bar{\mathbf{x}}_A\|_\infty > c_n\right) = P\left(\max_{j=1,\dots,p}\left|\frac{1}{n_A}\sum_{i\in A} x_{ij}\right| > c_n\right) \le \sum_{j=1}^{p} P\left(\left|\frac{1}{n_A}\sum_{i\in A} x_{ij}\right| > c_n\right). \tag{2.32}$$

By Cauchy-Schwarz inequality, we have

$$\frac{1}{n}\sum_{i=1}^{n} x_{ij}^2 \le \left(\frac{1}{n}\sum_{i=1}^{n} x_{ij}^4\right)^{\frac{1}{2}}\left(\frac{1}{n}\sum_{i=1}^{n} 1^2\right)^{\frac{1}{2}} \le \sqrt{L}. \tag{2.33}$$

Substituting the concentration inequality (2.71) into (2.32),

$$P\left(\|\bar{\mathbf{x}}_A\|_\infty > c_n\right) \le 2\exp\left\{\log p - \frac{p_A n_A c_n^2}{(1+\tau)^2 L^{1/2}}\right\} = 2\exp\left\{-\log p\right\} \to 0.$$

□

Lemma 2. *Assume conditions 8–11 hold. Then* $\|\mathbf{h}^{(a)}\|_1 = o_p\left(\frac{1}{\sqrt{\log p}}\right)$.

Proof. We start with the KKT condition, which characterizes the solution to the Lasso. Recall the definition of the Lasso estimator $\hat{\boldsymbol{\beta}}$:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{2n_A}\sum_{i\in A}\left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T\boldsymbol{\beta}\right)^2 + \lambda_a\|\boldsymbol{\beta}\|_1.$$

The KKT condition for $\hat{\boldsymbol{\beta}}$ is

$$\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)\left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T\hat{\boldsymbol{\beta}}\right) = \lambda_a\kappa, \tag{2.34}$$

where κ is the subgradient of $||\boldsymbol{\beta}||_1$ taking value at $\boldsymbol{\beta} = \hat{\boldsymbol{\beta}}$, i.e.,

$$\kappa \in \partial||\boldsymbol{\beta}||_1\Big|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}} \quad \text{with} \quad \begin{cases} \kappa_j \in [-1,1] \text{ for } j \text{ s.t. } \hat{\beta}_j = 0 \\ \kappa_j = \text{sign}(\hat{\beta}_j) \text{ otherwise} \end{cases} \tag{2.35}$$

Substituting a_i by the decomposition (2.3), (2.34) becomes

$$\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(\mathbf{x}_i-\bar{\mathbf{x}}_A)^T(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})+\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(e_i-\bar{e}_A)=\lambda_a\kappa. \tag{2.36}$$

Multiplying both sides of (2.36) by $-\mathbf{h}^T=(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T$, we have

$$\begin{aligned}&\frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i-\bar{\mathbf{x}}_A)^T\mathbf{h}\right)^2-\mathbf{h}^T\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(e_i-\bar{e}_A)\\ =\;&\lambda_a(\boldsymbol{\beta}-\hat{\boldsymbol{\beta}})^T\kappa\le\lambda_a\left(\|\boldsymbol{\beta}\|_1-\|\hat{\boldsymbol{\beta}}\|_1\right)\end{aligned}$$

where the last inequality is because

$$\boldsymbol{\beta}^T\kappa\le||\boldsymbol{\beta}||_1||\kappa||_\infty\le||\boldsymbol{\beta}||_1\quad\text{and}\quad\hat{\boldsymbol{\beta}}^T\kappa=||\hat{\boldsymbol{\beta}}||_1.$$

Rearranging and by Hölder's inequality, we have

$$\begin{aligned}\frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i-\bar{\mathbf{x}}_A)^T\mathbf{h}\right)^2&\le\lambda_a\left(\|\boldsymbol{\beta}\|_1-\|\hat{\boldsymbol{\beta}}\|_1\right)+\mathbf{h}^T\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(e_i-\bar{e}_A)\\&\le\lambda_a\left(\|\boldsymbol{\beta}\|_1-\|\hat{\boldsymbol{\beta}}\|_1\right)+\|\mathbf{h}\|_1\underbrace{\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(e_i-\bar{e}_A)\right\|_\infty}_{*}\end{aligned}$$

To control the term (∗), we define the event $\mathcal{L}=\{*\le\eta\lambda_a\}$. Lemma 3 shows that, with λ_a defined appropriately, $\mathcal{L}$ holds with probability approaching 1. On $\mathcal{L}$

$$\frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i-\bar{\mathbf{x}}_A)^T\mathbf{h}\right)^2\le\lambda_a\left(\|\boldsymbol{\beta}\|_1-\|\hat{\boldsymbol{\beta}}\|_1+\eta\|\mathbf{h}\|_1\right). \tag{2.37}$$

By substituting the definition of $\mathbf{h}$, and several applications of the triangle inequality, we have

$$\|\boldsymbol{\beta}\|_1-\|\hat{\boldsymbol{\beta}}\|_1\le\|\mathbf{h}_S\|_1-\|\mathbf{h}_{S^c}\|_1+2\|\boldsymbol{\beta}_{S^c}\|_1.$$

Therefore,

$$\begin{aligned}\frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i-\bar{\mathbf{x}}_A)^T\mathbf{h}\right)^2&\le\lambda_a\left(\|\mathbf{h}_S\|_1-\|\mathbf{h}_{S^c}\|_1+2\|\boldsymbol{\beta}_{S^c}\|_1+\eta\|\mathbf{h}\|_1\right)\\&\le\lambda_a\left((\eta-1)\|\mathbf{h}_{S^c}\|_1+(1+\eta)\|\mathbf{h}_S\|_1+2\|\boldsymbol{\beta}_{S^c}\|_1\right).\end{aligned}$$

Because $\frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i-\bar{\mathbf{x}}_A)^T\mathbf{h}\right)^2\ge0$, we obtain

$$(1-\eta)\|\mathbf{h}_{S^c}\|_1\le(1+\eta)\|\mathbf{h}_S\|_1+2\|\boldsymbol{\beta}_{S^c}\|_1\le(1+\eta)\|\mathbf{h}_S\|_1+2s\lambda_a. \tag{2.38}$$

where the last inequality is because of the definition of s in (2.22) and S in (2.26).

Consider the following two cases:

(I) If $(1+\eta)\|\mathbf{h}_S\|_1 + 2s\lambda_a \geq (1-\eta)\xi\|\mathbf{h}_S\|_1$ then by (2.38),

$$\|\mathbf{h}\|_1 = \|\mathbf{h}_S\|_1 + \|\mathbf{h}_{S^c}\|_1 \leq \left(\frac{1+\eta}{1-\eta} + 1\right)\|\mathbf{h}_S\|_1 + \frac{2s\lambda_a}{1-\eta} \leq \frac{2s\lambda_a}{1-\eta}\left(\frac{2}{(1-\eta)\xi - (1+\eta)} + 1\right).$$

By the definition of λ_a and the scaling assumptions (2.23), (2.24), we have that $s\lambda_a = o\left(\frac{1}{\sqrt{\log p}}\right)$.

(II) If $(1+\eta)\|\mathbf{h}_S\|_1 + 2s\lambda_a < (1-\eta)\xi\|\mathbf{h}_S\|_1$ then by (2.38) we have $\|\mathbf{h}_{S^c}\|_1 \leq \xi\|\mathbf{h}_S\|_1$. Applying the cone invertibility condition on the design matrix (2.25),

$$\|\mathbf{h}\|_1 = \|\mathbf{h}_S\|_1 + \|\mathbf{h}_{S^c}\|_1 \leq (1+\xi)\|\mathbf{h}_S\|_1 \leq (1+\xi)Cs\left\|\frac{1}{n}X^TX\mathbf{h}\right\|_\infty \tag{2.39}$$

Before applying this inequality we will revisit the KKT condition (2.35), but this time we will take the l_∞-norm, yielding

$$\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T\mathbf{h}\right\|_\infty \leq \lambda_a + \left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(e_i - \bar{e}_A)\right\|_\infty \leq (1+\eta)\lambda_a \tag{2.40}$$

where the latter inequality holds on the set $\mathcal{L}$. The final step is to control the deviation of the subsampled covariance matrix from the population covariance matrix, so that we can apply (2.39). We define another event with constant $C_1 = \frac{2(1+\tau)L^{1/2}}{p_A}$

$$\mathcal{M} = \left\{\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T - \frac{1}{n}X^TX\right\|_\infty \leq C_1\sqrt{\frac{\log p}{n}}\right\}$$

Lemma 4 shows that $P(\mathcal{M}) \to 1$. Continuing our inequalities, on the event $\mathcal{L}\cap\mathcal{M}$,

$$\begin{aligned} s\left\|\frac{1}{n}X^TX\mathbf{h}\right\|_\infty &\leq C_1 s\sqrt{\frac{\log p}{n}}\|\mathbf{h}\|_1 + s\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T\mathbf{h}\right\|_\infty \\ &\leq o(1)\|\mathbf{h}\|_1 + s(1+\eta)\lambda_a, \end{aligned}$$

where we have applied the scaling assumption (2.24) and (2.40) in the second line. Hence, by (2.39),

$$\|\mathbf{h}\|_1 \leq (1+\xi)C\left[o(1)\|\mathbf{h}\|_1 + s(1+\eta)\lambda_a\right].$$

Again, applying the scaling assumptions (2.23) and (2.24), we get $\|\mathbf{h}\|_1 = o_p\left(\frac{1}{\sqrt{\log p}}\right)$. □

Lemma 3. *Define* $\mathcal{L} = \left\{\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(e_i - \bar{e}_A)\right\|_\infty \leq \eta\lambda_a\right\}$. *Then under the conditions of Theorem 1,* $P(\mathcal{L}) \to 1$.

Proof. It is easy to verify that

$$\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(e_i-\bar{e}_A)=\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i-(\bar{\mathbf{x}}_A)(\bar{e}_A).$$

Hence,

$$\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i-\bar{\mathbf{x}}_A)(e_i-\bar{e}_A)\right\|_\infty\leq\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i\right\|_\infty+\left\|(\bar{\mathbf{x}}_A)(\bar{e}_A)\right\|_\infty. \tag{2.41}$$

We analyze these two terms on the right hand side of the inequality separately. For the first term, by triangle inequality and the definition of δ_n in (2.9),

$$\begin{aligned}\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i\right\|_\infty&\leq\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i-\frac{1}{n}\sum_{i=1}^n\mathbf{x}_i e_i\right\|_\infty+\left\|\frac{1}{n}\sum_{i=1}^n\mathbf{x}_i e_i\right\|_\infty\\&\leq\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i-\frac{1}{n}\sum_{i=1}^n\mathbf{x}_i e_i\right\|_\infty+\delta_n.\end{aligned} \tag{2.42}$$

We will again bound (2.42) by the concentration inequality of Lemma 7. By Cauchy-Schwarz inequality, we have for any $j=1,..,p$,

$$\frac{1}{n}\sum_{i=1}^n x_{ij}^2e_i^2\leq\left(\frac{1}{n}\sum_{i=1}^n x_{ij}^4\right)^{\frac{1}{2}}\left(\frac{1}{n}\sum_{i=1}^n e_i^4\right)^{\frac{1}{2}}\leq L.$$

Let $t_n=\frac{(1+\tau)L^{1/2}}{p_A}\sqrt{\frac{2\log p}{n}}$, then by the union bound and the concentration inequality (2.71),

$$\begin{aligned}P\left(\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i-\frac{1}{n}\sum_{i=1}^n\mathbf{x}_i e_i\right\|_\infty>t_n\right)&\leq 2\exp\left\{\log p-\frac{p_A n_A t_n^2}{(1+\tau)^2L)}\right\}\\&=2\exp\left\{-\log p\right\}\to 0.\end{aligned}$$

Taking this back to (2.42), we have

$$P\left(\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i e_i\right\|_\infty\leq t_n+\delta_n\right)\to 1. \tag{2.43}$$

For the second term, by Lemma 1, we have shown that,

$$P\left(\left\|\bar{\mathbf{x}}_A\right\|_\infty\leq\frac{(1+\tau)L^{1/4}}{p_A}\sqrt{\frac{2\log p}{n}}\right)\to 1.$$

Similar proof yields

$$P\left(\left\|\bar{e}_A\right\|_\infty\leq\frac{(1+\tau)L^{1/4}}{p_A}\sqrt{\frac{2\log p}{n}}\right)\to 1.$$

Hence, under the scaling condition (2.24),

$$P\left(\|(\bar{\mathbf{x}}_A)(\bar{e}_A)\|_\infty \leq \frac{(1+\tau)L^{1/2}}{p_A}\sqrt{\frac{2\log p}{n}}\right) \to 1. \tag{2.44}$$

Combining (2.43) and (2.44) yields

$$P\left(\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(e_i - \bar{e}_A)\right\|_\infty \leq \frac{2(1+\tau)L^{1/2}}{p_A}\sqrt{\frac{2\log p}{n}} + \delta_n\right) \to 1.$$

The conclusion follows from the condition $\lambda_a \in (\frac{1}{\eta}, M] \times \left(\frac{2(1+\tau)L^{1/2}}{p_A}\sqrt{\frac{2\log p}{n}} + \delta_n\right)$. □

Lemma 4. *Assume stability of treatment assignment probability condition 1 and moment condition 2 hold. Define*

$$\mathcal{M} = \left\{\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T - \frac{1}{n}X^TX\right\|_\infty \leq C_1\sqrt{\frac{\log p}{n}}\right\}$$

Then $P(\mathcal{M}) \to 1$.

Proof. It is easy to see that

$$\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T = \frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i\mathbf{x}_i^T - (\bar{\mathbf{x}}_A)(\bar{\mathbf{x}}_A)^T.$$

Then, by triangle inequality,

$$\left\|\frac{1}{n_A}\sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T - \frac{1}{n}X^TX\right\|_\infty \tag{2.45}$$

$$\leq \underbrace{\left\|\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i\mathbf{x}_i^T - \frac{1}{n}\sum_{i=1}^n\mathbf{x}_i\mathbf{x}_i^T\right\|_\infty}_{*} + \underbrace{\left\|(\bar{\mathbf{x}}_A)(\bar{\mathbf{x}}_A)^T\right\|_\infty}_{**}. \tag{2.46}$$

We control the first term $(*)$ again using the concentration inequality (2.71) and the union bound. By Cauchy-Schwarz inequality, for $j, k = 1, ..., p$,

$$\frac{1}{n}\sum_{i=1}^n x_{ij}^2x_{ik}^2 \leq \left(\frac{1}{n}\sum_{i=1}^n x_{ij}^4\right)^{\frac{1}{2}}\left(\frac{1}{n}\sum_{i=1}^n x_{ik}^4\right)^{\frac{1}{2}} \leq L.$$

Then,

$$P\left(||\frac{1}{n_A}\sum_{i\in A}\mathbf{x}_i\mathbf{x}_i^T - \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^T||_\infty \geq \frac{(1+\tau)L^{1/2}}{p_A}\sqrt{\frac{3\log p}{n}}\right)$$
$$\leq \quad 2\exp\left\{2\log p - \frac{3p_A n_A(1+\tau)^2 L\log p}{(1+\tau)^2 L p_A^2 n}\right\} = 2\exp\{-\log p\} \to 0. \tag{2.47}$$

The second term (∗∗) is bounded by again observing that, by Lemma 1 and the scaling condition (2.24),

$$(**) \leq ||\bar{\mathbf{x}}_A||_\infty^2 = o_p(\sqrt{\frac{\log p}{n}}). \tag{2.48}$$

Combining (2.47) and (2.48) yields the conclusion. □

2.B Proof of Corollary 1

Proof. By Theorem 1 in Freedman [34], the asymptotic variance of $\sqrt{n}\ \widehat{ATE}_{\text{unadj}}$ is $\frac{1-p_A}{p_A}\lim_{n\to\infty}\sigma_a^2 + \frac{p_A}{1-p_A}\lim_{n\to\infty}\sigma_b^2 + 2\lim_{n\to\infty}\sigma_{ab}$, so the difference is

$$\frac{1-p_A}{p_A}\lim_{n\to\infty}\left(\sigma_{e^{(a)}}^2 - \sigma_a^2\right) + \frac{p_A}{1-p_A}\lim_{n\to\infty}\left(\sigma_{e^{(b)}}^2 - \sigma_b^2\right) + 2\lim_{n\to\infty}\left(\sigma_{e^{(a)}e^{(b)}} - \sigma_{ab}\right).$$

We will analyze these three terms separately. Since $X\boldsymbol{\beta}^{(a)}$ and $X\boldsymbol{\beta}^{(b)}$ are the orthogonal projections of a and b onto the same subspace, we have

$$(X\boldsymbol{\beta}^{(a)})^T e^{(a)} = (X\boldsymbol{\beta}^{(a)})^T e^{(b)} = (X\boldsymbol{\beta}^{(b)})^T e^{(a)} = (X\boldsymbol{\beta}^{(b)})^T e^{(b)} = 0.$$

Simple calculations yield

$$\sigma_{e^{(a)}}^2 - \sigma_a^2 = \left\|e^{(a)}\right\|_2^2 - \|a\|_2^2 = -\left\|X\boldsymbol{\beta}^{(a)}\right\|_2^2, \tag{2.49}$$

$$\sigma_{e^{(b)}}^2 - \sigma_b^2 = \left\|e^{(b)}\right\|_2^2 - \|b\|_2^2 = -\left\|X\boldsymbol{\beta}^{(b)}\right\|_2^2, \tag{2.50}$$

$$\sigma_{e^{(a)}e^{(b)}} - \sigma_{ab} = (e^{(a)})^T(e^{(b)}) - a^T b = -(X\boldsymbol{\beta}^{(a)})^T(X\boldsymbol{\beta}^{(b)}) \tag{2.51}$$

Combining (2.49), (2.50) and (2.51), we obtain the corollary. □

2.C Proofs of Theorems 2 and 3

Preparatory Lemmas

Before proving Theorem 2, we present the following two lemmas. Lemma 5 bounds the number of selected covariates (covariates with a nonzero coefficient). Lemma 6 is an analogue of the weak law of large numbers for sampling without replacement.

Lemma 5. *Under conditions in Theorem 2, there exists a constant C, such that the following holds with probability going to 1:*

$$\hat{s}^{(a)} \leq Cs; \quad \hat{s}^{(b)} \leq Cs. \tag{2.52}$$

Proof. In the proof of Lemma 2, we have shown that, on $\mathcal{L}$ defined in Lemma 3,

$$\begin{aligned}\frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})\right)^2 &\leq \lambda_a\left(\|\boldsymbol{\beta}\|_1 - \|\hat{\boldsymbol{\beta}}\|_1 + \eta||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}||_1\right). \\ &\leq \lambda_a(1+\eta)||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}||_1. \end{aligned} \tag{2.53}$$

Let $\mathbf{x}^j$ be the j-th column of the design matrix X and $\bar{\mathbf{x}}_A^j = n_A^{-1}\sum_{i\in A} x_{ij}$. Again, by KKT conditon, we have

$$\left|\frac{1}{n_A}\sum_{i\in A}(x_{ij} - \bar{\mathbf{x}}_A^j)\left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T\hat{\boldsymbol{\beta}}\right)\right| = \lambda_a, \text{ if } \hat{\boldsymbol{\beta}}_j \neq 0.$$

Substituting a_i by the decomposition (2.3) yields

$$\left|\frac{1}{n_A}\sum_{i\in A}(x_{ij} - \bar{\mathbf{x}}_A^j)(e_i - \bar{e}_A) + \frac{1}{n_A}\sum_{i\in A}(x_{ij} - \bar{\mathbf{x}}_A^j)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})\right| = \lambda_a.$$

Combining with the definition of the event $\mathcal{L}$, we have if $\hat{\boldsymbol{\beta}}_j \neq 0$

$$\Delta_j := \left|\frac{1}{n_A}\sum_{i\in A}(x_{ij} - \bar{\mathbf{x}}_A^j)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})\right| \geq (1-\eta)\lambda_a. \tag{2.54}$$

Let $Z = (\mathbf{z}_1, ..., \mathbf{z}_n) \in R^{p\times n}$ with $\mathbf{z}_i = \mathbf{x}_i - \bar{\mathbf{x}}_A \in R^p$ and denote $\mathbf{w} = Z^T(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$, then

$$\frac{1}{n_A}||\mathbf{w}_A||_2^2 = \frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})\right)^2 \leq \lambda_a(1+\eta)||\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}||_1.$$

Let $Z_A = (\mathbf{z}_i : i \in A)$, since the largest eigenvalues of $Z_A^T Z_A$ and $Z_A Z_A^T$ are the same,

$$\begin{aligned}\frac{1}{n_A^2}\mathbf{w}_A^T Z_A^T Z_A \mathbf{w}_A &\leq \frac{1}{n_A^2}\lambda_{\max}(Z_A^T Z_A)||\mathbf{w}_A||_2^2 \\ &\leq \frac{1}{n_A}\lambda_{\max}(Z_A Z_A^T)\lambda_a(\eta+1)||\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}||_1 \\ &\leq \Lambda_{\max}\frac{n}{n_A}\lambda_a(1+\eta)||\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}||_1.\end{aligned}$$

The last inequality holds because

$$\begin{aligned}\lambda_{\max}(Z_A Z_A^T) &= \max_{\mathbf{u}:||\mathbf{u}||_2=1} \mathbf{u}^T Z_A Z_A^T \mathbf{u} \\ &= \max_{\mathbf{u}:||\mathbf{u}||_2=1} \mathbf{u}^T \sum_{i\in A}(\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \mathbf{u} \\ &= \max_{\mathbf{u}:||\mathbf{u}||_2=1} \mathbf{u}^T \sum_{i\in A}\mathbf{x}_i\mathbf{x}_i^T \mathbf{u} - n_A \mathbf{u}^T(\bar{\mathbf{x}}_A)(\bar{\mathbf{x}}_A)^T\mathbf{u} \\ &\leq \max_{\mathbf{u}:||\mathbf{u}||_2=1} \mathbf{u}^T \sum_{i\in A}\mathbf{x}_i\mathbf{x}_i^T \mathbf{u} \leq n\Lambda_{\max}. \qquad (2.55)\end{aligned}$$

On the other hand,

$$\frac{1}{n_A^2}\mathbf{w}_A^T Z_A^T Z_A \mathbf{w}_A = \sum_{j=1}^p \Delta_j^2 \geq \sum_{j:\hat{\beta}_j\neq 0}\Delta_j^2 \geq (1-\eta)^2\lambda_a^2\hat{s}. \qquad (2.56)$$

Combining (2.54), (2.56) and the fact that with probability going to 1 (see the proof of Lemma 2)

$$||\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}||_1 \leq Cs(1+\eta)\lambda_a,$$

where C is a constant. We conclude that with probability going to 1

$$\hat{s} \leq \frac{1}{(1-\eta)^2}\frac{1}{\lambda_a^2}\Lambda_{\max}\frac{n}{n_A}\lambda_a(1+\eta)Cs(1+\eta)\lambda_a \leq \frac{C(1+\eta)^2}{p_A(1-\eta)^2}s.$$

□

Lemma 6. *Let $\{z_i, i = 1, ..., n\}$ be a finite population of real numbers. Let $A \subset \{i, \ldots, n\}$ be a subset of deterministic size $|A| = n_A$ that is selected randomly without replacement. Suppose that the population mean of the z_i has a finite limit and that there exist constants $\epsilon > 0$ and $L < \infty$ such that*

$$\frac{1}{n}\sum_{i=1}^n |z_i|^{1+\epsilon} \leq L. \qquad (2.57)$$

If $\frac{n_A}{n} \to p_A \in (0,1)$, then

$$\bar{z}_A \xrightarrow{p} \lim_{n\to\infty}\bar{z}. \qquad (2.58)$$

Proof. For any $t > 0$, we have

$$P(|\bar{z}_A - \lim_{n\to\infty} \bar{z}| > t) \leq P(|\bar{z}_A - \bar{z}| > t/2) + P(|\bar{z} - \lim_{n\to\infty} \bar{z}| > t/2). \tag{2.59}$$

The second term in the right hand side of (2.59) obviously converges to 0 as $n \to \infty$. To bound the first term, we apply the concentration inequality (2.71). By (2.57), it is easy to show

$$\frac{1}{n}\sum_{i=1}^{n} z_i^2 = \frac{1}{n}\sum_{i=1}^{n} |z_i|^{1-\epsilon}|z_i|^{1+\epsilon} \leq (nL)^{\frac{1-\epsilon}{1+\epsilon}} \frac{1}{n}\sum_{i=1}^{n} |z_i|^{1+\epsilon} \leq L^{\frac{2}{1+\epsilon}} n^{\frac{1-\epsilon}{1+\epsilon}}.$$

Concentration inequality (2.71) yields

$$P(|\bar{z}_A - \bar{z}| > t/2) \leq 2\exp\left\{-\frac{p_A n_A t^2}{4(1+\tau)^2 L^{\frac{2}{1+\epsilon}} n^{\frac{1-\epsilon}{1+\epsilon}}}\right\} \to 0.$$

□

Proof of Theorem 2

Proof. To prove Theorem 2, it is enough to show that

$$\hat{\sigma}^2_{e^{(a)}} \xrightarrow{p} \lim_{n\to\infty} \sigma^2_{e^{(a)}}, \tag{2.60}$$

$$\hat{\sigma}^2_{e^{(b)}} \xrightarrow{p} \lim_{n\to\infty} \sigma^2_{e^{(b)}}. \tag{2.61}$$

We will only prove the statement (2.60) and omit the proof of the statement (2.61) since it is identical. By definition (2.20) and simple calculations,

$$\begin{aligned}
\hat{\sigma}^2_{e^{(a)}} &= \frac{1}{n_A - df^{(a)}} \sum_{i\in A} \left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \hat{\boldsymbol{\beta}}^{(a)}\right)^2 \\
&= \frac{1}{n_A - df^{(a)}} \sum_{i\in A} \left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \boldsymbol{\beta}^{(a)} + (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)})\right)^2 \\
&= \frac{1}{n_A - df^{(a)}} \sum_{i\in A} \left(a_i - \mathbf{x}_i^T \boldsymbol{\beta}^{(a)} - (\bar{a}_A - (\bar{\mathbf{x}}_A)^T \boldsymbol{\beta}^{(a)}) + (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)})\right)^2 \\
&= \frac{n_A}{n_A - df^{(a)}} \frac{1}{n_A} \sum_{i\in A} \left(e_i^{(a)} - \bar{e}_A^{(a)} + (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)})\right)^2 \\
&= \frac{n_A}{n_A - df^{(a)}} \left\{\frac{1}{n_A}\sum_{i\in A}\left(e_i^{(a)} - \bar{e}_A^{(a)}\right)^2 + \frac{1}{n_A}\sum_{i\in A}\left((\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)})\right)^2\right\} \\
&\quad + \frac{n_A}{n_A - df^{(a)}} \left\{\frac{1}{n_A}\sum_{i\in A}(e_i^{(a)} - \bar{e}_A^{(a)})(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T(\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)})\right\}.
\end{aligned}$$

The second to last equality is due to the decomposition of potential outcome a:

$$a_i = \mathbf{x}_i^T \boldsymbol{\beta}^{(a)} + e_i^{(a)}; \quad \bar{a}_A = (\bar{\mathbf{x}}_A)^T \boldsymbol{\beta}^{(a)} + \bar{e}_A^{(a)}.$$

It is easy to see that

$$\frac{1}{n_A} \sum_{i \in A} \left(e_i^{(a)} - \bar{e}_A^{(a)} \right)^2 = \frac{1}{n_A} \sum_{i \in A} (e_i^{(a)})^2 - (\bar{e}_A^{(a)})^2. \tag{2.62}$$

By the $4th$ moment condition on the approximation error $e^{(a)}$ (see (2.7)), and applying Lemma 6 gives

$$\frac{1}{n_A} \sum_{i \in A} (e_i^{(a)})^2 \xrightarrow{p} \lim_{n \to \infty} \sigma^2_{e^{(a)}}; \quad \bar{e}_A^{(a)} \xrightarrow{p} \lim_{n \to \infty} \bar{e}^{(a)} = 0.$$

Therefore,

$$\frac{1}{n_A} \sum_{i \in A} \left(e_i^{(a)} - \bar{e}_A^{(a)} \right)^2 \xrightarrow{p} \lim_{n \to \infty} \sigma^2_{e^{(a)}}. \tag{2.63}$$

By simple algebra,

$$\begin{aligned} & \frac{1}{n_A} \sum_{i \in A} \left((\mathbf{x}_i - \bar{\mathbf{x}}_A)^T (\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)}) \right)^2 \\ = \; & (\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)})^T \left[\frac{1}{n_A} \sum_{i \in A} (\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \right] (\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)}) \\ \leq \; & \left\| \boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)} \right\|_1^2 \cdot \left\| \frac{1}{n_A} \sum_{i \in A} (\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \right\|_\infty . \end{aligned} \tag{2.64}$$

We next show that (2.64) converges to 0 in probability. By Lemmas 2 and 4, we have

$$\left\| \boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)} \right\|_1 = \|\mathbf{h}^{(a)}\|_1 = o_p \left(\frac{1}{\sqrt{\log p}} \right), \tag{2.65}$$

$$\left\| \frac{1}{n_A} \sum_{i \in A} (\mathbf{x}_i - \bar{\mathbf{x}}_A)(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \right\|_\infty = O_p(1). \tag{2.66}$$

Therefore,

$$\frac{1}{n_A} \sum_{i \in A} \left((\mathbf{x}_i - \bar{\mathbf{x}}_A)^T (\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)}) \right)^2 \xrightarrow{p} 0. \tag{2.67}$$

By the Cauchy-Schwarz inequality,

$$\begin{aligned} & \left| \frac{1}{n_A} \sum_{i \in A} (e_i^{(a)} - \bar{e}_A^{(a)})(\mathbf{x}_i - \bar{\mathbf{x}}_A)^T (\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)}) \right| \\ \leq \; & \left[\frac{1}{n_A} \sum_{i \in A} \left(e_i^{(a)} - \bar{e}_A^{(a)} \right)^2 \right]^{\frac{1}{2}} \left[\frac{1}{n_A} \sum_{i \in A} \left((\mathbf{x}_i - \bar{\mathbf{x}}_A)^T (\boldsymbol{\beta}^{(a)} - \hat{\boldsymbol{\beta}}^{(a)}) \right)^2 \right]^{\frac{1}{2}} \end{aligned} \tag{2.68}$$

which converges to 0 in probability because of (2.63) and (2.67).

By Lemma 5 and condition 4, we have

$$\frac{n_A}{n_A - df^{(a)}} = \frac{n_A}{n_A - \hat{s}^{(a)} - 1} \xrightarrow{p} 1. \tag{2.69}$$

Combining (2.63), (2.67), (2.68) and (2.69), we conclude that

$$\hat{\sigma}^2_{e^{(a)}} \xrightarrow{p} \lim_{n\to\infty} \sigma^2_{e^{(a)}}.$$

The remaining part of the proof is to study the difference between the conservative variance estimate and the true asymptotic variance:

$$\begin{aligned}
&\left(\frac{1}{p_A}\lim_{n\to\infty}\sigma^2_{e^{(a)}} + \frac{1}{1-p_A}\lim_{n\to\infty}\sigma^2_{e^{(b)}}\right) - \left(\frac{1-p_A}{p_A}\lim_{n\to\infty}\sigma^2_{e^{(a)}} + \frac{p_A}{1-p_A}\lim_{n\to\infty}\sigma^2_{e^{(b)}} + 2\lim_{n\to\infty}\sigma_{e^{(a)}e^{(b)}}\right) \\
&= \lim_{n\to\infty}\sigma^2_{e^{(a)}} + \lim_{n\to\infty}\sigma^2_{e^{(b)}} - 2\lim_{n\to\infty}\sigma_{e^{(a)}e^{(b)}} \\
&= \lim_{n\to\infty}\sigma^2_{e^{(a)}-e^{(b)}} \\
&= \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\left(a_i - b_i - \mathbf{x}_i^T(\boldsymbol{\beta}^{(a)} - \boldsymbol{\beta}^{(b)})\right)^2.
\end{aligned} \tag{2.70}$$

□

Proof of Theorem 3

Proof. By Lemma 5, $\max(\hat{s}^{(a)}, \hat{s}^{(b)}) = o_p(\min(n_A, n_B))$. Therefore, $(\hat{\sigma}^2_{e^{(a)}}, \hat{\sigma}^2_{e^{(b)}})$ and $((\hat{\sigma}^*)^2_{e^{(a)}}, (\hat{\sigma}^*)^2_{e^{(b)}})$ have the same limits. The conclusion follows from Theorem 2. □

2.D Concentration inequality for sampling without replacement

Our proofs rely heavily on the following Massart concentration inequality for sampling without replacement.

Lemma 7. *Let $\{z_i, i = 1, ..., n\}$ be a finite population of real numbers. Let $A \subset \{i, \ldots, n\}$ be a subset of deterministic size $|A| = n_A$ that is selected randomly without replacement. Define $p_A = n_A/n$, $\sigma^2 = n^{-1}\sum_{i=1}^n (z_i - \bar{z})^2$. Then, for any $t > 0$,*

$$P\left(\bar{z}_A - \bar{z} \geq t\right) \leq \exp\left\{-\frac{p_A n_A t^2}{(1+\tau)^2\sigma^2}\right\}, \tag{2.71}$$

with $\tau = \min\left\{1/70, (3p_A)^2/70, (3 - 3p_A)^2/70\right\}$.

Remark 10. Massart showed in his paper [60] that for sampling without replacement, the following concentration inequality holds:

$$P\left(\bar{z}_A - \bar{z} \geq t\right) \leq \exp\left\{-\frac{p_A n_A t^2}{\sigma^2}\right\}.$$

His proof required that n/n_A must be an integer. We extend the proof to allow n/n_A to be a non-integer but with a slightly larger constant factor $(1+\tau)^2$.

Proof. Assume $\bar{z} = 0$ without loss of generality. For $n_A \leq n/2$, let $m \geq 2$ and $r \geq 0$ be integers satisfying $n - n_A m = r < n_A$. Let $u \geq 0$, we first prove that

$$E\exp\left(u\sum_{i\in A} z_i\right) \leq E\exp\left(u\delta\sum_{i\in B} z_i/\{m(m+1)\} + u^2 n\sigma^2/4\right) \tag{2.72}$$

for a random subset $B \subset \{1, \ldots, n\}$ of fixed size $|B| \leq n/2$ and a certain fixed $\delta \in \{-1, 1\}$. Let P_1 be the probability under which $\{i_1, \ldots, i_n\}$ is a random permutation of $\{1, \ldots, n\}$. Given $\{i_1, \ldots, i_n\}$, we divide the sequence into consecutive blocks $B_1, \ldots, B_{n_A}$ with $|B_j| = m+1$ for $j = 1, \ldots, r$ and $|B_j| = m$ for $j = r+1, ..., n_A$. Let $\bar{z}_k$ be the mean of $\{z_i : i \in B_k\}$ and P_2 be a probability conditionally on $\{i_1, \ldots, i_n\}$ under which w_k is a random element of $\{z_i : i \in B_k\}$, $k = 1, \ldots, n_A$. Then $\{w_1, \ldots, w_{n_A}\}$ is a random sample from $\{z_1, \ldots, z_n\}$ without replacement under $P = P_1 P_2$. Let $\Delta_k = \max_{i\in B_k} z_i - \min_{i\in B_k} z_i$ and denote E_2 the expectation under P_2. The Hoeffding inequality gives

$$E_2\exp\left(u\sum_{k=1}^{n_A} w_k\right) \leq \exp\left(u\sum_{k=1}^{n_A}\bar{z}_k + (u^2/8)\sum_{k=1}^{n_A}\Delta_k^2\right). \tag{2.73}$$

As $\Delta_i^2 \le 2\sum_{i\in B_k}(z_i-\bar{z}_k)^2 \le 2\sum_{i\in B_k} z_i^2$,

$$E_2 \exp\left(u\sum_{k=1}^{n_A} w_k\right) \le \exp\left(u\sum_{k=1}^{n_A}\bar{z}_k + u^2 n\sigma^2/4\right) \tag{2.74}$$

Let $B = \cup_{k=1}^{r} B_k$. As $\bar{z}=0$,

$$\sum_{k=1}^{n_A}\bar{z}_k = \sum_{i\in B} z_i/\{m(m+1)\}. \tag{2.75}$$

This yields (2.72) with $\delta = 1$ when $|B| \le n/2$. Otherwise, (2.72) holds with $\delta=-1$ when B is replaced by B^c, as $\sum_{i\in B} z_i = -\sum_{i\in B^c} z_i$ due to $\bar{z}=0$.

Now, as $m(m+1)\ge 6$, repeated application of (2.72) yields

$$\begin{aligned} E\exp\left(u\sum_{i\in A} z_i\right) &\le E\exp\left[u\delta'\sum_{i\in B'} z_i/\{m(m+1)m'(m'+1)\} + \left(1+\{m(m+1)\}^{-2}\right)u^2 n\sigma^2/4\right] \\ &\le \exp\left[\left(1+\{m(m+1)\}^{-2}(1+1/36+1/36^2+\cdots)\right)u^2 n\sigma^2/4\right] \\ &= \exp\left[\left(1+(36/35)\{m(m+1)\}^{-2}\right)u^2 n\sigma^2/4\right] \\ &\le \exp\left[(1+\tau)^2 u^2 n\sigma^2/4\right] \end{aligned} \tag{2.76}$$

with $\tau = (18/35)\{m(m+1)\}^{-2}$. The upper bound for τ follows from $2\le m < n/n_A < m+1$.

As $\bar{z}=0$, we also have

$$E\exp\left(u\sum_{i\in A} z_i\right) \le \exp\left[(1+\tau)^2 u^2 n\sigma^2/4\right] \tag{2.77}$$

for $n_A > n/2$. This yields (2.71) via the usual

$$\begin{aligned} P\{\bar{z}_A - \bar{z} > t\} &\le \exp\left[-ut + (1+\tau)^2 u^2 n\sigma^2/(4n_A^2)\right] \\ &= \exp\left[-2\frac{p_A n_A t^2}{(1+\tau)^2\sigma^2} + \frac{p_A n_A t^2}{(1+\tau)^2\sigma^2}\right] \end{aligned} \tag{2.78}$$

with $u = 2p_A n_A t/\{\sigma(1+\tau)\}^2$. □

2.E Simulations

In this section we carry out simulation studies to evaluate the finite sample performance of $\widehat{ATE}_{\text{Lasso}}$ estimator. We also present results for the $\widehat{ATE}_{\text{OLS}}$ estimator when $p < n$ and the two-step estimator $\widehat{ATE}_{\text{Lasso+OLS}}$ which adopts Lasso to select covariates and then uses OLS to refit the regression coefficients, see Efron et al. [30], Meinshausen [64], Belloni and

Chernozhukov [6] and Liu and Yu [56] for statistical properties of the Lasso+OLS estimator in the linear regression model.

Let $\hat{\boldsymbol{\beta}}^{(a)}$ be the Lasso estimator defined in eq. (2.1) (we omit the subscript "Lasso" for the sake of simplicity) and let $\hat{S}^{(a)} = \{j : \hat{\boldsymbol{\beta}}_j^{(a)} \neq 0\}$ be the support of $\hat{\boldsymbol{\beta}}^{(a)}$. The Lasso+OLS adjustment vector $\hat{\beta}^{(a)}_{\text{Lasso+OLS}}$ for treatment group A is defined by

$$\hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso+OLS}} = \underset{\boldsymbol{\beta}:\ \beta_j=0,\ \forall j \notin \hat{S}^{(a)}}{\arg\min} \frac{1}{2n_A} \sum_{i \in A} \left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \boldsymbol{\beta}\right)^2.$$

We can define the Lasso+OLS adjustment vector $\hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso+OLS}}$ for control group B similarly. Then $\widehat{ATE}_{\text{Lasso+OLS}}$ is given by

$$\widehat{ATE}_{\text{Lasso+OLS}} = \left[\bar{a}_A - (\bar{\mathbf{x}}_A - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(a)}_{\text{Lasso+OLS}}\right] - \left[\bar{b}_B - (\bar{\mathbf{x}}_B - \bar{\mathbf{x}})^T \hat{\boldsymbol{\beta}}^{(b)}_{\text{Lasso+OLS}}\right].$$

We use the R package "glmnet" to compute the Lasso solution path. We select the tuning parameters λ_a and λ_b by 10-fold Cross Validation (CV) and denote the corresponding adjusted estimators as cv(Lasso) and cv(Lasso+OLS) respectively. We should mention that for the cv(Lasso+OLS) adjusted estimator, we compute the CV error for a given value of the λ_a (or λ_b) based on the whole Lasso+OLS estimator instead of the Lasso estimator, see algorithm 1 for details. Therefore, the cv(Lasso+OLS) adjusted estimator and the cv(Lasso) adjusted estimator may select different covariates to do the adjustment. This type of cross validation for cv(Lasso+OLS) requires more computation effort than the cross validation based on just the Lasso estimator since it needs to compute the OLS estimator for each fold and for each λ_a (or λ_b), but it can give better prediction and covariates selection performance.

The potential outcomes a_i and b_i are generated from the following nonlinear model: for $i = 1, ..., n$,

$$a_i = \sum_{j=1}^{s} x_{ij}\beta_j^{(a1)} + \exp\left(\sum_{j=1}^{s} x_{ij}\beta_j^{(a2)}\right) + \epsilon_i^{(a)},$$

$$b_i = \sum_{j=1}^{s} x_{ij}\beta_j^{(b1)} + \exp\left(\sum_{j=1}^{s} x_{ij}\beta_j^{(b2)}\right) + \epsilon_i^{(b)},$$

where $\epsilon_i^{(a)}$ and $\epsilon_i^{(b)}$ are independent error terms. We set $n = 250$, $s = 10$, $p = 50$ and 500. For $p = 50$, we can compute OLS estimator and compare it with the Lasso. The covariates vector $\mathbf{x}_i$ is generated from a multivariate normal distribution $\mathcal{N}(0, \Sigma)$. We consider two different Toeplitz covariance matrices Σ which control the correlation among the covariates:

$$\Sigma_{ii} = 1;\ \Sigma_{ij} = \rho^{|i-j|}\ \forall i \neq j,$$

where $\rho = 0, 0.6$. The true coefficients $\beta_j^{(a1)}$, $\beta_j^{(a2)}$, $\beta_j^{(b1)}$, $\beta_j^{(b2)}$ are generated independently according to

$$\beta_j^{(a1)} \sim t_3; \quad \beta_j^{(a2)} \sim 0.1 * t_3, \quad j = 1, ..., s,$$

Algorithm 1: K-fold Cross Validation (CV) for the Lasso+OLS estimator

Input: Design matrix X, response Y and a sequence of tuning parameter $\lambda_1, ..., \lambda_J$; Number of folds K.

Output: The optimal tuning parameter selected by CV: $\lambda_{optimal}$.

Randomly divide the data $z = (X, Y)$ into K roughly equal parts $z_k, k = 1, ..., K$

for *each* $k = 1, ..., K$ **do**

  Denote $\hat{S}^{(k)}(\lambda_0) = \emptyset$ and $\hat{\beta}^{(k)}_{\text{Lasso+OLS}}(\lambda_0) = 0$

  Fit the model with parameters $\lambda_j, j = 1, ..., J$ to the other $K-1$ parts $z_{-k} = z \setminus z_k$ of the data, giving the Lasso solution path $\hat{\beta}^{(k)}(\lambda_j), j = 1, ..., J$ and compute the selected covariates set $\hat{S}^{(k)}(\lambda_j) = \{l : \hat{\beta}^{(k)}_l(\lambda_j) \neq 0\}, j = 1, ..., J$ on the path

  for *each* $j = 1, ..., J$ **do**

    compute the Lasso+OLS estimator:

$$\hat{\beta}^{(k)}_{\text{Lasso+OLS}}(\lambda_j) = \begin{cases} \underset{\boldsymbol{\beta}:\ \beta_j=0,\ \forall j \notin \hat{S}^{(k)}(\lambda_j)}{\arg\min} \left\{ \frac{1}{2|z_{-k}|} \sum_{i \in z_{-k}} (y_i - x_i^T \beta)^2 \right\}, & \text{if} \quad \hat{S}^{(k)}(\lambda_j) \neq \hat{S}^{(k)}(\lambda_{j-1}), \\ \hat{\beta}^{(k)}_{\text{Lasso+OLS}}(\lambda_{j-1}), & \text{otherwise} \end{cases} \tag{2.79}$$

  Compute the error in predicting the kth part of the data $PE^{(k)}$:

$$PE^{(k)}(\lambda_j) = \frac{1}{|z_k|} \sum_{i \in z_k} \left(y_i - x_i^T \hat{\beta}^{(k)}_{\text{Lasso+OLS}}(\lambda_j) \right)^2$$

Compute cross validation error $CV(\lambda_j)$, $j = 1, ..., J$:

$$CV(\lambda_j) = \frac{1}{K} \sum_{k=1}^{K} PE^{(k)}(\lambda_j)$$

Compute the optimal λ selected by CV

$$\lambda_{optimal} = \underset{\lambda_j:\ j=1,...,J}{\arg\min}\ CV(\lambda_j);$$

return $\lambda_{optimal}$.

$$\beta_j^{(b1)} \sim \beta_j^{(a1)} + t_3; \quad \beta_j^{(b2)} \sim \beta_j^{(a2)} + 0.1 * t_3, \quad j = 1, ..., s,$$

where t_3 denotes the t distribution with three degrees of freedom. This ensures that the treatment effects are not constant across individuals, and that the linear model does not hold in this simulation. The error terms $\epsilon_i^{(a)}$ and $\epsilon_i^{(b)}$ are generated according to the following linear model with hidden covariates $\mathbf{z}_i$:

$$\epsilon_i^{(a)} = \sum_{j=1}^{s} z_{ij}\beta_j^{(a1)} + \tilde{\epsilon}_i^{(a)},$$

$$\epsilon_i^{(b)} = \sum_{j=1}^{s} z_{ij}\beta_j^{(b1)} + \tilde{\epsilon}_i^{(b)},$$

where $\tilde{\epsilon}_i^{(a)}$ and $\tilde{\epsilon}_i^{(b)}$ are drawn independently from a standard normal distribution. The vector $\mathbf{z}_i$ is independent of $\mathbf{x}_i$ and is also drawn independently from the multivariate normal distribution $\mathcal{N}(0, \Sigma)$. The values of $\mathbf{x}_i$, $\beta^{(a1)}$, $\beta^{(a2)}$, $\beta^{(b1)}$, $\beta^{(b2)}$, $\mathbf{z}_i$, $\tilde{\epsilon}_i^{(a)}$, $\tilde{\epsilon}_i^{(b)}$, a_i and b_i are generated once and then kept fixed.

After the potential outcomes are generated, a completely randomized experiment is simulated 25000 times, assigning $n_A = 100, 125, 150$ subjects to treatment A and the remainder to control B. There are 12 different combinations of (p, ρ, n_A) in total.

Figures 2.E.1–2.E.3 show boxplots of different ATE estimators with their standard deviations (computed from 25000 replicates of randomized experiments) presented on top of each box. Regardless of whether the design is balanced ($n_A = 125$) or not ($n_A = 100, 150$), the regression based estimators have much smaller variances and than that of the unadjusted estimator and therefore improve the estimation precision.

To further compare the performance of these estimators, we present the bias, the standard deviation (SD) and the root-mean square error ($\sqrt{\text{MSE}}$) of the estimates in table 2.E.1. Bias is reported as the absolute difference from the true treatment effect. We find that the bias of each method is substantially smaller (more than 10 times smaller) than the SD. The cv(Lasso) and cv(Lasso+OLS) adjusted estimators perform similar in terms of SD and $\sqrt{\text{MSE}}$: reducing those of the OLS adjusted estimator and the unadjusted estimator by $10\% - 15\%$ and $15\% - 31\%$ respectively. We also compare the number of selected covariates by cv(Lasso) and cv(Lasso+OLS) for treatment group and control group separately, see table 2.E.2. It is easy to see that the cv(Lasso+OLS) adjusted estimator uses many fewer (more than 44%) covariates in the adjustment to obtain similar improvement of SD and $\sqrt{\text{MSE}}$ of ATE estimate as the cv(Lasso) adjusted estimator. Moreover, we find that the covariates selected by the cv(Lasso+OLS) are more stable across different realizations of treatment assignment than the covariates selected by the cv(Lasso). Overall, the cv(Lasso+OLS) adjusted, the cv(Lasso) adjusted, the OLS adjusted and the unadjusted estimators perform from best to worst.

We move now to study the finite sample performance of Neyman-type conservative variance estimates. For each simulation example and each one of the 25000 completely randomized experiments, we calculate the ATE estimates ($\widehat{ATE}$) and the Neyman variance estimates

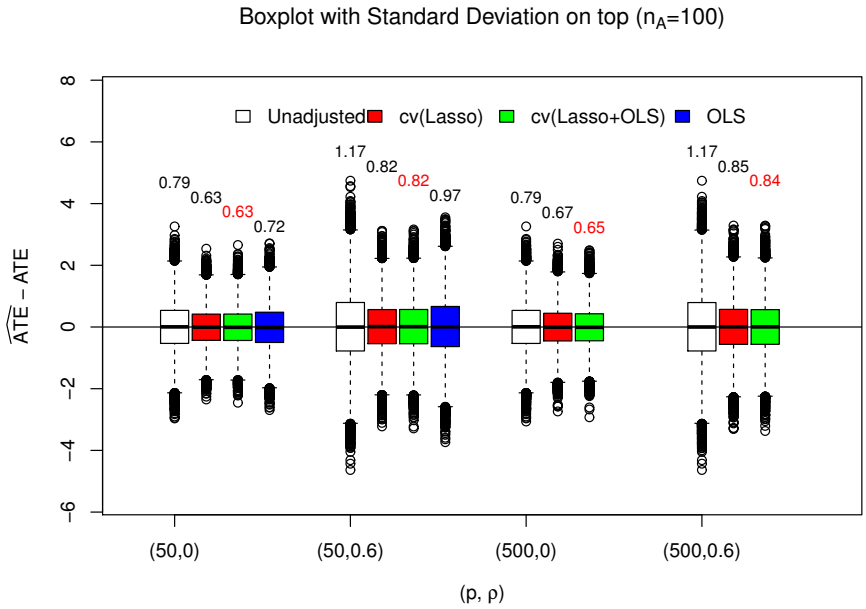

FIGURE 2.E.1: **Boxplot of errors of ATE estimates for $n_A = 100$.** Unadjusted, OLS adjusted, cv(Lasso) and cv(Lasso+OLS) adjusted estimators are shown with their standard deviations presented on top of each box. The OLS adjusted estimator is only computed when $p = 50$.

($\hat{\sigma}$) and then form the 95% confidence intervals $[\widehat{ATE} - 1.96 \cdot \hat{\sigma}/\sqrt{n}, \widehat{ATE} + 1.96 \cdot \hat{\sigma}/\sqrt{n}]$. Figures 2.E.4–2.E.6 present the boxplots of the interval length with the coverage probability noted on top of each box for the unadjusted, OLS adjusted (only computed when $p = 50$), cv(Lasso) adjusted and cv(Lasso+OLS) adjusted estimators. We find that all the confidence intervals for the unadjusted estimator are conservative. The cv(Lasso) adjusted and the cv(Lasso+OLS0 adjusted estimators perform very similar: although their coverage probability (at least 92%) may be slightly less than the pre-assigned confidence level (95%), their mean interval length is much shorter ($26\% - 37\%$) than that of the unadjusted estimator. The OLS adjusted estimator has comparable interval length with the cv(Lasso) and cv(Lasso+OLS) adjusted estimator, but has slightly worse coverage probability ($90\% - 93\%$).

We conduct more simulation examples to evaluate the conditions assumed for asymptotic normality of the Lasso adjusted estimator. We use the same simulation setup as above, but for simplicity, we generate the potential outcomes from a linear model (set $\beta^{(a2)} = \beta^{(b2)} = 0$) and remove the effects of the hidden covariates z_i in generating the error terms $\epsilon_i^{(a)}$ and $\epsilon_i^{(b)}$ and set $\rho = 0$, $n_A = 125$. We find that the distribution of the cv(Lasso) adjusted estimator may be non-normal when:

(1). The covariates are generated from Gaussian distribution and the error terms do not

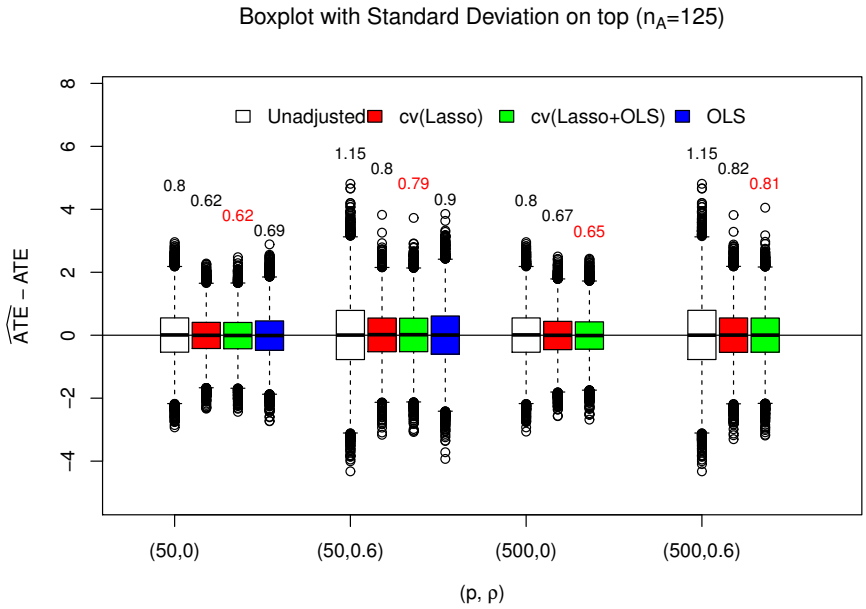

FIGURE 2.E.2: **Boxplot of errors of ATE estimates for $n_A = 125$.** Unadjusted, OLS adjusted, cv(Lasso) and cv(Lasso+OLS) adjusted estimators are shown with their standard deviations presented on top of each box. The OLS adjusted estimator is only computed when $p = 50$.

satisfy second moment condition, e.g., being generated from t distribution with one degree of freedom, see the upper two subplots of fig. 2.1 for the histograms of unadjusted the cv(Lasso) adjusted estimators (the corresponding p-values of Kolmogorov–Smirnov testing for normality are less than $2.2e-16$).

(2). The covariates do not have bounded fourth moments, e.g., being generated from t distribution with three degrees of freedom, see the lower two subplots of fig. 2.1 for the histograms of unadjusted the cv(Lasso) adjusted estimators (again, the corresponding p-values of Kolmogorov–Smirnov testing for normality are less than $2.2e-16$).

These findings indicate that our moment condition (condition 2 and remark 1) cannot be dramatically weakened. However, we also find that the cv(Lasso) adjusted estimator still has smaller SD and $\sqrt{\text{MSE}}$ than the unadjusted estimator even when these moment conditions do not hold.

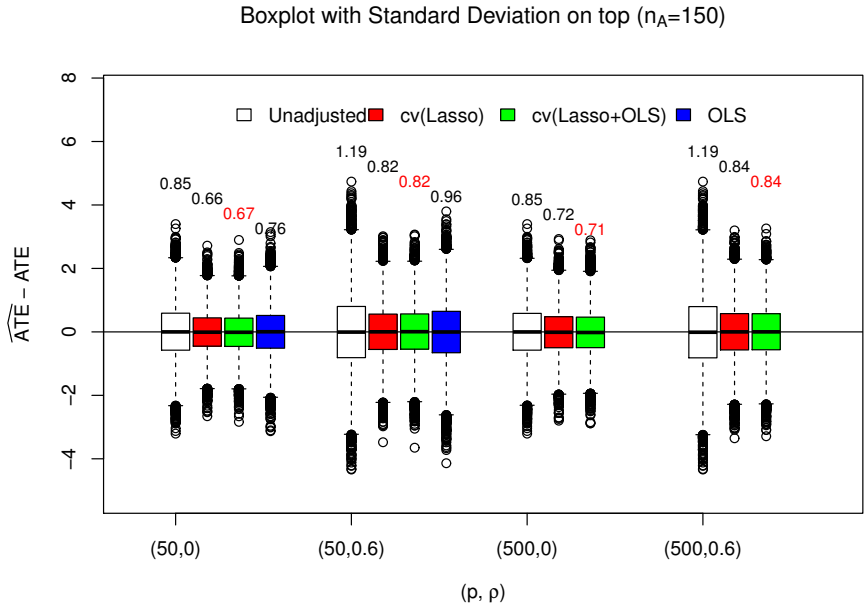

FIGURE 2.E.3: **Boxplot of errors of ATE estimates for $n_A = 150$.** Unadjusted, OLS adjusted, cv(Lasso) and cv(Lasso+OLS) adjusted estimators are shown with their standard deviations presented on top of each box. The OLS adjusted estimator is only computed when $p = 50$.

TABLE 2.E.1: **Bias, standard deviation (SD) and root-mean square error $\sqrt{\text{MSE}}$ of ATE estimates.**

		(p, ρ)			
Statistic	**Method**	(50,0)	(50,0.6)	(500,0)	(500,0.6)
		$n_A = 100$			
bias	Unadjusted	**0.003(0.004)***	**0.005(0.005)**	**0.002(0.003)**	**0.003(0.005)**
	OLS	0.014(0.005)	0.013(0.006)	-	-
	cv(Lasso)	0.007(0.004)	0.014(0.005)	0.006(0.004)	0.005(0.004)
	cv(Lasso+OLS)	0.011(0.004)	0.013(0.005)	0.009(0.004)	**0.003(0.004)**
SD	Unadjusted	0.79(0.08)	1.17(0.11)	0.79(0.07)	1.17(0.11)
	OLS	0.72(0.07)	0.96(0.09)	-	-
	cv(Lasso)	**0.62(0.06)**	**0.82(0.08)**	0.67(0.06)	**0.84(0.08)**
	cv(Lasso+OLS)	0.63(0.06)	**0.82(0.08)**	**0.65(0.06)**	**0.84(0.08)**
$\sqrt{\text{MSE}}$	Unadjusted	0.79(0.08)	1.17(0.11)	0.79(0.07)	1.17(0.11)
	OLS	0.72(0.07)	0.97(0.09)	-	-
	cv(Lasso)	**0.63(0.06)**	**0.82(0.08)**	0.67(0.06)	0.85(0.08)
	cv(Lasso+OLS)	**0.63(0.06)**	**0.82(0.08)**	**0.65(0.06)**	**0.84(0.08)**
		$n_A = 125$			
bias	Unadjusted	0.008(0.005)	0.011(0.007)	**0.006(0.004)**	0.01(0.007)
	OLS	0.008(0.004)	**0.005(0.005)**	-	-
	cv(Lasso)	**0.005(0.003)**	0.012(0.005)	0.007(0.004)	0.004(0.004)
	cv(Lasso+OLS)	0.012(0.004)	0.012(0.005)	0.011(0.004)	**0.003(0.003)**
SD	Unadjusted	0.80(0.08)	1.15(0.11)	0.8(0.08)	1.15(0.11)
	OLS	0.69(0.06)	0.90(0.09)	-	-
	cv(Lasso)	**0.62(0.06)**	**0.79(0.07)**	0.67(0.06)	0.82(0.08)
	cv(Lasso+OLS)	**0.62(0.06)**	**0.79(0.07)**	**0.65(0.06)**	**0.81(0.08)**
$\sqrt{\text{MSE}}$	Unadjusted	0.80(0.07)	1.15(0.11)	0.8(0.07)	1.15(0.11)
	OLS	0.69(0.07)	0.90(0.09)	-	-
	cv(Lasso)	**0.62(0.06)**	0.80(0.08)	0.67(0.06)	0.82(0.08)
	cv(Lasso+OLS)	**0.62(0.06)**	**0.79(0.07)**	**0.65(0.06)**	**0.81(0.08)**
		$n_A = 150$			
bias	Unadjusted	0.004(0.004)	**0.000(0.005)**	**0.002(0.003)**	0.005(0.005)
	OLS	**0.002(0.003)**	0.006(0.005)	-	-
	cv(Lasso)	0.003(0.003)	0.002(0.004)	0.01(0.005)	0.002(0.003)
	cv(Lasso+OLS)	0.011(0.004)	0.006(0.004)	0.017(0.005)	**0.001(0.003)**
SD	Unadjusted	0.85(0.08)	1.19(0.11)	0.85(0.08)	1.19(0.11)
	OLS	0.76(0.07)	0.96(0.09)	-	-
	cv(Lasso)	**0.66(0.06)**	0.82(0.08)	0.72(0.07)	**0.84(0.08)**
	cv(Lasso+OLS)	0.67(0.06)	**0.81(0.07)**	**0.71(0.07)**	**0.84(0.08)**
$\sqrt{\text{MSE}}$	Unadjusted	0.85(0.08)	1.19(0.11)	0.85(0.08)	1.19(0.11)
	OLS	0.76(0.07)	0.96(0.09)	-	-
	cv(Lasso)	**0.66(0.06)**	**0.82(0.08)**	0.72(0.07)	**0.84(0.08)**
	cv(Lasso+OLS)	0.67(0.06)	**0.82(0.08)**	**0.71(0.07)**	**0.84(0.08)**

* The numbers in parentheses are the corresponding standard errors estimated by using the bootstrap with $B = 500$ resamplings of the ATE estimates.

TABLE 2.E.2: **Mean number of selected covariates for treated and control group.**

		(p, ρ)			
Group	**Method**	(50,0)	(50,0.6)	(500,0)	(500,0.6)
		$n_A = 100$			
treated	cv(Lasso)	16	13	22	22
	cv(Lasso+OLS)	6	6	7	7
control	cv(Lasso)	20	11	32	28
	cv(Lasso+OLS)	8	6	7	7
		$n_A = 125$			
treated	cv(Lasso)	17	13	25	24
	cv(Lasso+OLS)	7	6	6	6
control	cv(Lasso)	19	11	32	27
	cv(Lasso+OLS)	8	6	9	8
		$n_A = 150$			
treated	cv(Lasso)	18	13	29	26
	cv(Lasso+OLS)	8	7	6	6
control	cv(Lasso)	19	12	30	25
	cv(Lasso+OLS)	8	6	11	8

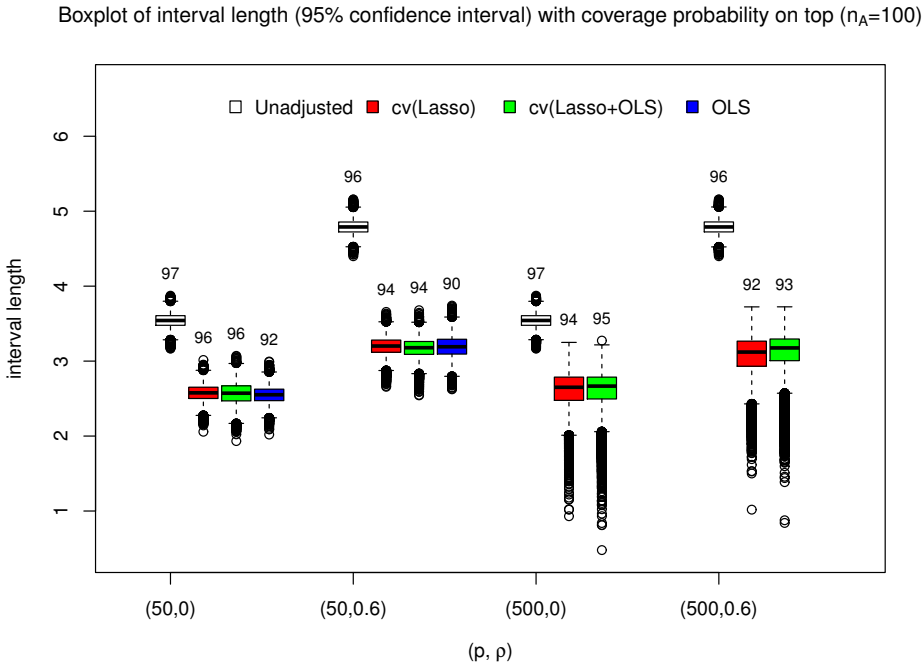

FIGURE 2.E.4: **Boxplot of the confidence interval length for $n_A = 100$.** Intervals for unadjusted, OLS adjusted, cv(Lasso) adjusted and cv(Lasso+OLS) adjusted estimators are shown. Coverage probability (%) is shown on top of each box. The OLS adjusted estimator is only computed when $p = 50$.

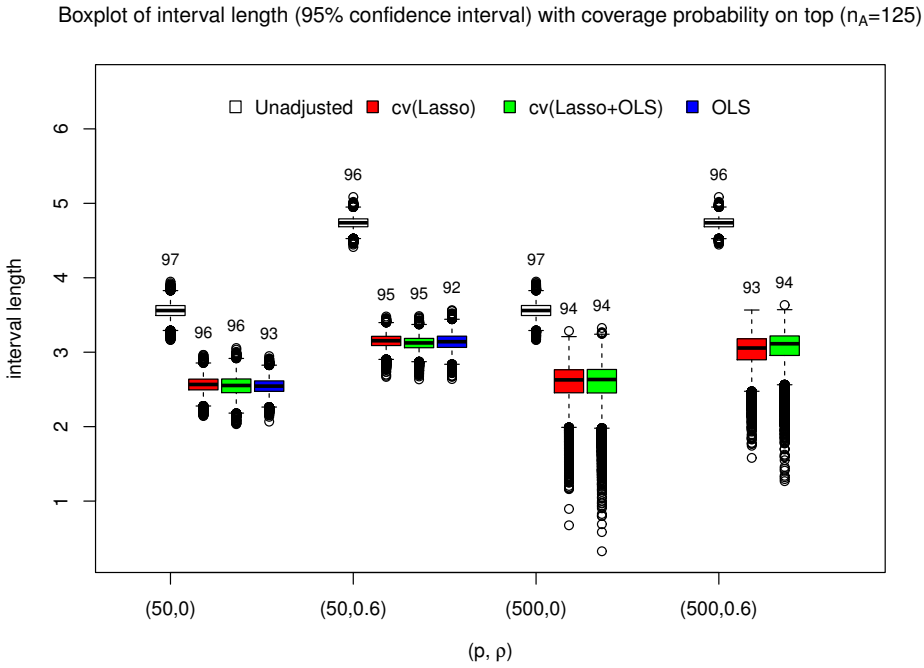

FIGURE 2.E.5: **Boxplot of the confidence interval length for $n_A = 125$.** Intervals for unadjusted, OLS adjusted, cv(Lasso) adjusted and cv(Lasso+OLS) adjusted estimators are shown. Coverage probability (%) is shown on top of each box. The OLS adjusted estimator is only computed when $p = 50$.

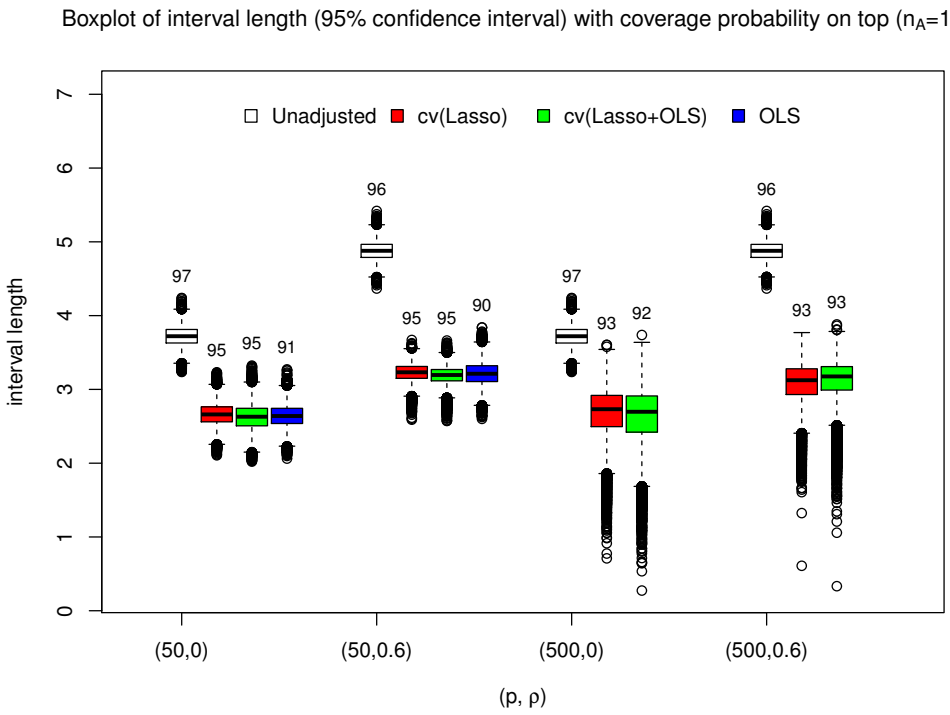

FIGURE 2.E.6: **Boxplot of the confidence interval length for $n_A = 150$.** Intervals for unadjusted, OLS adjusted, cv(Lasso) adjusted and cv(Lasso+OLS) adjusted estimators are shown. Coverage probability (%) is shown on top of each box. The OLS adjusted estimator is only computed when $p = 50$.

2.F The design matrix of the PAC data

In the PAC data, there are 59 covariates (main effects) including 50 indicators which may be correlated with the outcomes. One of the main effects (called interactnew) has heavy tail, so we perform the transform: $x \to \log(|x|+1)$ to make it more normally distributed. We then centralize and standardize the non-indicator covariates. The quadratic terms (9 in total) of non-indicator covariates and two-way interactions between main effects (1711 in total) may also help predict the potential outcomes, so we included them in the design matrix. The quadratic terms and the interactions between non-indicator covariates and the interactions between indicator covariates and non-indicator covariates are also centered and standardized. Some of the interactions are identical to other effects and we only retain one of them. We also remove the interactions which are highly correlated (with correlation larger than 0.95) with the main effects and remove the indicators with very sparse entries (where the number of 1's is less than 20). Finally, we form a design matrix X with 1172 columns (covariates) and 1013 rows (subjects).

2.G Estimation of constants in the conditions

Let $S^{(a)} = \{j : \boldsymbol{\beta}_j^{(a)} \neq 0\}$ and $S^{(b)} = \{j : \boldsymbol{\beta}_j^{(b)} \neq 0\}$ denote the sets of relevant covariates for treatment group and control group respectively. Denote $S = S^{(a)} \bigcup S^{(b)} = \{j : \boldsymbol{\beta}_j^{(a)} \neq 0 \text{ or } \boldsymbol{\beta}_j^{(b)} \neq 0\}$. We use bootstrap to get an estimation of the relevant covariates sets $S^{(a)}, S^{(b)}$ and then the approximation errors $e^{(a)}$ and $e^{(b)}$ are estimated by regressing the observed potential outcomes a and b on the covariates in S respectively. We only present how to estimate $S^{(a)}$ and $e^{(a)}$ in detail and the estimation of $S^{(b)}$ and $e^{(b)}$ are similar.

Let A, B be the set of treated subjects (using PAC) and control subjects (without using PAC) respectively. Denote $a_i, i \in A$ the potential outcomes (quality-adjusted life years (QALYs)) under treatment and $x_i \in \mathbb{R}^{1172}$ the covariates vector of the ith subject. For each $d = 1, ..., 1000$, we draw a bootstrap sample $\{(a_i^*(d), x_i^*(d)) : i \in A\}$ with replacement from the data points $\{(a_i, x_i) : i \in A\}$. We then compute the LassoOLS(CV) adjusted vector $\hat{\boldsymbol{\beta}}(d)$ based on each bootstrap sample $\{(a_i^*(d), x_i^*(d)) : i \in A\}$. Let τ_j be the selection fraction of non-zero $\hat{\beta}_j(d)$ in the 1000 bootstrap estimators, i.e., $\tau_j = (1/1000)\sum_{d=1}^{1000} \mathbb{I}_{\{\hat{\beta}_j(d) \neq 0\}}$, where $\mathbb{I}$ is the indicator function. We form the relevant covariates $S^{(a)}$ by the covariates whose selection fraction are larger than 0.5: $S^{(a)} = \{j : \tau_j > 0.5\}$.

To estimate the approximation error $e^{(a)}$, we regress a_i on the relevant covariates $x_{ij}, j \in S^{(a)}$ and compute OLS estimate and the corresponding residual. That is, let $T^{(a)}$ denote the complement set of $S^{(a)}$,

$$\boldsymbol{\beta}_{\text{OLS}}^{(a)} = \underset{\boldsymbol{\beta}:\ \beta_j = 0,\ \forall j \in T^{(a)}}{\arg\min} \frac{1}{2n_A} \sum_{i \in A} \left(a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \boldsymbol{\beta}\right)^2.$$

$$e_i^{(a)} = a_i - \bar{a}_A - (\mathbf{x}_i - \bar{\mathbf{x}}_A)^T \boldsymbol{\beta}_{\text{OLS}}^{(a)},\ i \in A.$$

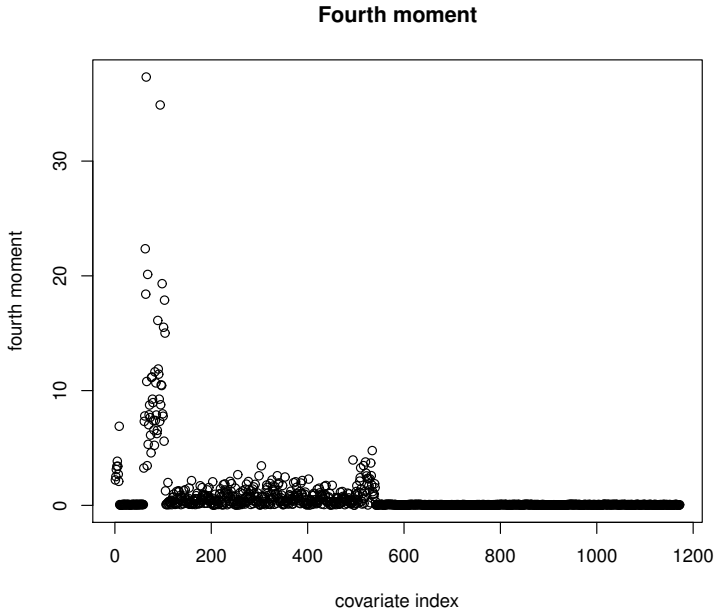

FIGURE 2.G.1: **Fourth moment of each covariate.** The covariates with the largest two fourth moments (37.3 and 34.9 respectively) are quadratic term $interactnew^2$ and interaction term $IMscorerct$: $systemnew$ respectively. Neither is selected by the Lasso to do the adjustment. All the fourth moments of the main effects are less than 7.

The maximal covariance δ_n is estimated as:

$$\max\left\{\max_j\left|\frac{1}{n_A}\sum_{i\in A}\left(x_{ij}-(\bar{\mathbf{x}})_j\right)\left(e_i^{(a)}-\bar{e}_A^{(a)}\right)\right|,\ \max_j\left|\frac{1}{n_B}\sum_{i\in B}\left(x_{ij}-(\bar{\mathbf{x}})_j\right)\left(e_i^{(b)}-\bar{e}_B^{(b)}\right)\right|\right\}.$$

Chapter 3

Modeling area MT with sparse coding principles

3.1 Introduction

The mammalian visual cortex is organized in a hierarchy of regions delineated by physiological and functional boundaries [94]. A major challenge in systems neuroscience is to understand the roles of these regions in performing the various tasks of visual cognition, including object detection and localization, face recognition, object tracking, and depth perception. Complete characterization of nervous system function is essentially impossible due to the staggering complexity of the brain and the limitations of the various technologies for recording neural activity. However, the last 50 years have shown that it is possible to make progress on our understanding of the computations performed in the visual cortex through analyzing data available from current technologies.

One problem which has received considerable attention is the characterization of single-neuron receptive fields, which is made possible by recordings of single-neuron extracellular voltage. The term "receptive field" was coined by Sherrington in the context of studying the spatial extent of the scratch reflex in vertebrates [83]. It was applied to the study of visual neurons by Hartline, who defined it as the region of the visual world, in retinal coordinates, that must be illuminated in order to achieve a neuronal response [39]. Since then, the concept has broadened considerably to delineate sensitivity of neurons to many parameters beyond just spatial localization. In visual neuroscience this may include spatial frequency, orientation, phase, or more complex image features. In this chapter we consider a very general definition: the receptive field of a neuron is a real-valued function of the recent history of the contents of the visual field of the experimental subject. This function describes the strength of a single neuron's response (probability of spike, expected spiking rate, or deviation relative to the resting spike rate). The problem of receptive field estimation is to learn this function from recordings of a neuron's activity in response to visual stimuli.

Of course even successful estimation of an encoding function over an arbitrary input does

not provide complete understanding of the neuron or a brain region, but does capture some feed-forward transmission of stimulus information from the outside world to the neuron's cell body. It does not necessarily offer any biophysical information on how this information is transmitted or transformed, nor does it address the role of recurrent or feedback neural connections, which in fact are as plentiful as feed-forward connections in ventral stream of the visual cortex [33]. It is indeed surprising that neural activity can be explained at all in this way, as, *a priori* it is not obvious that single neurons would marginally encode information about the stimulus [74]. However, the foundational work of Hubel and Wiesel showed that receptive fields of neurons in anesthetized cat and monkey lateral geniculate nucleus (LGN) and V1 could be meaningfully characterized according to a relatively small number of stimulus parameters [47, 46]. This led to biological models explaining how early visual cortex neurons pool the activity of upstream neurons to produce these receptive fields. Since then, there has been a blossoming of experimental work on establishing receptive fields of neurons in the LGN and V1 that now there are established models of receptive fields in these regions [18].

A current frontier in visual neuroscience is in understanding receptive fields of brain regions further downstream in the extrastriate cortex. Here the visual pathway diverges into two dominant streams of information, the so-called dorsal and ventral streams. Through ablation studies and single neuron recordings, it is understood that the dorsal stream performs spatial reasoning, motion perception, object tracking, and visual guidance for movement (e.g., reaching). The ventral stream is generally understood to be involved in shape, color, object and facial recognition [93]. It is still largely a mystery how these complex cognitive tasks are performed, and understanding single-neuron receptive fields in these areas is an essential step.

In this chapter, we focus on visual area MT, or V5, a major area in the dorsal stream which is involved in the perception of motion and is sensitive to speed and direction of moving objects. We use recordings of 51 MT cells from *macacca mulatta* taken while the subject was stimulated by 20,000 - 40,000 frames of short natural video clips. A prior study, for which this data was collected, found that a V1-like Gabor wavelet transform, followed by biologically motivated nonlinearities, provided very high-accuracy fits of these MT cells [69]. These findings were consistent with the theory that MT receptive fields pool the output of V1 inputs corresponding to the same velocity of motion [13]. In this chapter we build models for these neurons from a different neuroscientific motivation. We adapt a pipeline developed in Mairal et al. [59] for modeling responses of neurons in V4 to natural images. Their model was inspired by the principle of sparse coding, which posits that visual representation is optimized such that neuron spiking is a sparse phenomenon. It insert an extra layer between outputs of V1-type cells and the activity of V4 cells which recodes the activity of pooled V1 cells in a sparse manner. Besides being biologically plausible, this approach is similar to a popular architecture for computer vision systems that perform automated object recognition [100]. Because motion information is so important in area MT, we adapt the pipeline to represent spatiotemporal features. Averaged over the set of 51 MT cells, our resulting model is more accurate than the model in Nishimoto and Gallant [69] in

predicting neuron spike rates on a holdout set of natural images. This does not contradict the conclusions of Nishimoto and Gallant [69], but suggests that the sparse coding layer generates feature representations that can better match the tuning of some MT neurons.

To better understand the difference between our model and the spatiotemporal Gabor wavelet model, we compare the representations of natural video provided by the two models. We perform canonical correlation analysis, which measures the amount of overlapping information in the two representations in linear a sense. We find that, despite the nonlinearity of the sparse coding map, that there are many canonical variables that are highly correlated between the two representations. In other words, the two representations are identical in certain directions. Hence, many MT cells that lie approximately in the span of these top canonical variables are equally well modeled by the two approaches. However, the sparse coding operation adds flexibility to the representation, and encodes some more complex features that are a significantly better match for a subset of MT neurons. Our findings indicates that MT is a mixed region, in that some cells have tuning to wavelet-like features similar to V1 cells, whereas some are more sensitive to features of greater complexity.

3.2 Receptive field estimation in extrastriate visual cortex

Nonlinear receptive fields

Receptive field estimation is challenging because of the essential nonlinearity of neural systems. Visual receptive fields are determined by the coordinated activity of many neurons connected in a circuit that originates in the rod and cone cells of the retina. Photoreceptor cells encode their signal through varying levels of hyperpolarization of membrane potential, but very quickly, the signal is digitized into discrete action potentials, or spikes. Hence, neural activity is fundamentally a nonlinear phenomenon, as it is encoded in a discrete-valued signal. It is thus perhaps surprising that linear methods such as spike-triggered average (more generally, linear regression), or simple nonlinear extensions thereof are able to produce fairly accurate receptive field models all the way downstream to area V1. Here, 'linear' signifies a function that is a linear transformation of the pixel intensities of the image viewed by an experimental subject. In V1, these models are accurate in the sense that they can accurately predict neuron spiking rate in response to a wide variety of images, both artificially designed (eg. Fourier basis functions or wavelets) and naturally occurring (eg. scenes, faces, movie clips). In particular, simple cells in V1 can be modeled as performing a linear filtering to the input image, followed by a few simple nonlinear operations such as response normalization and a static thresholding nonlinearity [18]. Such models capture about 40% of the variance in V1 cells that can be explained through feed-forward propagation of signal from the retina.

However, linear methods do not generalize well to more complex brain regions downstream from V1 [91]. This presents a fundamental challenge for receptive field estimation. By analyzing neural responses to repeated presentations of visual stimuli, it is clear that a

large proportion of the variance in spike rates of single neurons in these areas is explainable by feed-forward propagation from the stimulus input. However, nonlinear modeling, in full generality, is essentially intractable in high dimensions. Minimax theory shows that, if one wants to entertain all possible nonlinear functions in a certain smoothness class, to reach a fixed mean-square error in estimation of the function, sample size must scale exponentially with the dimension. Training data is limited by the difficulty of recording from single neurons; for example, data from awake macaques can only be gathered while the monkey is successfully fixating, which may only last for several seconds at a time. In the MT datasets we analyze in this chapter, each neuron was successfully probed for 20,000-40000 frames of video; while these are extraordinarily large training sets by the standards of electrophysiological experiments, they are still too small for unconstrained nonlinear function estimation in the dimensions of the input. The stimulus images are 128 x 128 pixels, and, because the neuron integrates information over recent history of the stimulus, approximately 10 prior frames of video must be included to model the spike count occuring in the duration of a single frame.

Hence, the space of possible models of receptive fields must be constrained to allow for tractable estimation. Fortunately, decades of research into the connectomics of the visual cortex, the biophysics of single neurons and synaptic transmission, and basic properties of spiking neurons provide a wealth of knowledge about what kinds of receptive field models are plausible, biologically. Our general strategy is to fit models that are a combination of linear and nonlinear components. Denote the stimulus video as a function of two spatial coordinates and a time coordinate, i.e. $\mathbf{I}(x, y, t)$ where $x, y \in \{0, \ldots, W\}, t \in \{0, \ldots, T-1\}$ where W is the width of the input video (assumed here to be square), and T is the number of frames in the stimulus set. We denote the nonlinear feature extraction component of our model as a vector-valued function $\mathbf{f} : \mathbb{R}^{w \times w \times l} \to \mathbb{R}^p$ where l delineates the amount of temporal information incorporated into the features, and p is the dimension of the feature set. For notational ease, we define the feature vector at time t in terms of the last l frames of video prior to t

$$\mathbf{x}_t \stackrel{\mathsf{def}}{=} \mathbf{f}\left(\mathbf{I}(\cdot, \cdot, t), \mathbf{I}(\cdot, \cdot, t-1), \ldots, \mathbf{I}(\cdot, \cdot, t-l)\right)$$

The linear component then predicts the actual neuron's response at time t as a linear function of the features of the last h frames of stimulus video shown up to and including time t. h is a tuning parameter that reflects duration of the 'memory' of a cell:

$$\hat{y}(t) = \sum_{j=0}^{h-1} \hat{\boldsymbol{\beta}}_{j+1}^T \mathbf{x}_{t-j} \tag{3.1}$$

In the approach taken in this chapter, the nonlinear transformation $\mathbf{f}$ is constructed without consideration of neural data, being guided instead by prior knowledge of the visual cortex, neurobiological motivations, and unsupervised learning methods. The recordings of neural data are only used in determining the vectors $\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_h$. This approach is also commonly known as 'linearizing the response,' since a well-designed $\mathbf{f}$ will transform the stimulus video

in such a way that the linear modeling step will produce accurate models of neuron spike rates [26, 98]. More precisely, we desire that the neuron's response is close in ℓ_2 distance to a linear combination of the components $\mathbf{f}$. This is essential for tractable estimation of the receptive fields, since estimating linear models requires far fewer samples than nonlinear estimation. In this chapter we use feature representations that are high-dimensional: p is approximately 10,000, $h = 9$, and hence the total number of parameters is approximately 100,000. Because our training data is limited ($n < 40{,}000$), we are in the regime where $p > n$, and ordinary least squares is ill-posed. Hence we use modern statistical techniques for high-dimensional estimation. Sample-size requirements for high-dimensional linear regression are much more favorable than for nonlinear regression: if the true model can be assumed to lie in an ℓ_0 ball (i.e. a sparse signal), then sample size must only scale linearly in the true size of the model, s and logarithmically in the total number of variables, p [101, 78]. Hence another objective for our transformation $\mathbf{f}$ is that the neuron's response be close to a *sparse* linear combination of elements of $\mathbf{f}$.

Feature representations via sparse coding

We leverage the principle of sparse coding in designing our linearizing transform $\mathbf{f}$. Sparse coding was originally motivated from two physiological properties of the early visual cortex. First, individual neurons have high lifetime sparsity. We informally define this to mean that their spike rates have a heavy-tailed distribution (or high kurtosis); see Willmore et al. [97] for a more precise definition. In the case of a binary-valued variable (e.g. spike indicators), this corresponds to low probability of an event. Second, V1 seems to use an overcomplete code in the sense that there are far more output fibers than input fibers from the retina (estimated at 50:1 in macaque V1) [74]. In Olshausen [71] these two properties are taken as constraints for designing a coding scheme for natural image patches. This defines the problem of sparse dictionary learning. We let $\mathbf{a}_1, \ldots, \mathbf{a}_N$ be a large bank of small patches from natural images (16×16 patches in Olshausen [71]). Sparse dictionary learning finds a k-element dictionary $\{\mathbf{d}_1, \ldots, \mathbf{d}_k\}$ that is able to reconstruct each patch accurately using only a small number of dictionary elements. Let m be the total dimensionality of each patch (e.g. 16^2). The dictionary learning problem is then:

$$\hat{\mathbf{D}} = \arg\min_{\mathbf{D}\in\mathcal{C}} \min_{\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N} \frac{1}{2}\sum_{i=1}^{N} \left(\|\mathbf{a}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1\right) \qquad (3.2)$$
$$\text{where } \mathcal{C} = \{\mathbf{D} \in \mathbb{R}^{m\times k} \text{ s.t. } \|\mathbf{d_j}\|_2 \leq 1 \text{ for all } j\}$$

The constraint set $\mathcal{C}$ is used to prevent the dictionary elements from becoming arbitrarily large (this was not originally present in Olshausen [71], but has now become standard). Remarkably, the resulting basis functions (columns of $\hat{\mathbf{D}}$) share the essential properties of V1 simple cell receptive fields: they are band-pass, spatially localized, and oriented. As is shown in Olshausen and Field [73], the dictionary elements decompose natural images into

independent components; this is in line with the hypothesis that neurons tranform natural stimuli into independent features to maximize efficiency of information transmission [3, 4].

Sparse coding model for area MT

Due to the extensive innervation of MT by fibers projecting from V1, we develop a model whose guiding principle is that MT neurons may be well approximated by using a sparse code on the output of V1 neurons. The model has three main layers. The first layer is a caricature of the activity of V1 neurons. It consists of a bank of spatiotemporal wavelet filters of varying orientations, followed by several biologically plausible nonlinearities. The second layer consists of a patch-wise application of the sparse coding transformation. The third layer is a spatial pooling operation. The model is similar to the popular SIFT + sparse coding approach in computer vision [14, 100], but with several modifications that make it more successful in modeling neural data.

In section section 3.3 we provide background on area MT, in section 3.4 we present the model in more detail, in section 3.5 we show results of fitting the model to the MT data, comparing prediction accuracy to other popular approaches. In this section we also compare the representations of natural videos from our model with that of Nishimoto and Gallant [69]; the two representations, though architected in very different ways, end up producing linearizing transformation with a great deal of overlap. We quantify this relationship using canonical correlation analysis (CCA).

3.3 Area MT

This section gives a brief introduction to the functional and physiological properties of MT relevant to this chapter. MT is one of the earliest discovered and most studied extrastriate visual cortex regions. Part of the reason for this is its distinctive physiological and functional characteristics, which enables reliable identification in electrophysiological studies. For a more detailed overview, see Born and Bradley [13].

Lesion studies

Several studies have investigated deficits in performance of visual tasks after chemical lesions of area MT. These results underscore the importance of MT in perception of motion. Newsome et al. [68] found that, after lesions, a monkey's ability to track smoothly translating objects was impaired, as was its ability to perform a rapid saccadic eye movement to a moving target; saccades to stationary objects were unaffected. A later study performed an experiment in which a monkey was trained to detect a subtle motion bias amidst otherwise randomly moving dots; after lesions to MT the subjects only could detect the motion if the signal was made much stronger [67].

Tuning to artificial stimuli

MT has been extensively studied using artificial stimuli similar to those used in early studies of V1, such as moving bars of varying orientation, speed, and direction of motion. In Maunsell and Van Essen [61], the authors recorded from over 100 MT cells in anesthetized, paralyzed macaque using such stimuli. They provided a fairly complete characterization of tuning properties along these dimensions. They found that MT cells become very active when exposed to bars moving in a preferred direction (up to 126 spikes per second), and that most cells' responses were very peaked around this direction. The cells were also tuned to a specific speed of motion, and, again, most showed a sharp peak around the preferred speed (average preferred speed of 32 $^\circ/_s$). Neurons showed comparatively weak responses when exposed to static bars; however, they did show some specificity in orientation tuning, and generally the preferred of orientation of static bars was perpendicular to the preferred direction of motion.

Given these results, it is an open question as to how MT receptive fields differ from those of V1 cells [13]. V1 cells are similarly direction and orientation selective, though relatively more orientation selective than MT neurons. One major theory is that MT solves the aperture problem. This refers to the issue that motion of edges that extend beyond the spatial extent of a cell's receptive field can only be perceived as occuring in a direction perpendicular to the orientation of the edge. There is strong physiological evidence supporting this hypothesis. In Pack and Born [75], the authors stimulated MT neurons with a moving field of bars where the direction of motion was varied relative to the orientation of the bars. They found that MT neurons initially respond to the component of motion perpendicular to bar orientation, but after a period of 60 ms their response encoded the true motion; this tuning was reflected in the pursuit eye movements of the subjects, which were initially in the direction perpendicular to the orientation of the bars no matter what the direction of motion. It is still a matter of debate as to how exactly this is accomplished; it may involve nonlinear combination of inputs from several V1 cells, or may involve innervation V1 hypercomplex cells, which are sensitive to the length of an edge.

Tuning to natural stimuli

Surprisingly, there have been relatively few studies of MT that use naturalistic stimuli. This is surprising since other visual cortical areas have different response properties when stimulated with natural images vs. artificial images [27, 91]. Hence, the paper of Nishimoto and Gallant [69], in which MT cells were recorded while being stimulated with 20,000-40,000 frames of natural video, represents an important milestone. This paper found that a spatiotemporal Gabor wavelet model with several biologically motivated nonlinearities provided an effective linearizing transform for MT neurons. Their major scientific finding was that responses to natural images were consistent with theoretical predictions as to how MT pools inputs of V1 neurons. Specifically, they found that many MT cells' receptive fields, when plotted in the motion-energy Fourier domain, were excitatory along a planar subset, and inhibitory off the plane. This is consistent with the model of Simoncelli and Heeger [84],

who suggested that such receptive fields could account for MT cells' solution to the aperture problem, as well as the observed tuning of MT cells to speed and direction of motion, rather than 2-D content of the stimulus.

3.4 Description of sparse-coding model for MT

The encoding model we develop for MT is an extension of the model introduced in Mairal et al. [59] for V4 neurons. The major challenge in adapting this model is that MT requires features that incorporate motion information. The stimulus set used in Mairal et al. [59] was a series of still images (i.e. uncorrelated over time), whereas the set used in Nishimoto and Gallant [69] was a series of short video clips of 25-30 frames each. It is not sufficient to simply concatenate frames of static image features derived from the pipeline of Mairal et al. [59], as MT is sensitive more complex, nonlinear, low-level spatiotemporal features. It was similarly shown in Nishimoto and Gallant [69] that concatenation of static Gabor wavelet features lead to poor MT models. In this section, we present the model in the context of area MT; the major difference from the V4 model of Mairal et al. [59] is the use of spatiotemporal wavelets in the first layer.

We first give a high-level outline of steps of the model, then each step is described in detail.

Summary of MT Encoding Model

- First, generate features $(\mathbf{x}_1, \ldots, \mathbf{x}_T)$ for each frame of the stimulus video in an unsupervised way.
  - **Layer 1:**
    1. 3-D convolution with 38 spatiotemporal Gaussian derivative wavelets
    2. Positive rectification
    3. ℓ_α pooling and downsampling
  - **Layer 2:**
    1. Extract 3-D patches (width $\times$ height $\times$ wavelet orientations)
    2. Contrast-normalize patches
    3. Code patches in sparse dictionary
  - **Layer 3:**
    1. Pool patches in spatial regions within each frame of video
- Finally, learn a linear model of MT neuron spike rate
  1. Perform trace-norm regularized linear regression of spike count at time t against features $\mathbf{x}_{t-h}$ to $\mathbf{x}_t$
  2. Regularization parameter chosen by testing several values on a holdout set.

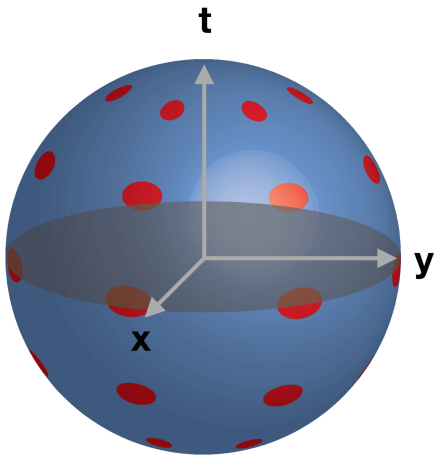

FIGURE 3.1: **Tiling of wavelet angles on spatiotemporal sphere.** Wavelets lying on the xy-plane (i.e. $\theta = \pi/2$) have no motion content and respond to fixed edges. Wavelets oriented exactly along the t-axis (i.e. $\theta = 0$) have no spatial information but respond to changes from light to dark and vice-versa.

First layer: a caricature of a population V1 simple cells

We represent the activity of the population of V1 simple cells as a 3-D convolution of the input video with wavelets of varying orientation, followed by a half-rectification and spatial pooling.

3-D convolution with 38 spatiotemporal Gaussian derivative wavelets

While V1 is generally modeled using Gabor wavelets filters, we use 3-D Gaussian derivative wavelets, as they are faster to compute and are qualitatively very similar to Gabor wavelets. While Gabor wavelets provide a better fit to V1 neurons, we have found empirically that Gaussian derivative and Gabor wavelets are equally successful in providing a first layer for our MT models.

We compute the first layer transformation as follows. Define the spatiotemporal Gaussian function

$$\varphi_{\sigma_x,\sigma_y,\sigma_t}(x,y,t) \propto \exp\left\{-\frac{1}{2}\left[\left(\frac{x}{\sigma_x}\right)^2 + \left(\frac{y}{\sigma_y}\right)^2 + \left(\frac{t}{\sigma_t}\right)^2\right]\right\}$$

Define the usual specification of angles in 3-D using spherical coordinates, (θ, ϕ), then the first-layer convolutional operator is defined as

$$U_{\theta,\phi}[I] \stackrel{\text{def}}{=} \sin\theta\cos\phi\frac{\partial}{\partial x}(I * \varphi_{\sigma_1,\sigma_1,\sigma_t}) + \sin\theta\sin\phi\frac{\partial}{\partial y}(I * \varphi_{\sigma_1,\sigma_1,\sigma_t}) + \cos\theta\frac{\partial}{\partial t}(I * \varphi_{\sigma_1,\sigma_1,\sigma_t}) \quad (3.3)$$

The widths σ_1 and σ_t are parameters. Note that, due to properties of convolution, $\frac{\partial}{\partial x}(I*\varphi) = I * \frac{\partial}{\partial x}\varphi$; hence the above is equivalent to filtering the input video with Gaussian derivatives

of various orientations. The partial derivates $\frac{\partial}{\partial x}(I * \varphi)$ are computed numerically by simply taking the differences of neighboring pixels. To achieve a sufficiently rich feature set, we tile the spatiotemporal sphere with 38 different orientations: $\theta \in \left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi\right\}$, and for each value of $\theta \notin \{0, \pi\}$, 12 different values of ϕ evenly spaced from 0 to 2π. See fig. 3.1 for an illustration of this scheme.

Positive rectification and ℓ_α pooling and downsampling

The operator $U_{\theta,\phi}$ is followed by two biologically plausible nonlinearities. The first is the positive rectification, i.e. define as usual, for a generic function g

$$[g(x)]_+ \stackrel{\text{def}}{=} \begin{cases} g(x) \text{ if } g(x) > 0 \\ 0 \text{ otherwise} \end{cases}$$

This simply reflects the fact that neuron spike rates are positive. Without this nonlinearity, the filter outputs would be redundant, as the output of a linear filter is simply the negative of the output of the filter with opposite orientation.

The second nonlinearity is a spatial ℓ_α pooling operation, which we implement as another convolution with a Gaussian filter after applying a pointwise nonlinearity

$$F_{\theta,\phi}[I] = \left[([U_{\theta,\phi}[I]]_+)^\alpha * \varphi_{\sigma_2,\sigma_2,0}\right]^{1/\alpha}$$

The exponent α and the width σ_2 are parameters. Setting the temporal width to 0 here is an abuse of notation that simply indicates that we are applying a spatial filter only. After applying this spatial pooling, we downsample the output of the operator F, since the Gaussian filter has removed much high-frequency information (this is equivalent to antialiasing before downsampling). We refer to output of the operator F, after downsampling, as an 'orientation map,' since it reflects the energy in the stimulus at various orientations in the spatiotemporal sphere.

Second layer: sparse coding of patches

Instead of working directly on the orientation map $F_{\theta,\phi}$, we now operate on small patches extracted from the orientation map. Each patch captures a very spatially localized representation of the contents of the input video. We apply two nonlinear functions to these patches. These steps operate on the patches in the spatial dimensions, and no longer extract extra motion information. However, motion information is encoded in the patches themselves, due to the use of spatiotemporal gradients above.

Extract 3-D patches (width $\times$ height $\times$ wavelet orientations)

We define $\mathbf{p}_{x,y,t}$ to be the patch whose top left coordinates are (x, y), taken from frame t. Each patch is a 3-dimensional array: $\mathbf{p}_{x,y,t} \in \mathbb{R}^{w \times w \times 38}$. The first two dimensions are spatial

(x and y coordinates), and the third dimension represents the orientation of the first-layer spatiotemporal filter. In practice we use filter with 38 different angles spaced out on the spatiotemporal sphere. The patch width w is another tuning parameter of the model.

Contrast-normalize patches

The first nonlinearity we apply to each patch is a divisive contrast normalization. The particular normalization we use dampens the magnitude of high-energy patches, while not boosting the magnitude of low-energy patches (which may be dominated by noise). It takes the form

$$\mathbf{p}^{\text{norm}}_{x,y,t} \stackrel{\text{def}}{=} \frac{\mathbf{p}_{x,y,t}}{\max\left(\left\|\mathbf{p}_{x,y,t}\right\|_2, c\right)}$$

We set c to be the median ℓ_2 norm of a large bank of natural image patches; effectively, the normalization reduces the energy of patches above the median to a fixed value and leaves the patches below the median untouched.

Code patches in sparse dictionary

The second nonlinear transformation is the sparse coding transformation. This transformation uses the Lasso to represent a patch $\mathbf{p}$ as a sparse linear combination of elements from a learned dictionary $\hat{\mathbf{D}}$. We notate it as follows

$$\tilde{\mathbf{p}}_{x,y,t} = \underset{\boldsymbol{\alpha}\in\mathbb{R}^k}{\arg\min}\left[\frac{1}{2}\left\|\mathbf{p}^{\text{norm}}_{x,y,t} - \hat{\mathbf{D}}\boldsymbol{\alpha}\right\|_2 + \lambda_1\left\|\alpha\right\|_1\right]$$

This is a complex nonlinearity; there is no explicit evidence that such a coding scheme is implemented in the circuitry of the visual system. However, it has been shown that the sparse coding nonlinearity can be plausibly implemented in an artificial neural network [38]. These authors reinterpret the iterations of the standard first-order soft thresholding algorithm for the Lasso [25] as layers of a neural network. They show that a relatively shallow network, with parameters tuned to the particular dictionary, can approximate the ISTA algorithm very well.

We learn the dictionary $\hat{\mathbf{D}}$ by solving the optimization problem given in eq. (3.2), but the dictionary is learned in the domain of the 3-D patches $\mathbf{p}^{\text{norm}}_{x,y,t}$ taken from the orientation maps, rather than raw image patches. This leads to dictionary elements that encode more complex, higher order features. Figure 3.2 shows a visualization of randomly selected dictionary elements. We see some elements are similar to spatiotemporal wavelets, and some describe curves in motion or undergoing transformation. To actually perform the optimization we use the online algorithm presented in Mairal et al. [58] and implemented in the SPAMS toolbox for MATLAB. We train the dictionary on a set of 1,000,000 patches randomly drawn from the stimulus set for a particular neuron.

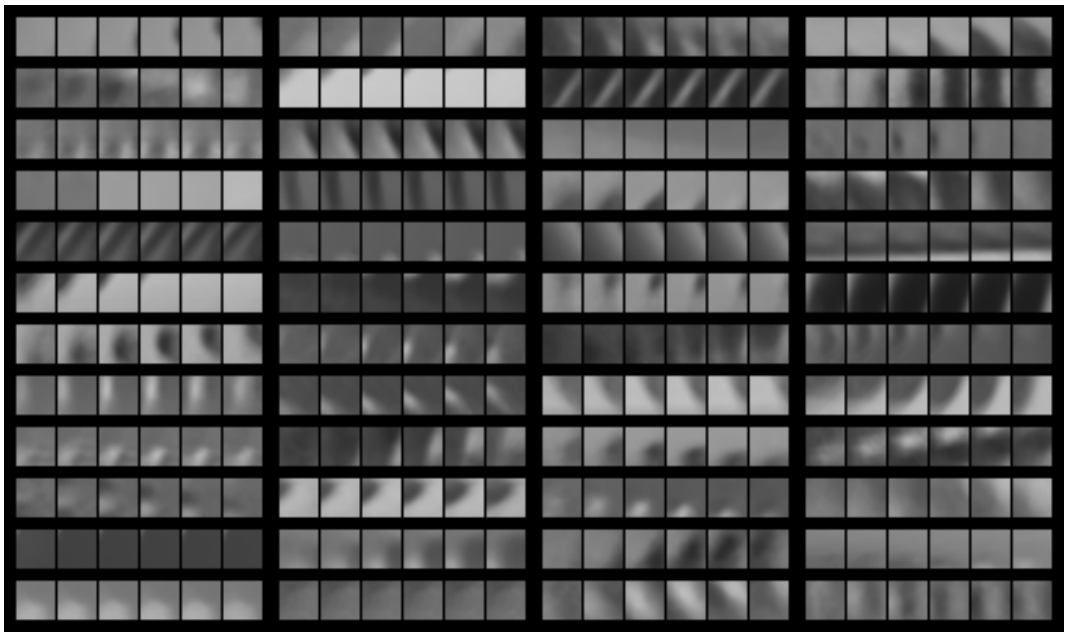

FIGURE 3.2: **Dictionary elements visualized in the original video domain.** 48 randomly selected dictionary elements from a dictionary of size 2048 are shown. Each dictionary element is $16 \times 16 \times 6$ in the original video domain. Dictionaries are learned in the domain of orientation maps; to visualize in the original video domain, we average 100 video patches that are highly correlated with the dictionary element. The dictionary encodes a variety of spatiotemporal features; we can see moving blobs, moving bars, and moving wavelets.

Third layer: spatial pooling

The final step is an element-wise ℓ_2 pooling of the coded patches within regions of the stimulus image. This pooling operation provides further invariance to spatial deformation, and greatly reduces the dimensionality of the output. We define 5 different image regions as in fig. 3.3. For each image region q, we define the pooled feature

$$f_q(t) = \sum_{(x,y) \in R_q} \tilde{\mathbf{p}}^2_{x,y,t}$$

where R_q is the set of (x, y) coordinates in pooling region q, and $\tilde{\mathbf{p}}^2_{x,y,t}$ represents elementwise squaring of $\tilde{\mathbf{p}}_{x,y,t}$. Note that the dimension of $f_q(t)$ is equal to k, the number of dictionary elements in $\hat{\mathbf{D}}$.

In the end, concatenating the $f_k(t)$'s of the different pooling regions gives our final feature representation for the stimulus frame at time t. It is perhaps worth taking stock of the set of tuning parameters of the model. Table 3.1 shows the full set. In the first layer filters, we fixed the values $\sigma_1 = 1$ pixel (width of first spatial filter), $\sigma_2 = 2$ pixels (width of second spatial filter, as we are downsampling by a factor of 4, this performs standard anti-aliasing).

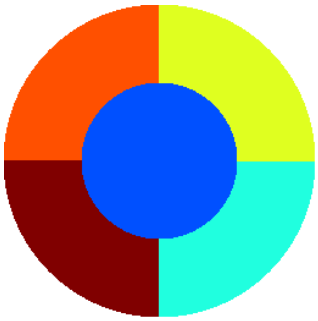

FIGURE 3.3: **Pooling regions for final representation of video frames.** Patches lying within each regions are combined with elementwise ℓ_2 pooling. We use this circular layout of regions to better match the center-surround dynamics of neurons.

TABLE 3.1: **Parameters of our two-layer sparse coding pipeline.**

Parameter	Description
σ_1	Spatial frequency of Gaussian derivative wavelet
σ_t	Temporal frequency of Gaussian derivative wavelet
α	Degree of exponent for spatial pooling of convolutional outputs
σ_2	Width of Gaussian filter prior to downsampling the orientation maps
w	Patch size after downsampling
k	Size of dictionary
λ_1	ℓ_1 penalty for sparse coding

In the second layer, we set $w = 4$ pixels (patch size), $k = 2048$ (dictionary size), and $\lambda = 0.1$ (penalty for sparse coding). These are standard parameter values used in computer vision. The other two parameters were tuned on a small subset of the MT neurons using the holdout set. We found that $\alpha = 4$ (exponent for spatial pooling in first layer) and $\sigma_t = 1$ pixel (temporal width of filters in first layer) led to the best predictive models. Aside from these two tuning parameters, the three-layer feature representation just described was designed and trained without supervision from neural data. It is based on a simple model of the activity of V1 neurons and the principle of sparse coding. We will see that it provides an effective linearizing transform for modeling MT neurons.

Learning the top-level linear model of neuron spike counts

With this linearizing transformation in hand, the final step in our model is learning the linear map connecting the above video features to expected spike count for each MT cell. The feature set described above is very high-dimensional: with the stated parameters, the final dimensionality is $k \times$ (number of pooling regions) $\times$ (number of delays) $= 92{,}160$. Hence the number of features is greater than the number of training examples, and we employ penalized regression techniques that exploit low-dimensional structure. The most popular

such techniques are the least absolute shrinkage and selection operator (Lasso) [90] and ridge regression [43]. While these are both effective in this problem, find that trace-norm penalized regression performs best.

This trace-norm penalization method is motivated by the space-time separability of many neural receptive fields [62]. Perfect space-time separability would mean that the spatiotemporal receptive field is, in fact, perfectly described by the tensor product of a two-dimensional spatial receptive field and a one dimensional temporal receptive field. In the context of our model, we can define this as follows. First, define the coefficient matrix

$$\mathbf{B} \stackrel{\text{def}}{=} \begin{pmatrix} | & | & & | \\ \hat{\boldsymbol{\beta}}_1 & \hat{\boldsymbol{\beta}}_2 & \cdots & \hat{\boldsymbol{\beta}}_h \\ | & | & & | \end{pmatrix}$$

This is simply the coefficient vector of our model rearranged into a matrix. Perfect space-time separability would imply that $\mathbf{B}$ should have rank 1. We hypothesize the MT models should be approximately space-time separable, and its singular values should be sparse or delay quickly. The trace-norm (i.e. the sum of singular values) provides a convex heuristic for the rank of a matrix [32], and we use it as a penalty term when performing linear regression:

$$\hat{\mathbf{B}} = \arg\min_{\mathbf{B}} \frac{1}{2} \sum_{t=h}^{T} \left[\begin{pmatrix} | & & | \\ \mathbf{x}_t & \cdots & \mathbf{x}_{t-h+1} \\ | & & | \end{pmatrix}^T \mathbf{B} \right] + \lambda_2 \|\mathbf{B}\|_* \tag{3.4}$$

Here $\|\cdot\|_*$ is the trace norm. The tuning parameter λ_2 was set individually for each neuron by testing several values and using a small subset of the training data as a holdout set. This regularization was previously shown to be effective in fitting V4 receptive field models in Mairal et al. [59].

3.5 Prediction results

Predictive accuracy

Using the above pipeline, we fit encoding models of the 52 MT neurons analyzed in Nishimoto and Gallant [69]. To measure we accuracy of the models, we calculate the correlation between the prediction of the model and the observed spike rates of the neurons on the holdout sets. As described in Nishimoto and Gallant [69], the holdout data consists of averaged spike counts from repeated presentations of the stimuli; each frame in the holdout set was repeated between 5 and 20 times (the number varies due to the attentional shifts of the macaques during stimulus presentation). Although the model fitting procedure described in eq. (3.4) optimizes for mean squared error (MSE), we evaluate using the correlation coefficient both because it allows for comparison to the results in Nishimoto and Gallant [69] and is invariant to scale and constant biases in prediction.

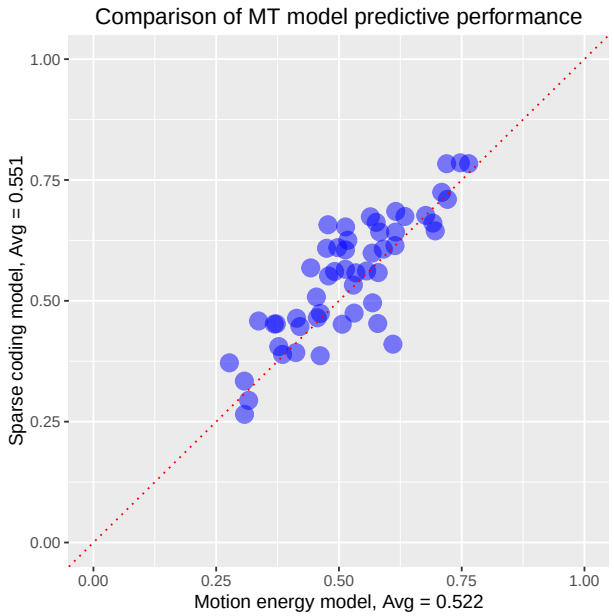

FIGURE 3.1: **Scatter plot of correlation scores for the 3 layer sparse coding model of section 3.4 and the Gabor motion energy model of Nishimoto and Gallant [69].** The sparse coding model has better prediction accuracy when averaged over the 52 MT ($p = 0.002$, Wilcoxon signed rank test). However, the improvement is not uniform, as many cells are better modeled by the Gabor motion-energy wavelet model.

When calculating the correlation, we first convolve the time series of holdout spike rates with a Gaussian filter, following the same procedure as in Nishimoto and Gallant [69]. This was done to smooth over discontinuities introduced by the tiling of separate recording sessions in the averaging procedure, and we observe that it boosts the correlations of the models. Figure 3.1 compares the predictive accuracy of our encoding model with that of the Gabor motion-energy model used in Nishimoto and Gallant [69]. While the two procedures give similar accuracies, the procedure of section 3.4 is statististically significantly higher when averaged over the 52 neurons. The average correlation for our models is 0.551, while the average correlation for the Gabor motion energy model is 0.522. A Wilcoxon signed-rank test for the population of 52 differences in scores yields a p-value of 0.002. Our model yields a 7% increase in predictive accuracy on average.

We have just made a comparison between two complex end-to-end modeling pipelines that are distinct in many ways; hence it is difficult to establish what properties of our model lead to the improvement in performance. One plausible hypothesis is that the improvement in fact comes from the first layer of our pipeline, where we use Gaussian derivative wavelets as opposed to the Gabor wavelets used in Nishimoto and Gallant [69], and that the sparse coding nonlinearity is simply a reparametrization that does not effect (or perhaps even hinders) model performance. To investigate this hypothesis and demonstrate that the sparse

coding step is essential, we have fit models of all neurons using a modification the first layer of our model as the feature representation. We do not attempt to use the raw first-layer outputs because they are extremely high-dimensional and likely would swamp the amount of training data, even with regularization: with the parameter values defined in section 3.4, the number of features (including delays) would be $32 \times 32 \times 36 \times 9 = 331{,}776$. Instead, we optimized the first layer in several ways to make it more compact and similar to the model of Nishimoto and Gallant [69], which has 5956 wavelets $\times$ 9 delays $= 53{,}604$ dimensions. A major component of the model in Nishimoto and Gallant [69] is the use of wavelets of different spatial scales and frequencies. We have implemented something similar in our pipeline by using a spatial pyramid [52]. Instead of using Gaussian derivative filters of varying scales, we apply a fixed-scale filter to rescaled versions of the original stimulus image, and then concatenate the representations. While the two operations are not identical, they give similar results. The full pyramid we use is a concatenation of features from the stimulus image scaled to 128×128 (original dimensions), $96 \times 96, 64 \times 64$, and 32×32. Also, to reduce overall dimension, we have performed more aggressive subsampling: for each level of the pyramid, we subsample to 8×8 images after applying the wavelet filters and pointwise nonlinearities. The final dimensionality of the feature set is $4 \times 8 \times 8 \times 9 = 82{,}944$

We find these that first-layer features do not explain the overall superior performance of our model. The predictive accuracy of models built from the first-layer features is significantly inferior to that of the model of [69]. The average correlation score over the 52 neurons is 0.481. Figure 3.2 compares the predictive accuracies.

Thse sparse-coding layer, then, is crucial for the success of our feature representation in linearizing MT responses. To further establish this point, we compare predictive accuracy of models based on the pyramid of first-layer features and models using the full three-layer pipeline which includes the sparse coding map. We perform an additional analysis to assess which individual neurons are better fit by the three-layer pipeline. To motivate this, we see that it is clear from fig. 3.1 that the improvement from our three-layer model, while significant, is quite heterogenous over the set of neurons. Some neurons have higher correlation scores using the model of Nishimoto and Gallant [69], and many neurons lie very close to the diagonal, meaning the two models perform very similarly. We perform a significance test for model selection on an individual neuron basis; the particular test we use is quite simple and is described in the next section. Figure 3.3 shows the scatter plot of correlation scores between models based on one-layer and three-layer feature representations, and neurons are additionally colored according to whether the three-layer representation yields a significantly better model according to our test. Adjusting for multiple comparisons by setting the FDR at 0.05 and determining the p-value cutoff with Benjamini-Hochberg procedure [8], 13 neurons are significantly better modeled using the three-layer feature set, and 1 neuron is significantly better modeled using the one-layer feature set only. The remainder do not show a significant difference.

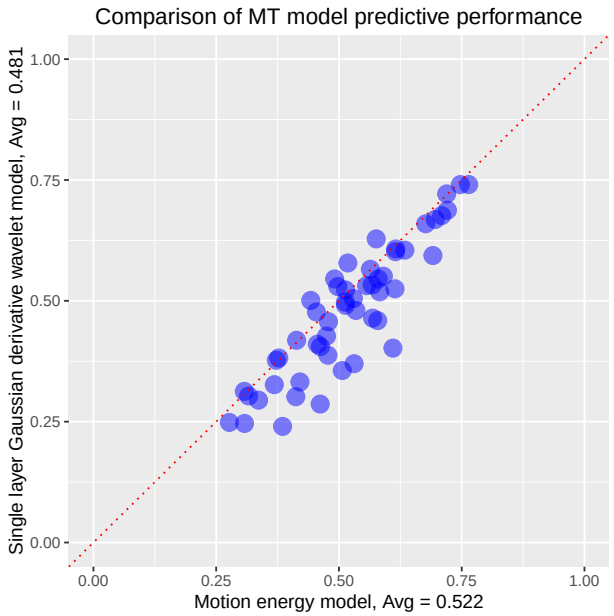

FIGURE 3.2: **Scatter plot of correlation scores for layer 1 of our model (Gaussian derivative wavelets) vs. the Gabor motion energy model of Nishimoto and Gallant [69].** The Gabor motion energy model performs significantly better, indicating that layer 1 of our model cannot account for the superior performance shown in fig. 3.1

Description of the significance test for model selection

The significance test whose results are shown in fig. 3.3 is a refinement of traditional model selection procedures such as AIC and BIC which make a discrete choice between competing models. The significance test we describe here, in a sense, offers a third option, which is that there is no evidence to prefer either model. We test the null hypothesis of Vuong [95], that both models are equally close, in the sense of Kullback-Leibler distance, to the true distribution. We do not use the directed tests of non-nested models of Cox [23, 22], as we do not have particular reason to prefer either model, and we do not seek more power in testing either alternative. We treat the fitted models as fixed (i.e. we condition on the training data), and perform the test on the holdout data. Assuming errors in observed spike rate are normally distributed, the Vuong test boils down to a simple paired, two-sided t-test of squared residuals from the two models. Because of dependence in errors due to correlations in the noise of spike trains, as well as correlations between frames of video, we subsampled the test set by a factor of 5. The normality assumption is a strong one for these data, since relatively few repeats are averaged in the test set, and hence the central limit theorem may not apply. Sacrificing this assumption means the test no longer has an interpretation in terms of Kullback-Leibler divergence; the paired t-test of residuals still has value as an exploratory tool, however.

Note that, as we see in fig. 3.3, this test is not a simple function of the difference in

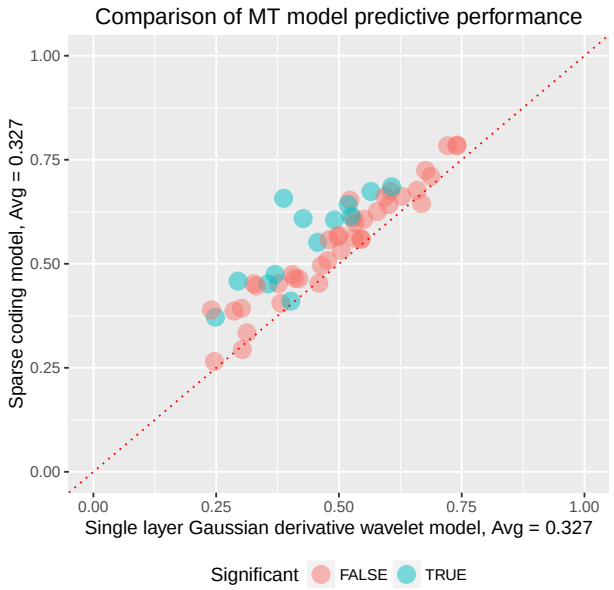

FIGURE 3.3: **Scatter plot of correlation scores for models based on the full three-layer sparse coding feature set of section 3.4 and models based on layer 1 features only.** Colors of points show the result of a significance test for model selection; rejection (blue points) indicate that the three-layer sparse coding pipeline provides a significantly better fit for that neuron.

correlation scores or MSE. The test differentiates between the case where the residuals from one model are consistently slightly smaller than those of the other model and the case where the residuals from one model are smaller, on average, than those of the other model, though the differences are quite varied and can be both positive and negative. For instance, we see in fig. 3.3, that there is a significant cell that lies very close to the diagonal; this would correspond to the former case. Similarly, some non-significant cells lie far above the diagonal, corresponding to the latter case.

Since we are testing many cells at once, as stated, we set the threshold for significance using the Benjamini-Hochberg procedure. The p-values for all 52 MT neurons are plotted in fig. 3.4 along with the BH cutoff line. The tests reveal that, for most neurons, the three-layer and one-layer feature representations are essentially indistinguishable. The additional sparse coding layer is important to capture the activity of only a small subset of neurons. However, since only one cell is significantly better modeled with first-layer features only, we can conclude that the sparse coding layer preserves most information from the first layer that is relevant for MT cells, while adding additional flexibility to the representation. In the next section we examine, in more detail, the similarity between the feature representations to better understand the nonlinear transformation performed by sparse coding.

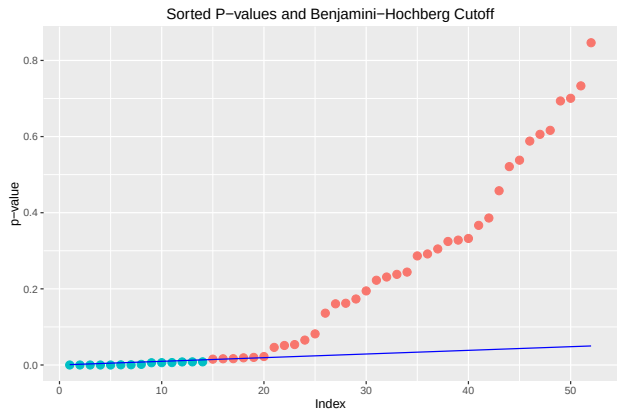

FIGURE 3.4: **Benjamini-Hochberg procedure for model selection test between 1-layer and 3-layer models of 52 MT neurons.** FDR was set at 0.05

3.6 Similarity between stimulus representations

We have seen in the previous section that our three-layer sparse coding representation leads to more accurate MT models, on average, than either the Gabor motion-energy wavelet model of Nishimoto and Gallant [69] or a single-layer Gaussian derivative wavelet representation that does not perform sparse coding. However, for many neurons, the models perform very similarly, or are perhaps even indistiguishable when the model selection test is applied. Hence, although the feature representations are architected very differently, they arrive at feature sets which contain a great deal of overlapping information when used as the basis for a linear model; indeed they are indistinguishable from the standpoint of many MT neurons. In this section we explore these relationships, quantify the amount of overlapping information, and examine the action of sparse coding in more detail.

At one extreme, it is plausible that two different feature extraction architectures could arrive at representations that are related by a unitary linear transformation (i.e. a rotation). From the standpoint of any unitarily invariant regression procedure (which includes OLS, ridge regression, and in our specific setup, trace-norm regularization), such feature sets would be completely indistinguishable - they would give the same model predictions, and hence have the same accuracy. At the other extreme, the two architectures could give feature sets that span subspaces that are orthogonal in expectation over natural images. The actual neuron's spiking rate, then, may lie closer to one subspace than another, or, in fact, may have the same distance from both subspaces (in which case the two models are equally poor at providing a linearizing transform for the neuron's behavior). Our feature representations lie somewhere in between these two extremes.

We characterize these relationships by calculating principal angles between the spaces described by the feature sets; in the regression setting this is equivalent to performing canonical correlation analysis (CCA) [45]. Given two feature sets (i.e. design matrices) $\mathbf{X}_1 \in \mathbb{R}^{n \times p_1}$ and $\mathbf{X}_2 \in \mathbb{R}^{n \times p_2}$, canonical correlation analysis finds contrasts $\mathbf{u} \in \mathbb{R}^{p_1}$ and $\mathbf{v} \in \mathbb{R}^{p_1}$ that

maximize the correlation between the resulting vectors

$$\rho_1 = \max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^T \mathbf{X}_1^T \mathbf{X}_2 \mathbf{v}}{\|\mathbf{X}_1\mathbf{u}\|_2 \|\mathbf{X}_2\mathbf{v}\|_2}$$

We define $\mathbf{u}_1, \mathbf{v}_1$ as the vectors that solve this optimization. The quantity ρ_1 is also the cosine of the first principal angle between the column spaces of the matrices $\mathbf{X}_1$ and $\mathbf{X}_2$. Further principal angles, or canonical correlations, can be defined recursively by optimizing the correlation under an orthogonality constraint relative to the earlier vectors:

$$\rho_j = \max_{\mathbf{u},\mathbf{v}} \left\{ \frac{\mathbf{u}^T \mathbf{X}_1^T \mathbf{X}_2 \mathbf{v}}{\|\mathbf{X}_1\mathbf{u}\|_2 \|\mathbf{X}_2\mathbf{v}\|_2} \text{ s.t. } \mathbf{u}^T \mathbf{X}_1^T \mathbf{X}_1 \mathbf{u}_k = \mathbf{u}^T \mathbf{X}_1^T \mathbf{X}_2 \mathbf{v}_k = \mathbf{v}^T \mathbf{X}_2^T \mathbf{X}_2 \mathbf{v}_k = 0 \text{ for all } k < j \right\}$$

The vectors $\mathbf{X}_1\mathbf{u}_j, \mathbf{X}_2\mathbf{v}_j$ are called the jth pair of canonical variables. If two feature sets have many high canonical correlations, this does not necessarily mean that they will have similar performance in modeling neuron spike rates: the neuron spike rate could, in principle, be orthogonal to the pairs of canonical variables. However, the canonical correlations give a good indication, before considering neural data, of how much representational similarity is shared between the feature sets of the two models.

Canonical correlation analysis of three-layer model and spatiotemporal Gabor-wavelet model

We first compare our three-layer sparse coding model with the spatiotemporal Gabor-wavelet model of Nishimoto and Gallant [69]. In this comparison, we do not tile the design matrix with time delays, as we are interested in the similarity of the stimulus representations frame-by-frame. Because both models are still very high-dimensional, unreguarlized CCA is very unstable and is prone to overfitting. Hence we used a regularized version of CCA introduced in Bach and Jordan [2], which solves the following. Let λ be a regularization parameter. Then define

$$\rho_1 = \max_{\alpha_1,\alpha_2 \in \mathbb{R}^n} \frac{\alpha_1^T \left(\mathbf{X}_1\mathbf{X}_1^T\right)\left(\mathbf{X}_2\mathbf{X}_2^T\right)\alpha_2}{\left(\alpha_1^T \left(\mathbf{X}_1\mathbf{X}_1^T + \lambda I\right)^2 \alpha_1\right)^{1/2} \left(\alpha_2^T \left(\mathbf{X}_2\mathbf{X}_2^T + \lambda I\right)^2 \alpha_2\right)^{1/2}}$$

The linear transformations of the original design matrices are then $\mathbf{u_1} = \mathbf{X}_1^T\hat{\alpha}_1, \mathbf{v_1} = \mathbf{X}_2^T\hat{\alpha}_2$, where $\hat{\alpha}_1, \hat{\alpha}_2$ solve the above optimization. We solve this using the method given in Bach and Jordan [2], which recasts it as a generalized eigenvalue problem. We perform this regularized CCA on the natural video stimulus set from Nishimoto et al. [70]. We pick the regularization parameter by splitting the data and validating on a holdout set: we fit the CCA parameters using 90% of the data, and then use the learned vectors $\mathbf{u_1}, \mathbf{v_1}$ to recalculate the top canonical correlation on the holdout set consisting of the last 10% of the data. When test a series of possible values of λ and choose the one that leads to the largest top canonical correlation on the holdout set.

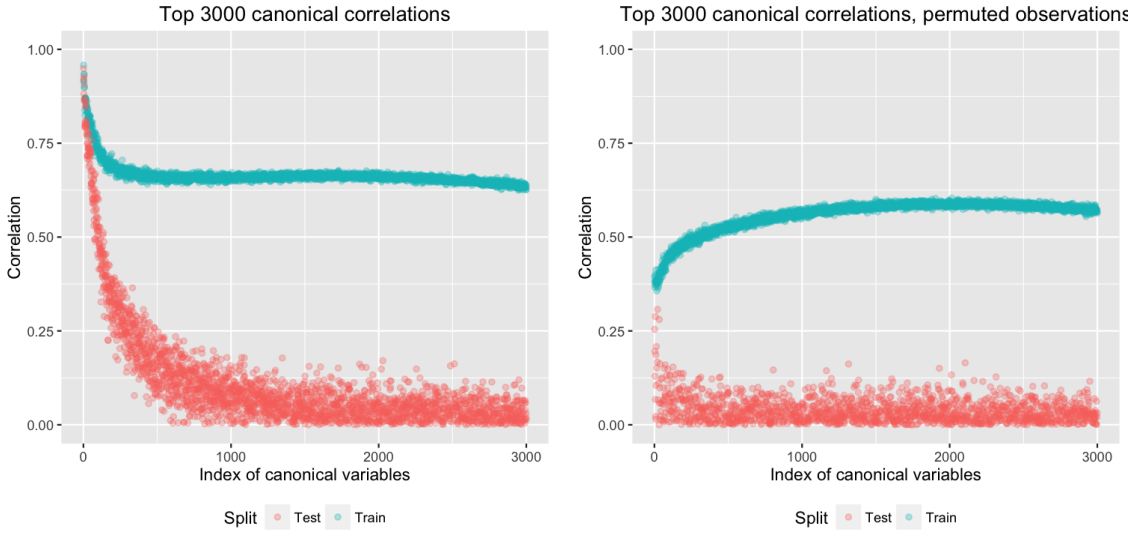

FIGURE 3.1: **Regularized canonical correlation analysis of three-layer sparse coding pipeline and Gabor motion energy model.** Both plots show canonical correlations on the training set (blue points) and on a test set (red points). The test set correlations are similar to the training set correlations for the initial canonical variables, but are much smaller for later variables, due to overfitting. The right plot provides a null model for the left plot: by permuting rows of one of the design matrices, the two signals are made independent, and any correlation found is noise.

To establish a null model for comparison, we permute the rows of one of the design matrices, re-run CCA, and examine the resuling correlations. The permutation makes the two sets of design matrices independent, and should yield a pair of design matrices where the top canonical correlation is zero in expectation. In fig. 3.1 we show the resulting correlations. We see that, in the unpermuted data, the top canonical correlations are very high in both the training and test set: the first canonical correlation is 0.95, and the top 50 canonical correlations are all larger than 0.65 in the test set. Hence the two representations encode much overlapping information. This explains why many MT neurons are equally well-modeled by the two pipelines: their responses must lie somewhere in the space of these top canonical variables. We also see that, even after aggressive regularization, the lower canonical variables are purely the result of overfitting. Although their training set correlations are quite high (the first 2000 do not drop below 0.65), most do not attain test-set correlation that is any better than that of the permuted datasets. Using the holdout set correlations of the permuted dataset as a null distribution, we calculate empirical p-values for the holdout set correlations of the original datasets. Setting the FDR at 0.05, and applying the Benjamini-Hochberg-Yekutieli procedure (allowing for arbitrary dependence between the p-values [9]) we find that there are 258 significant canonical variables, with the smallest correlation being 0.29.

Canonical correlation analysis of representations at the patch level

Considering that the pipeline of section 3.4 contains a highly nonlinear sparse coding operation, it is surprising that it yields features that can be so highly correlated with those of the spatiotemporal Gabor wavelet model. However, the model of Nishimoto and Gallant [69] contains several nonlinear operations (divisive normalization, compressive nonlinearity, ℓ_2 pooling) that may, in fact, perform actions that are similar to our sparse coding layer. Thus, we examine the degree of nonlinearity of the sparse coding transform itself. To do so, we perform CCA of orientation map patches before and after sparse coding. We use patch size 4×4 (giving total feature dimension $4 \times 4 \times 38 = 608$). To reduce dimensionality and prevent overfitting of CCA, we use a dictionary of size 512, which is qualitatively very similar to the dictionary of size 2048 that we use in fitting the encoding models. We extract 700,000 patches from the stimulus set and perform unregularized CCA. The results are shown in fig. 3.2. The canonical correlations are surprisingly large: the top 100 are all above 0.5. This can explain the representational similarity of the two pipelines shown in fig. 3.1. Although the sparse coding operation is followed by pooling operations in our pipeline, which may further decorrelate the representations, sparse coding does leave some information relatively intact. MT cells whose responses lie in the span of these top canonical variables would be equally well modeled by both pipelines.

Regression of dictionary-coded features against layer 1 features

To gain a more detailed understanding of the nonlinear action of sparse coding transform, we perform a complementary analysis to CCA. We use the same procedure as in the previous section, extracting 700,000 patches from the orientation maps, and coding them in the sparse dictionary. Instead of running CCA, we perform column-wise linear regression of dictionary-coded features against the patch representations before sparse coding; that is, we calculate

$$\boldsymbol{\beta} = \left(\mathbf{X}_1^T \mathbf{X}_1\right)^{-1} \mathbf{X}_1^T \mathbf{X}_2$$

We split the data into training and holdout sets, and then, for each dictionary element, we calculate a correlation score of on the holdout set (i.e. correlation of predicted dictionary activation vs. actual dictionary activation). High correlation indicates that a particular dictionary element can be accurately represented by a linear combination of 1st-layer features, while low correlation indicates that the dictionary element is not easily represented by such a linear combination. Figure 3.3 shows a sample of dictionary elements both at the high and low ends. The dictionary elements whose activations are more easily predicted with 1st-layer features (top plot) tend to resemble moving bars or wavelets. Those that are not easily predicted (bottom plot) tend to encode complex curved features or textural elements.

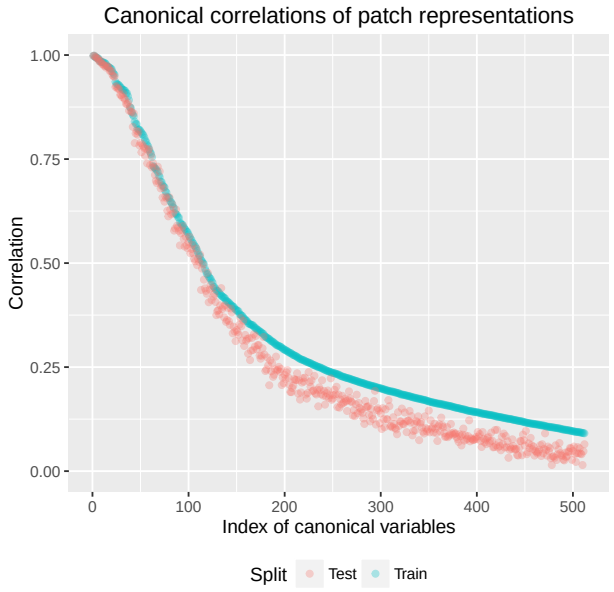

FIGURE 3.2: **CCA of orientation map patches before and after sparse coding.** We see that sparse coding, despite being a highly nonlinear operation, yields features that can have very high correlation with the original patch representation. Here the original patch representation has dimension 608, and a dictionary of size 512 was used for sparse coding. CCA was trained with a large number of patches (700,000), so regularization was unnecessary. We see that training and test set correlations are very similar.

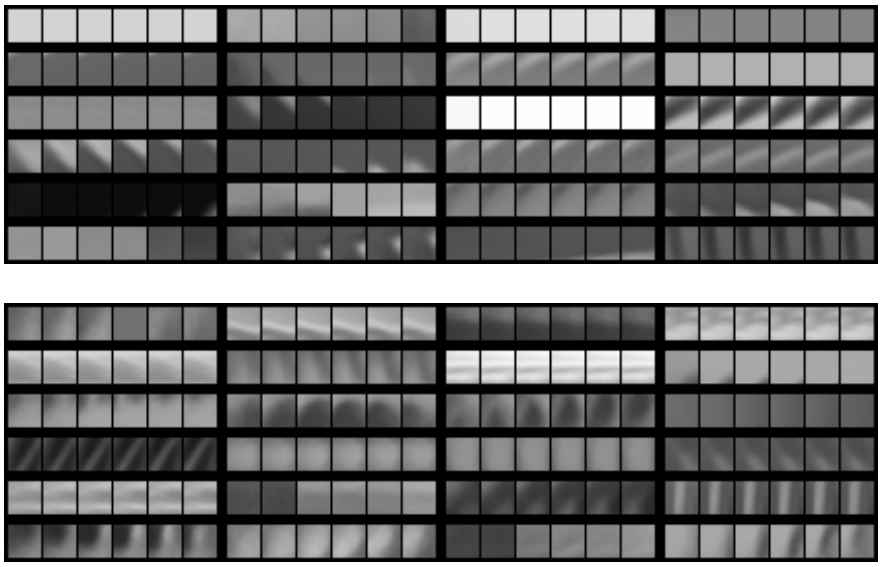

FIGURE 3.3: **Dictionary elements that are well-modeled (top) and poorly-modeled (bottom) as linear combinations of 1st-layer features.** The top plot shows the 48 dictionary elements with highest holdout set correlation; the bottom plot shows the 48 elements with lowest correlation. The dictionary elements that are easily modeled by 1st-layer features are simple moving bars or static luminances. The dictionary elements that are poorly modeled are generally more complex curved features.

3.7 Discussion

Ablation studies and electrophysiological experiments using artificial stimuli have clearly shown the importance of area MT in tasks related to motion cognition. There have been many electrophysiological studies of MT using artificial stimuli, which have revealed MT's tuning to direction and speed of moving bars. There have been surprisingly few investigations of MT using natural stimuli. A major exception is the paper of Nishimoto and Gallant [69], which is the source of the data analyzed in this chapter. They present a model of MT that uses a bank of spatiotemporal Gabor wavelet filters, followed by several biologically plausible nonlinearities, as a linearizing transform. Their results largely confirms the theoretical predictions of Simoncelli and Heeger [84], which proposes that MT receptive fields integrate the results of V1-like filters that lie along a plane in the spatiotemporal Fourier domain. However, they do find that many MT neurons do not precisely follow this model, and have receptive fields that are more selective to specific spatial frequencies than expected.

The model developed in this chapter differs from that of Nishimoto and Gallant [69] and Simoncelli and Heeger [84] in that the outputs of V1-like filters are first transformed by a nonlinear sparse-coding step before being used as features for linear models of MT neurons. The sparse coding step is motivated by physiological evidence of sparsity in neuron spike rates [97], as well as success of sparse coding in explaining tuning properties in other cortical regions [72]. We find that, although this model was originally developed to study shape selectivity of V4 neurons [59], it also describes MT neurons very well. It performs better, averaging over 52 MT cells, than the spatiotemporal Gabor wavelet model of Nishimoto and Gallant [69]. This success suggests that sparse coding may provide a very good approximation, in general, of the transformation that takes place between striate and extrastriate visual cortex regions.

However, we find that, for many MT neurons, the sparse coding layer does not significantly affect the prediction accuracy of the model. A significance test for model selection reveals that only 13 of 52 MT neurons are significantly better modeled with the inclusion of the sparse-coding layer. Thus, we have investigated the degree to which the sparse-coding transformation actually changes the feature representation. We have found through canonical correlation analysis that the sparse coding step, surprisingly, preserves a great deal of information, in the sense that many of the principal angles between the two feature sets (before and after sparse coding) are very small. This explains why the majority of MT neurons in our dataset are not significantly better modeled by features based only on the first, V1-like, layer of our pipeline, than on the full 2-layer pipeline incorporating sparse coding. However, the nonlinear operation of sparse coding adds flexibility to the representation, which we can see by visualizing the features, and generates many video features that cannot be explained by linear combinations of the outputs of the filters in the first layer of our model.

Our results underscore the heterogeneity of receptive fields of neurons in MT. Wavelets constructed using the classical parameters of orientation, direction of motion, and speed of motion accurately describe many MT neurons. However, our results indicate that some MT neurons are tuned to more complex video features and shapes that cannot be constructed

from linear combinations of spatiotemporal wavelets.

Chapter 4

Supervised neighborhoods for distributed nonparametric regression

4.1 Introduction

Modern datasets collected via the internet and in scientific domains hold the promise for a variety of transformational applications. Non-parametric methods present an attractive modeling choice for learning from these massive and complex datasets. While parametric methods that optimize over a fixed function space can suffer from high approximation error when the task is complex, non-parametric methods augment the capacity of the underlying statistical model as the datasets grows in scale and diversity. Moreover, the scale of these datasets makes non-parametric estimation feasible even when considering minimax rates for non-parametric functions that require the data to scale exponentially in the feature dimension.

Global non-parametric models, e.g., neural networks and kernel methods, learn complex response surfaces and can be quite expensive to train on large-scale datasets. Both machine learning and computer systems researchers are actively exploring techniques to efficiently embed these learning approaches into parallel and distributed computing frameworks, e.g., [79, 106, 28, 49, 57]. In contrast, local models are simple, interpretable, and provide an attractive computational profile, by potentially drastically reducing training time at the expense of a moderate increase in test time.

However, to date local methods have been rarely employed on large-scale learning tasks due to both statistical and computational concerns. Statistically, classical methods struggle with even a moderate number of features due to the curse of dimensionality. Although these local methods are minimax optimal, this minimax rate is quite conservative when the response does not involve all dimensions. Although local approaches relying on 'supervised neighborhoods,' e.g., DANN [41], CART [16], Random Forests [15], and Rodeo [51], demonstrate empirical and theoretical success at overcoming this dimensionality issue, they do so at great computational expense.

In this work, we address these statistical and computional issues, focusing on the problem of non-parametric regression. We propose a novel family of **S**uperv**i**sed **Lo**cal modeling methods (SILO) that build on the interpretation of random forests as a local model to identify supervised neighbors. Like k-NN methods, the width of our neighborhoods adapts to the local density of data: higher density leads to smaller neighborhoods while lower density means neighborhoods spread out and borrow strength from distant points. Unlike k-NN or local polynomial methods, our neighborhoods are determined by the shape of the response rather than some fixed metric (hence they are supervised neighborhoods).

Additionally, we observe that datasets of the size we need to fit nonparametric functions are often stored and processed in a distributed fashion. We thus devise an efficient distributed variant of SILO, which naturally targets massive, distributed datasets. In this setting, worker nodes at training time independently calculate supervised neighborhoods using information from distinct random forests. Given a test point, the master node gathers supervised neighborhood data from all workers, and then trains a local linear model using this neighborhood data.

To summarize, our contributions are as follows:

- We introduce a novel adaptive local learning approach, SILO, based on the idea of supervised neighbors.
- We present a distributed variant of SILO that is well suited for large-scale distributed datasets.
- We prove consistency of SILO under a simple model of random forests.
- We show experimentally that SILO outperforms both standard random forests and Euclidean-distance based local methods in single node settings.
- We implement the distributed variant of SILO in Apache Spark [102], and demonstrate its favorable performance relative to the default Spark/MLlib [65] Random Forest implementation.

4.2 Methods based on local models

There are three ingredients to a method for local model-based nonparametric regression: a loss function $L : \mathbb{R} \to \mathbb{R}$, a weight function $w(\cdot, \cdot)$, and a function space $\mathcal{F}$. In this work, we assume that the loss function is the squared loss, i.e., $L(u) = u^2$. The weight function is a mapping $w(\cdot, \cdot) : \mathbb{R}^p \times \mathbb{R}^p \to [0, \infty)$, which may be fixed, or may depend on the training dataset. $\mathcal{F}$ is a family of functions where $f \in \mathcal{F}$ maps $\mathbb{R}^p \to \mathbb{R}$. Generally $\mathcal{F}$ is a fairly simple function space, as it need only provide a good approximation to the response surface locally. Common function spaces are constant functions, linear functions, or higher-order polynomials of fixed degree.

We now describe the general form for local modeling methods. Assume we have a set of training data pairs, $\{(y_i, \mathbf{x}_i)\}_{i=1}^n$. Given a test point $\mathbf{x} \in \mathbb{R}^p$, we define a function $\hat{g}(\mathbf{x})$ which

approximates $E(y \mid \mathbf{x})$, via the following two-step process:

$$\text{let } \hat{f}_{\mathbf{x}}(\cdot) = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n} w(\mathbf{x}_i, \mathbf{x}) \left(y_i - f\left(\mathbf{x}_i - \mathbf{x}\right)\right)^2 \tag{4.1}$$

$$\hat{g}(\mathbf{x}) = \hat{f}_{\mathbf{x}}(\mathbf{0}) \,. \tag{4.2}$$

For instance, we can consider k-NN regression under this framework by first defining $\mathcal{K}_{\mathbf{x}}$ as the set of $\mathbf{x}$'s k nearest neighbors. Then, $w(\mathbf{x}_i, \mathbf{x}) = 1$ if $\mathbf{x}_i \in \mathcal{K}_x$ and zero otherwise, and $\hat{f}_{\mathbf{x}}(\cdot)$ is a constant function that returns $\frac{1}{k} \sum_{x_i \in \mathcal{K}_{\mathbf{x}}} y_i$. Partitioning methods like CART [16], and random forests [15] can also be seen as examples of local models with constant $\hat{f}_{\mathbf{x}}(\cdot)$ functions.

LOESS [20] uses a fixed weighting function, and fits local polynomial functions. We describe the particular setting of linear functions in more detail in section 4.3 (see eqs. (4.8) and (4.9)). Other methods exist that are similar in spirit to the local modeling framework defined in eqs. (4.1) and (4.2) but do not fit precisely with our formulation, e.g., tree-based piecewise polynomials models [19], and multivariate adaptive regression splines [37].

Euclidean distance-based local methods

A common form for the weight function w is the following:

$$w(\mathbf{x}_1, \mathbf{x}_2) = k(\|\mathbf{x}_1 - \mathbf{x}_2\|^2) \tag{4.3}$$

where $\|\cdot\|$ is the Euclidean, or L^2 norm; $k : [0, \infty) \to [0, \infty)$ is a univariate kernel function. Common examples include the rectangular, triangular, Epanechnikov, cosine, and Gaussian kernels. If the local model family $\mathcal{F}$ is chosen to be polynomials of degree ℓ, such an estimator is denoted as an $LP(\ell)$ method (we adopt the notation of Tsybakov [92]). If the true conditional expectation function $E(y \mid \mathbf{x})$ belongs to the Hölder class $\Sigma(\beta, L)$, then, with appropriate scaling of the kernel function, the mean square error of the $LP(\lfloor \beta \rfloor)$ estimator converges to zero at the minimax optimal rate of $n^{-\frac{2\beta}{2\beta+p}}$. In practice, despite their optimality properties, $LP(\ell)$ estimators are not generally applied to higher dimensional problems where some dimensions may be irrelevant. As an example, fig. 4.1 shows the empirical performance of k-nearest neighbors, LOESS with local linear models, and random forests on the popular Friedman1 simulation from Friedman [37]. In this simulation, $p = 10$, $\mathbf{x}$ is distributed uniformly on the hypercube in 10 dimensions, and $y = 10 \sin(\pi x_1 x_2) + 20 \left(x_3 - \frac{1}{2}\right)^2 + 10 x_4 + 5 x_5 + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 9)$. Note that only 5 of the 10 dimensions of $\mathbf{x}$ are relevant to y. We see that random forests outperform LOESS and nearest neighbor methods; in this case the tuning parameters of LOESS and k-NN have been optimized on the holdout set, while the tuning parameters of random forests have been set to their default. Unlike random forests, the Euclidean distance based methods to not adapt to the fact that irrelevant predictors are included in the model, and the local neighborhoods for each test point do not 'spread out' in the flat directions. We explore this property of random forests in the next section.

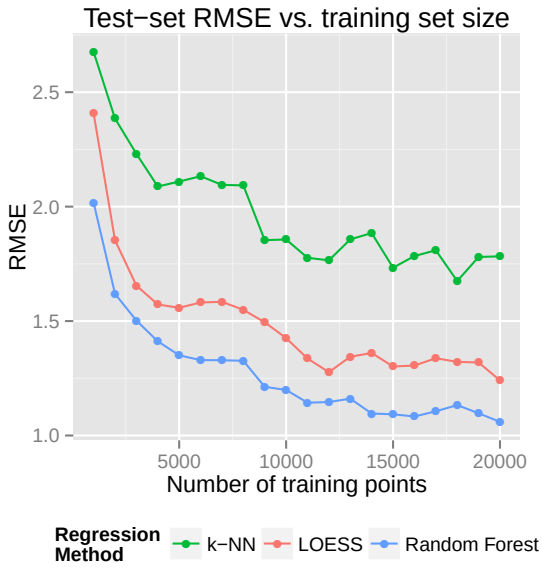

FIGURE 4.1: **Comparison of nonparametric techniques on the Friedman1 simulation.** Parameters of LOESS and k-NN were tuned on the holdout set, while random forest parameters were set at default values (3 candidate variables at each node, trees trained to 5 points per leaf.

Random forests as adaptive nearest neighbors

Soon after the introduction of the random forest algorithm, Lin and Jeon [55] observed that random forests can be considered an adaptive potential nearest neighbor method (PNN). To explain this idea, we introduce some notation. Given a random forest consisting of K trees, we define θ to be the random parameter vector that determines the growth of a tree (for example, θ specifies the sampling of training points and the selection of covariates at each node). The tree built with θ is denoted as $T(\theta)$, and for $\mathbf{x} \in \mathbb{R}^p$, let $R(\mathbf{x}, \theta)$ be the rectangle corresponding to the terminal node of $T(\theta)$ containing $\mathbf{x}$. We define the connection function of a tree, which is the indicator that two points share the same terminal node of a particular tree.

$$w(\mathbf{x}_1, \mathbf{x}_2, \theta) = \mathbb{1}\left\{\mathbf{x}_1 \in R(\mathbf{x}_2, \theta)\right\} \tag{4.4}$$

Define the number of training points contained in a leaf-node containing the point $\mathbf{x}$, for a tree trained with parameter θ as $k_\theta(\mathbf{x}) = \sum_{i=1}^n w(\mathbf{x}_i, \mathbf{x}, \theta)$. Then the prediction of a random forest can be written as

$$\hat{g}(\mathbf{x}) = \frac{1}{K}\sum_{j=1}^{K}\left[\frac{\sum_{i=1}^n w(\mathbf{x}_i, \mathbf{x}, \theta_j) y_i}{k_{\theta_j}(\mathbf{x})}\right] \tag{4.5}$$

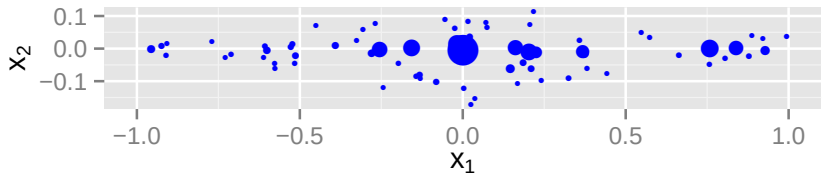

FIGURE 4.2: **Weights of a random forest.** Each circle represents a point in the training set, and diameter of the circle represents the weight of that point when making a prediction at the origin. The function being fit does not vary along the x_1 coordinate.

Note that this takes the form of eqs. (4.1) and (4.2), where $\mathcal{F}$ is the family of constant functions, and we define the random forest weighting function

$$w_{\text{RF}}(\mathbf{x}_i, \mathbf{x}) = \frac{1}{K} \sum_{j=1}^{K} \left[\frac{w(\mathbf{x}_i, \mathbf{x}, \theta_j)}{k_{\theta_j}(\mathbf{x})} \right] \tag{4.6}$$

Note that this weighting function is not derived from any explicit distance metric. However, as noted in Lin and Jeon [55], assuming the training procedure trains decision trees with at most k training points per leaf node, and the full training set is used for each tree, then any point $\mathbf{x}_i$ with positive weight is in fact a k-nearest neighbor under some monotone distance metric. Moreover, due to the training procedure of each constituent regression tree, this function assigns high weight to pairs of points that share a similar value of the response, y. Hence, we can interpret it as providing an adaptive distance measure, where distances are shortened in directions where the function $E(y \mid \mathbf{x})$ is flat. This is illustrated in fig. 4.2, where we plot the values of the weighting function for a random forest fit on a function of two variables that does not vary in the first coordinate, when making a prediction at the origin $(0, 0)$. Note that the weights taper off much more quickly along the x_2 dimension, and that the random forest 'borrows strength' from points that have very different values of $\mathbf{x}_1$.

4.3 Supervised neighborhoods

The prior discussion sets up the introduction of our method for nonparametric regression. The first step is to use the standard random forest training algorithm to fit a global random forest to the training data $\{(y_i, \mathbf{x}_i)\}$. This yields a random forest weighting function $w_{\text{RF}}(\cdot, \cdot)$; note that, given the training set, this is still a random function, as it depends on the underlying tree-training parameters $\theta_1, \ldots, \theta_K$. We then use this weighting function in eq. (4.1), but unlike random forests, we expand the function class $\mathcal{F}$ to a broader, more flexible class than just constants. In our experiments and theoretical analysis, we use the set of linear functions with an intercept term:

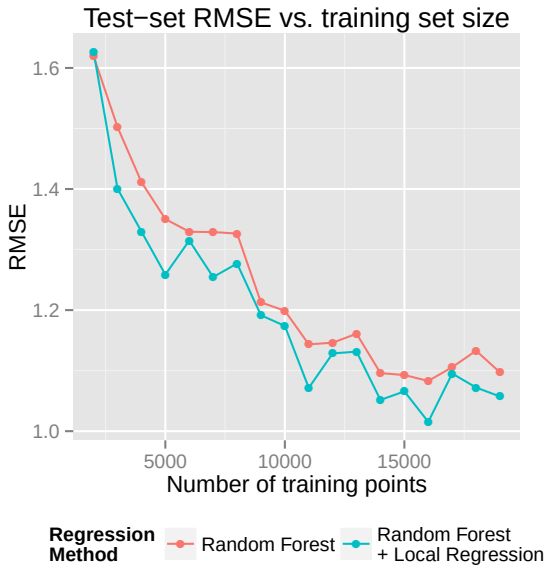

FIGURE 4.1: **Test-set RMSE of standard random forest vs. random forest combined with local linear modeling.**

$$\mathcal{F} = \left\{ f \text{ s.t. } f(\mathbf{x}) = \alpha + \boldsymbol{\beta}^T \mathbf{x}; \alpha \in \mathbb{R}, \boldsymbol{\beta} \in \mathbb{R}^p \right\} . \tag{4.7}$$

For convenience, define $w_i = w_{\text{RF}}(\mathbf{x}_i, \mathbf{x})$, and let $\mathbf{U}(\mathbf{x}) \in \mathbb{R}^{p+1} = (1, x_1, \ldots, x_p)^T$, i.e. $\mathbf{U}(\mathbf{x})$ is the vector $\mathbf{x}$ prefixed by 1. Substituting into eqs. (4.1) and (4.2), to make a prediction at a point $\mathbf{x} \in \mathbb{R}^p$, our method can be written in the following form:

$$\boldsymbol{\beta}_{\mathbf{x}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{p+1}}{\arg\min} \sum_{i=1}^{n} w_i \left(y_i - \boldsymbol{\beta}^T \mathbf{U}(\mathbf{x}_i - \mathbf{x}) \right)^2 \tag{4.8}$$

$$\hat{g}(\mathbf{x}) = \boldsymbol{\beta}_{\mathbf{x}}^T \mathbf{U}(\mathbf{0}) . \tag{4.9}$$

Equation (4.8) is standard weighted linear regression, re-centered at $\mathbf{x}$, and the prediction at point $\mathbf{x}$ is the resulting intercept term (which motivates evaluating $\boldsymbol{\beta}_{\mathbf{x}}$ at $\mathbf{U}(\mathbf{0})$). We note that this method can, in fact, be written as a local averaging method, but where the original weights from the random forest are transformed [92]. Define

$$\boldsymbol{\Sigma}_{\mathbf{x}} = \sum_{i=1}^{n} w_i \mathbf{U}(\mathbf{x}_i - \mathbf{x}) \mathbf{U}(\mathbf{x}_i - \mathbf{x})^T \tag{4.10}$$

$$\mathbf{a}_{\mathbf{x}} = \sum_{i=1}^{n} \mathbf{U}(\mathbf{x}_i - \mathbf{x}) w_i y_i \tag{4.11}$$

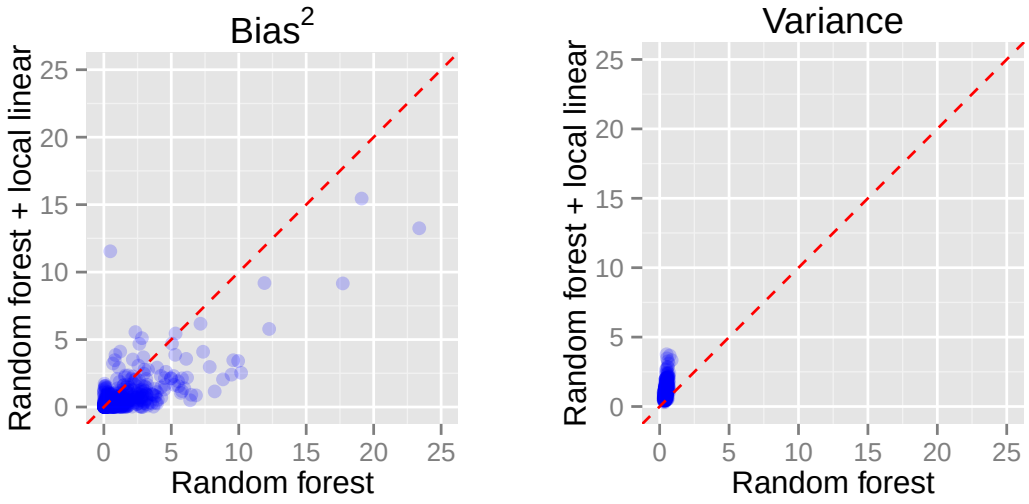

FIGURE 4.2: **Empirical bias and variance of standard random forest vs. random forest + local linear regression.** Both were trained on the same prediction task as fig. 4.1, with 5000 training points.

Then, if we assume that $\mathbf{\Sigma_x}$ is invertible,

$$\begin{aligned} \hat{g}(\mathbf{x}) &= \mathbf{U(0)}^T \mathbf{\Sigma_x^{-1}} \mathbf{a_x} \\ &= \frac{1}{n} \sum_{i=1}^{n} \mathbf{U(0)}^T \mathbf{\Sigma_x^{-1}} \mathbf{U}(\mathbf{x}_i - \mathbf{x}) w_i y_i \end{aligned} \tag{4.12}$$

Thus, local modeling transforms the weights as $w_i \to W_i := \mathbf{U(0)}^T \mathbf{\Sigma_x^{-1}} \mathbf{U}(\mathbf{x}_i - \mathbf{x}) w_i$. Note that the weight update is unsupervised (y is not incorporated), and has the effect of correcting local imbalances in the design; for an empirical investigation of this property, see, for example, Hastie and Loader [42]. It is a well-known feature of local linear regression that it reproduces linear functions exactly; see proposition 1.12 of Tsybakov [92]. Hence the transformed weights balance the design about the point $\mathbf{x}$; we have that:

$$\frac{1}{n} \sum_{i=1}^{n} \mathbf{U(0)}^T \mathbf{\Sigma_x^{-1}} \mathbf{U}(\mathbf{x}_i - \mathbf{x}) w_i \mathbf{U}(\mathbf{x}_i - \mathbf{x})^T = \mathbf{0}$$

Note that the term $\mathbf{U(0)}^T \Sigma_{\mathbf{x}}^{-1}$ extracts the first row of $\Sigma_{\mathbf{x}}^{-1}$; the elements of this row are related to the difference between $\mathbf{x}$ and the weighted average of the $\mathbf{x}_i$; we explictly calculate this vector in the proof of Theorem 4.

Figure 4.1 compares the predictive root mean-square-error (RMSE) of standard random forests with local-regression augmented random forests, i.e., SILO, when training on the Friedman1 simulated dataset, for varying sizes of training sets. We see that the additional

local linear modeling step improves the predictive RMSE. In fig. 4.2 we calculate an empirical bias-variance trade-off for predictions of each method. To calculate these estimates of bias and variance, we fit the models 500 times, fixing all the training and test set data except the values of ϵ_i in the training data (i.e. this is bias and variance conditional on the design). We then estimate the bias by calculating the difference between the mean prediction over the 500 models and the true expected outcome, and the variance is estimated by calculating the variance of the predictions over the 500 models. We see that local modeling reduces the bias of the original random forest at the expense of a slight increase in the variance; overall the RMSE is diminished.

Consistency of random forests + local regression

Consistency properties of random forests have been extensively studied since the original results in Breiman [15]. The true random forest algorithm is notoriously difficult to analyze, due to the many complex ingredients (greedy partitioning scheme, bagging, and deeply trained trees), so most analyses make simplifications to make the analysis more tractable. Local polynomial methods based on Euclidean distance, on the other hand, are very well understood theoretically: they are known to have minimax-optimal rates of convergence [88], can correct bias problems at the boundary of the data [31], and that, with slight modifications to avoid pathological behavior, they are known to be consistent over arbitrary joint distributions of y and $\mathbf{x}$ with $Ey^2 < \infty$ [50].

In our analyses, we make several simplifications to the true random forest algorithm. Most notably, our analysis relies on the assumption that the procedure used to build the regression trees proceeds independently of the data used in building the local models; this assumption is also made in Biau [12] and is similar to the 'honest tree' assumption defined in Wager and Athey [96]. The more detailed mathematical analysis in Scornet et al. [82] proves consistency of random forests for additive regression functions while avoiding this assumption. In our results, we refer the 'training data' as the dataset $\{(y_i, \mathbf{x}_i)\}$ which is used to fit local models, but is not used to determine the structure of the trees. Also, we assume that these data are sampled without replacement in each of the trees of the forest, to avoid difficulties in analyzing sampling with replacement. More specifically, we require the following:

Assumption 1. The training data $\{(y_i, \mathbf{x}_i), i = 1, \ldots, n\}$ are generated i.i.d. from a joint distribution that satisfies the following properties:

$$\mathbf{x}_i \sim \text{Uniform}([0,1]^p) \tag{4.13}$$

$$y_i = g(\mathbf{x}_i) + \omega(\mathbf{x}_i)\epsilon_i \tag{4.14}$$

where ϵ_i is independent of $\mathbf{x}_i$, the function $\omega(\mathbf{x})$ is bounded, $E(\epsilon_i) = 0$, and $E(\epsilon_i^2) < \infty$. The function g must be sufficiently well-behaved such that assumption 4 can be satisfied. A minimal requirement is that g is continuous.

Assumption 2. The splits of the constituent regression trees of the random forest are calculated using a dataset that is independent of the training data.

Assumption 3. We require that

$$\min_{\substack{\mathbf{x}\in[0,1]^p \\ j\in(1,\ldots,K)}} k_{\theta_j}(\mathbf{x}) \to \infty \tag{4.15}$$

i.e., the number of training points contained in each node of each tree of the random forest goes to infinity.

Assumption 4. For each $\mathbf{x} \in [0,1]^p$, the trees are trained in such a way that

$$\max_{i,j} \left[w(\mathbf{x}_i, \mathbf{x}, \theta_j) |g(\mathbf{x}_i) - g(\mathbf{x})| \right] \xrightarrow{p} 0 \tag{4.16}$$

i.e., that the cells containing $\mathbf{x}$ shrink in such away that the maximal variation of the function g within a cell shrinks to 0 in probability.

Assumption 5. The data in each tree are sampled without replacement from the original training set, so that all training points occuring in a particular leaf node have the same weight.

Theorem 4. *Under assumptions 1–5, for all* $\mathbf{x} \in [0,1]^p$,

$$\hat{g}(\mathbf{x}) - g(\mathbf{x}) \xrightarrow{p} 0 \tag{4.17}$$

The proof is provided in section 4.A.

Remark 11. Assumption 4 allows us to ensure that the approximation error at point $\mathbf{x}$ vanishes asymptotically. This can be shown to hold, for example, by combining Lemma 2 of Meinshausen [63] with continuity of g; however, this requires additional assumptions on the tree-training procedure that deviate somewhat from the usual random forest tree-training procedure. Proposition 2 in Scornet et al. [82] proves a slightly weaker version of this assumption: the pointwise statement is not proven, but a similar statement is shown to hold for a random draw of $\mathbf{x}$. It is an open question whether such pointwise control over the leaves can hold for the true random forest tree-training procedure.

Algorithm 2: Distributed Nonparametric Regression via SILO

Input: Number of workers W; Distributed training dataset $\{(y_{i,j}, \mathbf{x}_{i,j}); j = 1, \ldots, W; i = 1, \ldots, N_j\}$ where $\sum_j^W N_j = n$; Test points $(\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_M)$ stored on master node

for *each worker* $j = 1 \to W$ **do in parallel**
  Fit random forest on local data $(y_{i,j}, \mathbf{x}_{i,j})$
for *each* $m = 1 \to M$ **do**
  Master broadcasts test point $\tilde{\mathbf{x}}_m$
  for *each worker* $j = 1 \to W$ **do in parallel**
    Worker j calculates supervised neighborhood of $\tilde{\mathbf{x}}_m$, yielding $\{w_{i,j}, (y_{i,j}, \mathbf{x}_{i,j})\}$
    Worker j calculates weighted covariance matrix and cross-covariance vector:

$$\Sigma_{j,m} = \sum_i^{N_j} w_{j,i} U \left(\mathbf{x}_{j,i} - \tilde{\mathbf{x}}_m\right) U \left(\mathbf{x}_{j,i} - \tilde{\mathbf{x}}_m\right)^T \qquad \mathbf{a}_{j,m} = \sum_i^{N_j} w_{j,i} U \left(\mathbf{x}_{j,i} - \tilde{\mathbf{x}}_m\right) y_{j,i}$$

    Worker j communicates $(\Sigma_{j,m}, \mathbf{a}_{j,m})$ to the master
  Master node solves linear system and calculates
  $\hat{g}\left(\tilde{\mathbf{x}}_m\right) = U\left(\mathbf{0}\right)^T \left[\sum_{j=1}^W \Sigma_{j,m}\right]^{-1} \sum_{j=1}^W a_{j,m}$
return $\left(\hat{g}\left(\tilde{\mathbf{x}}_1\right), \ldots, \hat{g}\left(\tilde{\mathbf{x}}_M\right)\right)$

4.4 Distributed Silo

There is a rich literature on algorithms for recovering nearest neighbors from large datasets in high dimension, leading to efficient implementations of k-nearest neighbors in the distributed setting [10, 48, 1]. The possibility of using supervised neighborhoods in a distributed setting remains less explored. We propose an approach for distributed nonparametric estimation which is based on the random forest-supervised neighborhood method introduced above. In the setting of distributed data, communication latency is high, and implementing the random forest algorithm on a distributed dataset is prohibitively expensive, due to the necessity of performing several distributed sorts at each internal node in the CART training procedure. However, approximations based on binning data, extracting order statistics, and using randomized searches for split-points has led to several scalable implementations of distributed random forests [76, 65].

Here, we take a different approach, which avoids communication at training time. We extend SILO to the distributed setting, but instead of attempting to train a global random forest, we train random forests separately on each worker. At test-time, supervised neighborhoods of each test point are determined independently by each worker node, using the usual random forest procedure of finding weighted PNNs. Then the master node gathers the workers' supervised neighborhoods and fits a local linear model. The local model helps to

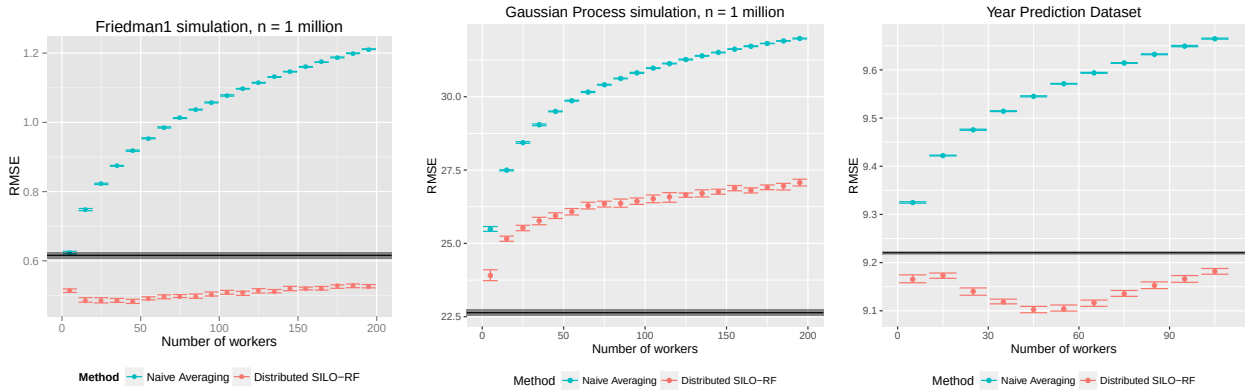

FIGURE 4.1: **Effect of distributing the data for the Friedman1 and Gaussian Process simulations, and the Year Prediction dataset.** The black bar represents a random forest trained on the full dataset. The error bars represent the ranges between the 0.1 - 0.9 quantiles after performing 40 different runs of the experiments, fixing the training and test data data, but fitting different random forests. In the case of the simulations, the test datasets are noise-free, so the true regression function would have an RMSE of zero.

lower the bias of the workers' individual predictions; this is similar in spirit to the approach taken in Zhang et al. [104], which uses bootstrap bias correction. Our method is presented in more detail in algorithm 2. Note that, instead of communicating the training data itself, the workers communicate sufficient statistics for performing local regression - the weighted covariance matrix $\Sigma_{\mathbf{x}}$ and the weighted cross covariance vector $a_{\mathbf{x}}$; the master then must simply add together these results from the workers and solve a linear system. Additionally, to amortize latency costs, we batch the communication at test-time; the master can broadcast a set of several test-points in a single message, and the workers can communicate a corresponding set of sufficient statistics. In practice, this significantly speeds up test-time computations.

Simulation and Real Data Experiments

We analyze the performance of algorithm 2 in two ways. First, we plot the out-of-sample predictive accuracy of distributed estimation as we increase the number of worker nodes but fix the size of the overall dataset. Second, we plot the accuracy as the size of the dataset is fixed per worker, but the number of workers is varied. We compare the performance of our algorithm to both naive averaging of divide-and-conquer random forests (equivalently, local constant models in algorithm 2), and, in the second case, the global distributed random forest implementation in MLlib.

Fixing the training set size

We demonstrate the effect of parallelization on two simulated datasets and one real dataset. The first simulation is the Friedman1 function described in section 4.2. The second simulation is a Gaussian process, generated by fixing M different vectors in $\mathbb{R}^p$, drawing $Z_1, \ldots, Z_M$ i.i.d. $\mathcal{N}(0, 1)$, setting $\sigma^2 = 0.05$, and generating the function $g(\mathbf{x}) = \sum_{k=1}^{M} Z_k e^{-\frac{1}{2}\|\boldsymbol{\omega}_k\|^2\sigma^2} \cos(\boldsymbol{\omega}_k^T \mathbf{x})$. We generate a function in $p = 10$ variables, but for our simulation, we append an additional 20 variables unrelated to the outcome. This prediction task is very difficult - even with 1 million training points, a random forest attains a test-set correlation of 0.83. The real dataset is a prediction task drawn from the Million Song Dataset [11]. The task is to predict the year of a song's release given 90 acoustic features of the song. The training set consists of $463,715$ songs appearing between 1922 and 2011, and the test set consists of 51,630 songs. The results are shown in fig. 4.1. The plots show the performance of a random forest trained on the full datasets (black lines), a 'naive'-divide and conquer random forest, which simply averages predictions from random forests trained on the workers (blue points), and the distributed SILO procedure outlined in algorithm 2. The number of workers ranges from 5 to 195 for the two simulations (corresponding to 200k to 5k training points per worker), and from 5 to 105 for the Year Prediction task (corresponding to approximately 92k to 4k training points per worker). The error bars in the plots represent 0.1-0.9 quantile ranges, as the experiments were repeated 40 times, holding the training and test set fixed, while fitting different random realizations of random forests. We see that, in the case of the Friedman1 simulation, the Distributed SILO procedure consistently outperforms the full random forest and, in contrast to the naively distributed random forest, shows little decay in performance as the data are distributed. In the Gaussian process simulation, the performance of Distributed SILO does deteriorate as the data are distributed, and we can see an increase in variance, but it still significantly outperforms the naively distributed random forest. For the Year Prediction task, Distributed SILO significantly improves upon the full random forest for all numbers of workers.

Fixing the dataset size per worker

We now explore the performance of distributed-SILO when we fix the training dataset size per worker, but increase the number of workers, and hence, the overall amount of training data. We use the two simulation setups outlined above: the Friedman1 function, and the higher-dimensional Gaussian process. We slightly altered the Friedman1 simulation, adding an additional 45 noise features to increase the overall size of the training dataset and to make the random forest fitting procedure more computationally challenging, as the `mtry` parameter (number of variables considered at each node) is increased from 3 to 18. We implemented distributed-SILO in Spark, a popular open source framework for distributed computation, and we compare our method with the implementation of distributed random forests in MLlib. To attain similar accuracy between SILO and MLlib's random forests, we set the `maxDepth` parameter of MLlib to be 15 and the `minInstancesPerNode` parameter to be 10. We ran our

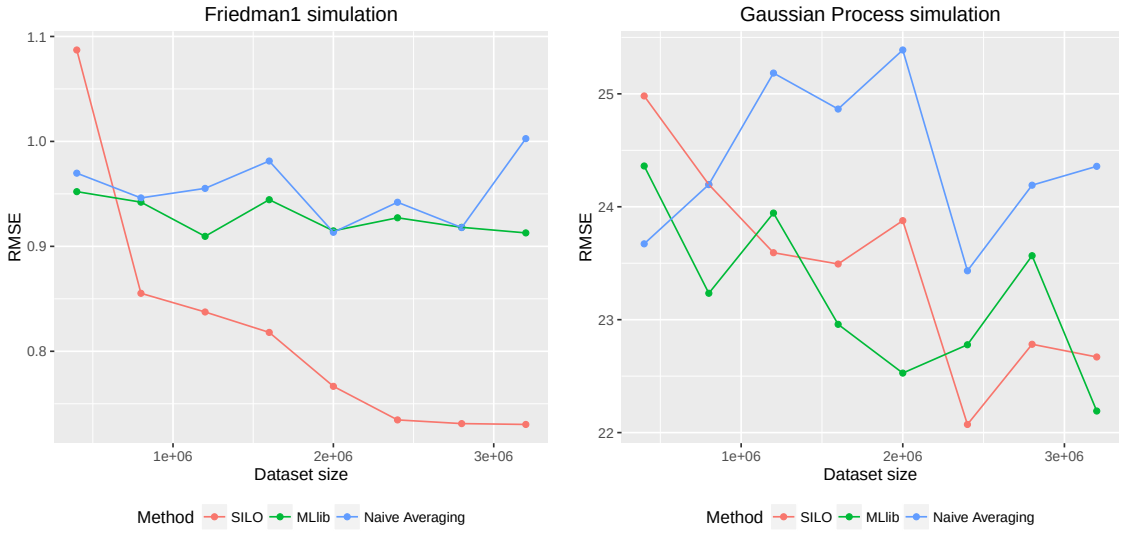

FIGURE 4.2: **Performance of Silo, MLlib's random forest, and naively distributed random forests on simulated datasets with growing training set size.** We fixed the amount of training data per partition to 100,000 observations, and varied the number of partitions from 4 to 32. The y-axis shows the test-set RMSE. In the Friedman1 simulation, SILO consistently fits better models with more training data. In the Gaussian process simulation, both SILO and MLlib generally improve with more training data. For most cluster sizes, SILO improves upon naive averaging of workers' predictions. For large cluster sizes, it is more than 2× faster than MLlib due to its avoidance of communication; see fig. 4.3 for details.

experiments on Amazon EC2, using r3.xlarge instances, which have 4 processors and 30.5 GB of RAM per node. We set the number of Spark partitions to equal the number of processors, testing clusters of size 2 to 8 nodes. The training dataset size was fixed at n = 100,000 per partition, yielding n ranging from 800k to 3.2 million. The results of our experiments are shown in fig. 4.2. In these plots we show three different methods: Distributed-SILO, naive divide-and-conquer random forests, and MLlib's distributed random forest. Similar to the results in fig. 4.1, the local modeling step in SILO improves performance relative to the naive averaging method for most cluster sizes. The performance of Distributed-SILO is particularly strong in the Friedman1 simulation, as the predictions consistently improve with more training data, unlike MLlib. In the Gaussian process simulation, both Distributed-SILO and MLlib generally improve with more data, though the trend is not monotone. Training time of Distributed-SILO is essentially constant because it avoids communication entirely. In both simulations, it is more than 2× faster than MLlib for the largest clusters (see fig. 4.3).

We note that distributed-SILO has traded training time for test time, as communication of supervised neighborhoods between workers and master must occur, and a local model must be fit for every test point. However, we find that the increase in test time is negligible

compared to the gains in training time; for example, by batching test points into groups of size 100, we are able to amortize the cost of communication latency, and find that the predictions can be made at an overall rate of approximately 20 milliseconds per test point. While this is several orders of magnitude slower than making predictions using a model that is stored locally (as is the case with MLlib), it is still fast enough that overall gains in training time are dominant unless very large test sets are required. We also note that, in the standard random forest algorithm, the trees are built such that the leaves contain a small number of test points (usually less than 10 for regression tasks). Thus, the overall storage costs of random forests scales linearly with the size of the training set, and the models may grow beyond the in-memory storage capacity of a single machine; for example, we estimate that, in our Scala implementation, a random forest trained on a dataset of size 3.2 million points would require more than 6 gigabytes of memory.

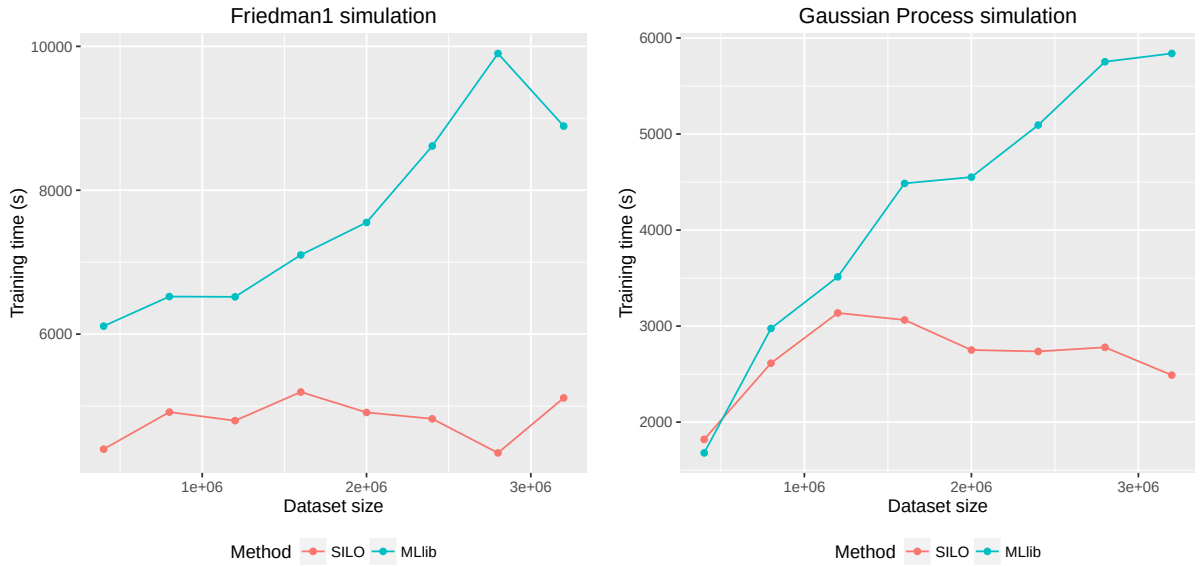

FIGURE 4.3: **Training timing for Distributed-Silo and MLlib with growing dataset size.** The amount of training data per Spark partition was fixed at 100,000 observations. Experiments were run on Amazon EC2 clusters using r3.xlarge instances, which have 4 processors and 30.5 GB of RAM per node. For each experiment, the cluster size was chosen such that the number of partitions was equal to the number of processors. As expected, training time of Distributed SILO is fairly constant as size of the dataset is increased, due to the lack of communication between workers. While MLlib's implementation avoids communication, particularly at deeper nodes in the trees, it does pay some communication penalty as more workers are added.

4.5 Conclusion

SILO is a novel local learning algorithm that uses random forests to identify supervised neighborhoods for the problem of non-parametric regression. We proved the consistency of SILO, introduced a distributed variant, and demonstrated its favorable empirical performance (in terms of accuracy and computation) relative to natural baselines.

We note that the contemporaneous work of Xu et al. [99] introduces a local learning method that also relies on random forests to identify supervised neighborhoods for non-parametric regression. They also introduce a reweighting procedure for the local models that, in contrast to ours, is supervised using a small scale local random forest. This work focuses on empirical studies and does not investigate the scalability of the proposed algorithm. However, in followup work, they show that the underlying ideas motivating the distributed variant of SILO are applicable to their approach as well [105].

Moving forward, it would be interesting to extend SILO to the classification setting, study the degree to which SILO can be parallelized by characterizing the relationship between n, N_j and W in algorithm 2, and investigate the theoretical performance of local methods with supervised neighborhoods relative to classical non-adaptive methods under sparsity assumptions, e.g., when the response only involves $s \ll p$ predictors.

Appendix

4.A Proof of Theorem 4

Proof. For convenience, we introduce a shorthand for the random forest weight of training point i, defining $w_i = w_{\mathrm{RF}}(\mathbf{x}_i, \mathbf{x})$. We substitute eq. (4.14) into the definition of $\hat{g}(\mathbf{x})$ in eq. (4.12).

$$\hat{g}\left(\mathbf{x}\right) - g\left(\mathbf{x}\right) = \mathbf{U}\left(\mathbf{0}\right)^T \Sigma_{\mathbf{x}}^{-1} \left[\sum_{i=1}^{n} w_i \mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right)\left(g\left(\mathbf{x}_i\right) - g\left(\mathbf{x}\right) + \omega(\mathbf{x})\epsilon_i\right)\right] \tag{4.18}$$

Note that we have used the fact that

$$\mathbf{U}\left(\mathbf{0}\right)^T \Sigma_{\mathbf{x}}^{-1} \left[\sum_{i=1}^{n} w_i \mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right) g\left(\mathbf{x}\right)\right] = g(\mathbf{x}) \tag{4.19}$$

This is a well-known property of local linear estimators: that they reproduce linear functions (in this case, a constant). See, for example, Proposition 1.12 in Tsybakov [92].

We decompose $\hat{g}(\mathbf{x}) - g(\mathbf{x})$ into a bias-type term and a variance-type term, defining

$$b\left(\mathbf{x}\right) = \mathbf{U}\left(\mathbf{0}\right)^T \Sigma_{\mathbf{x}}^{-1} \left[\sum_{i=1}^{n} w_i \mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right)\left(g\left(\mathbf{x}_i\right) - g\left(\mathbf{x}\right)\right)\right] \tag{4.20}$$

$$v\left(\mathbf{x}\right) = \mathbf{U}\left(\mathbf{0}\right)^T \Sigma_{\mathbf{x}}^{-1} \left[\sum_{i=1}^{n} w_i \mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right)\omega(\mathbf{x}_i)\epsilon_i\right] \tag{4.21}$$

We will show that each of these terms converges to 0 in probability. □

Lemma 8. $v(\mathbf{x}) \xrightarrow{p} 0$

Proof. For notational convenience, we introduce a shorthand for the indicator that a training point $\mathbf{x}_i$ belongs to the same leaf node as $\mathbf{x}$ in a tree trained with random parameter θ: let $w_i(\theta) = w(\mathbf{x}_i, \mathbf{x}, \theta)$. We drop the dependence on $\mathbf{x}$ in the notation, because we will work with a fixed $\mathbf{x}$ for the duration of the proof. By assumption 5, a particular training point will only occur once in a leaf node, so $w_i(\theta) \in \{0, 1\}$, and it both indicates the presence of

i in the leaf node, and can be used to represent the weight of training point x_i. We define the bandwidth matrix of the random forest to be a diagonal matrix with diagonal elements set to be the largest component-wise distances from $\mathbf{x}$ to a training point that has nonzero weight. Let $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,p})$, and let $\mathbf{x} = (x_1, \ldots, x_p)$. We define

$$\begin{aligned} h_m &= \max_{i,j} \left[w_i(\theta_j) \left| x_{i,m} - x_m \right| \right] \\ \mathbf{H} &= \text{diag}(1, h_1, \ldots, h_p) \end{aligned} \tag{4.22}$$

By assumption 3, the number of training points falling in a leaf node goes to infinity. Using assumption 2, if we condition on the variables $w_i(\theta_j)$, the subset of the training data falling in $R(\mathbf{x}, \theta_j)$ is independent and identically distributed uniformly in the rectangle $R(\mathbf{x}, \theta_j)$. By definition, $\left\| \mathbf{H}^{-1}\mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right) \right\|_\infty < 1$, and we have assumed that $\omega(\mathbf{x})$ is bounded and $E(\epsilon_i) = 0$. We apply the weak law of large numbers (in fact, just a variance calculation) to obtain

$$\frac{1}{k_{\theta_j}(\mathbf{x})} \sum_{i=1}^{n} w_i\left(\theta_j\right) \mathbf{H}^{-1}\mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right) \omega(\mathbf{x}_i)\epsilon_i \xrightarrow{p} \mathbf{0} \tag{4.23}$$

Thus, averaging over trees, we have

$$\sum_{i=1}^{n} w_i \mathbf{H}^{-1}\mathbf{U}\left(\mathbf{x}_i - \mathbf{x}\right) \omega(\mathbf{x}_i)\epsilon_i \xrightarrow{p} \mathbf{0} \tag{4.24}$$

We now examine the covariance matrix $\Sigma_{\mathbf{x}}$.

$$\begin{aligned} \Sigma_{\mathbf{x}} &= \sum_{i=1}^{n} w_i \mathbf{U}(\mathbf{x}_i - \mathbf{x})\mathbf{U}(\mathbf{x}_i - \mathbf{x})^T \\ &= \frac{1}{K} \sum_{j=1}^{K} \left[\frac{1}{k_{\theta_j}(\mathbf{x})} \sum_{i=1}^{n} w_i(\theta_j)\mathbf{U}(\mathbf{x}_i - \mathbf{x})\mathbf{U}(\mathbf{x}_i - \mathbf{x})^T \right] \end{aligned}$$

We define the tree-level contributions to $\Sigma_{\mathbf{x}}$ as

$$\Sigma_{\mathbf{x}}(\theta_j) = \frac{1}{k_{\theta_j}(\mathbf{x})} \sum_{i=1}^{n} w_i(\theta_j)\mathbf{U}(\mathbf{x}_i - \mathbf{x})\mathbf{U}(\mathbf{x}_i - \mathbf{x})^T$$

We define $\boldsymbol{\delta}_i = w_i(\theta_j)(\mathbf{x}_i - \mathbf{x})$, and denote the components of $\boldsymbol{\delta}_i$ as $\boldsymbol{\delta}_i = (\delta_{i,1}, \ldots, \delta_{i,p})$. For convenience, we have dropped the dependence on j in the notation for $\boldsymbol{\delta}_i$, but it is to be understood that it is only nonzero for data falling in the leaf node of tree j. Then

$$\Sigma_{\mathbf{x}}\left(\theta_j\right) = \frac{1}{k_{\theta_j}\left(\mathbf{x}\right)} \begin{pmatrix} k_{\theta_j}(\mathbf{x}) & \sum_i \delta_{i,1} & \cdots & \sum_i \delta_{i,p} \\ \sum_i \delta_{i,1} & \sum_i \delta_{i,1}^2 & \cdots & \sum_i \delta_{i,1}\delta_{i,p} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_i \delta_{i,p} & \sum_i \delta_{i,p}\delta_{i,1} & \cdots & \sum_i \delta_{i,p}^2 \end{pmatrix} \tag{4.25}$$

We define the width of the rectangle $R(\mathbf{x}, \theta_j)$ in each dimension as $\omega_{m,j}$, and define the offsets of $\mathbf{x}$ from the center of the rectangle $R(\mathbf{x}, \theta_j)$ in each dimension as $\Delta_{m,j}$, $m \in \{1, \ldots, p\}, j \in \{1, \ldots, K\}$. Then we have, by the weak law of large numbers, assumptions 2 and 3, and the i.i.d. uniform distribution of $\mathbf{x}_i$,

$$\frac{1}{k_{\theta_j}(\mathbf{x})} \sum_i \delta_{i,m} + \Delta_{m,j} \xrightarrow{p} 0 \tag{4.26}$$

$$\frac{1}{k_{\theta_j}(\mathbf{x})} \sum_{i=1}^{n} \delta_{i,l}\delta_{i,m} - \Delta_{l,j}\Delta_{m,j} \xrightarrow{p} 0 \text{ for } l \neq m \tag{4.27}$$

$$\frac{1}{k_{\theta_j}(\mathbf{x})} \sum_i \delta_{i,m}^2 - \frac{1}{3}\left(3\Delta_{m,j}^2 + \frac{\omega_{m,j}^2}{4}\right) \xrightarrow{p} 0 \tag{4.28}$$

We define the tree-averaged quantities

$$\Delta_m = \frac{1}{K}\sum_{j=1}^{K} \Delta_{m,j} \tag{4.29}$$

$$\sigma_m^2 = \frac{1}{K}\sum_{j=1}^{K} \frac{1}{3}\left(3\Delta_{m,j}^2 + \frac{\omega_{m,j}^2}{4}\right) \tag{4.30}$$

We define the vector $\mathbf{\Delta} = (\Delta_1, \ldots, \Delta_p)^T$, and define the matrix

$$S = \begin{pmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_p^2 \end{pmatrix} + \Delta\Delta^T \tag{4.31}$$

The above has shown that

$$\Sigma_{\mathbf{x}} - \begin{pmatrix} 1 & \Delta^T \\ \Delta & S \end{pmatrix} = o_p(1) \tag{4.32}$$

Then, we will apply the Woodbury formula and the formula for the inverse of a block-partitioned matrix to explicitly calculate the inverse of this matrix. We define

$$\eta = \sum_{m=1}^{p} \frac{\Delta_m^2}{\sigma_m^2} - \frac{1}{1 + \sum_{j=1}^{p} \frac{\Delta_i^2}{\sigma_i^2}} \sum_{m=1}^{p}\sum_{l=1}^{p} \frac{\Delta_m^2\Delta_l^2}{\sigma_m^2\sigma_l^2} \tag{4.33}$$

Then

$$\mathbf{U}(\mathbf{0})^T \begin{pmatrix} 1 & \Delta^T \\ \Delta & S \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{1-\eta} \\ \frac{1}{1-\eta}\left[\frac{\Delta_1}{\sigma_1^2} - \frac{1}{1+\sum_{j=1}^{p}\frac{\Delta_i^2}{\sigma_i^2}} \sum_{m=1}^{p} \frac{\Delta_1\Delta_m^2}{\sigma_1^2\sigma_m^2}\right] \\ \vdots \\ \frac{1}{1-\eta}\left[\frac{\Delta_p}{\sigma_p^2} - \frac{1}{1+\sum_{j=1}^{p}\frac{\Delta_i^2}{\sigma_i^2}} \sum_{m=1}^{p} \frac{\Delta_p\Delta_m^2}{\sigma_p^2\sigma_m^2}\right] \end{pmatrix} \tag{4.34}$$

Since $\Delta_m^2 = O(\sigma_m^2)$, we see that this vector is

$$\mathbf{U}(\mathbf{0})^T \bar{\Sigma} = \begin{pmatrix} O(1) \\ O\left(\frac{1}{\sigma_1}\right) \\ \vdots \\ O\left(\frac{1}{\sigma_p}\right) \end{pmatrix} \tag{4.35}$$

Hence, because $h_m = O(\sigma_m)$,

$$\mathbf{U}(\mathbf{0})^T \Sigma_{\mathbf{x}}^{-1} \mathbf{H} = (O_p(1), \ldots, O_p(1)) \tag{4.36}$$

Thus, we have that $\mathbf{U}(\mathbf{0})^T \Sigma_{\mathbf{x}}^{-1} \sum_{i=1}^n w_i \mathbf{U}(\mathbf{x}_i - \mathbf{x}) = o_p(1)$ □

Lemma 9. $b(\mathbf{x}) \xrightarrow{p} 0$

Proof. We have, by definition,

$$b(\mathbf{x}) = \mathbf{U}(\mathbf{0})^T \Sigma_{\mathbf{x}}^{-1} \mathbf{H} \left[\sum_{i=1}^n w_i \mathbf{H}^{-1} \mathbf{U}(\mathbf{x}_i - \mathbf{x}) (g(\mathbf{x}_i) - g(\mathbf{x})) \right] \tag{4.37}$$

where $\mathbf{H}$ was defined in eq. (4.22). By assumption 4, and the definition of $\mathbf{H}$, we have that

$$\left[\sum_{i=1}^n w_i \mathbf{H}^{-1} \mathbf{U}(\mathbf{x}_i - \mathbf{x}) (g(\mathbf{x}_i) - g(\mathbf{x})) \right] = o_p(1) \tag{4.38}$$

As we have shown in Lemma 8, $\mathbf{U}(\mathbf{0})^T \Sigma_{\mathbf{x}}^{-1} \mathbf{H} = (O_p(1), \ldots, O_p(1))$, hence $b(\mathbf{x}) = o_p(1)$ □

Bibliography

[1] A. Andoni and P. Indyk. "Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions". In: *Communications of the ACM* 51.1 (2008), pp. 117–122.

[2] F. R. Bach and M. I. Jordan. "Kernel Independent Component Analysis". In: *The Journal of Machine Learning Research* 3 (Mar. 2003), pp. 1–48.

[3] H. B. Barlow. "Possible principles underlying the transformations of sensory messages". In: *Sensory Communication*. Ed. by W. A. Rosenblith. Cambridge, MA: MIT Press, 1961, pp. 217–234.

[4] A. J. Bell and T. J. Sejnowski. "The 'independent components' of natural scenes are edge filters". In: *Vision Research* 37.23 (Dec. 1997), pp. 3327–3338.

[5] A. Belloni, V. Chernozhukov, I. Fernández-Val and C. Hansen. "Program Evaluation and Causal Inference with High-Dimensional Data". In: *ArXiv e-prints* (Nov. 2013). arXiv: 1311.2645 [math.ST].

[6] A. Belloni and V. Chernozhukov. "Least Squares After Model Selection in High-dimensional Sparse Models". In: *Bernoulli* 9.2 (2013), pp. 521–547.

[7] A. Belloni, V. Chernozhukov and C. Hansen. "Inference on Treatment Effects after Selection among High-Dimensional Controls". In: *The Review of Economic Studies* 81.2 (Nov. 2013), pp. 608–650.

[8] Y. Benjamini and Y. Hochberg. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 57.1 (1995), pp. 289–300.

[9] Y. Benjamini and D. Yekutieli. "The Control of the False Discovery Rate in Multiple Testing under Dependency". In: *The Annals of Statistics* 29.4 (2001), pp. 1165–1188.

[10] J. L. Bentley. "Multidimensional Binary Search Trees Used for Associative Searching". In: *Communications of the ACM* 18.9 (1975), pp. 509–517.

[11] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman and P. Lamere. "The Million Song Dataset". In: *Proceedings of the 12th International Society for Music Information Retrieval Conference*. University of Miami, 2011.

[12] G. Biau. "Analysis of a random forests model". In: *The Journal of Machine Learning Research* 13 (2012), pp. 1063–1095.

[13] R. T. Born and D. C. Bradley. "Structure and function of visual area MT". In: *Annual Review of Neuroscience* 28.1 (July 2005), pp. 157–189.

[14] Y. L. Boureau, F. Bach, Y. LeCun and J. Ponce. "Learning mid-level features for recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2010, pp. 2559–2566.

[15] L. Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[16] L. Breiman, J. Friedman, C. J. Stone and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[17] P. Bühlmann and S. van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Berlin, Heidelberg: Springer, 2011.

[18] M. Carandini, J. B. Demb, V. Mante, D. J. Tolhurst, Y. Dan, B. A. Olshausen, J. L. Gallant and N. C. Rust. "Do We Know What the Early Visual System Does?" In: *The Journal of Neuroscience* 25.46 (Nov. 2005), pp. 10577–10597.

[19] P. Chaudhuri, M.-C. Huang, W.-Y. Loh and R. Yao. "Piecewise-polynomial regression trees". In: *Statistica Sinica* 4.1 (1994), pp. 143–167.

[20] W. S. Cleveland and S. J. Devlin. "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting". In: *Journal of the American Statistical Association* 83.403 (1988), pp. 596–610.

[21] A. F. Connors, T. Speroff, N. V. Dawson, C. Thomas, F. E. Harrell, D. Wagner, N. Desbiens, L. Goldman, A. W. Wu, R. M. Califf et al. "The effectiveness of right heart catheterization in the initial care of critically III patients". In: *JAMA: The Journal of the American Medical Association* 276.11 (1996), pp. 889–897.

[22] D. R. Cox. "Further Results on Tests of Separate Families of Hypotheses". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 24.2 (1962), pp. 406–424.

[23] D. R. Cox. "Tests of Separate Families of Hypotheses". In: *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. Berkeley, Calif.: University of California Press, 1961, pp. 105–123.

[24] J. E. Dalen. "The Pulmonary Artery Catheter — Friend, Foe, or Accomplice?" In: *JAMA: The Journal of the American Medical Association* 286.3 (July 2001), p. 348.

[25] I. Daubechies, M. Defrise and C. De Mol. "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint". In: *Communications on Pure and Applied Mathematics* 57.11 (2004), pp. 1413–1457.

[26] S. V. David and J. L. Gallant. "Predicting neuronal responses during natural vision". In: *Network: Computation in Neural Systems* 16.2-3 (Jan. 2005), pp. 239–260.

[27] S. V. David, W. E. Vinje and J. L. Gallant. "Natural Stimulus Statistics Alter the Receptive Field Structure of V1 Neurons". In: *The Journal of Neuroscience* 24.31 (Aug. 2004), pp. 6991–7006.

[28] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Aurelio Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le and A. Y. Ng. "Large Scale Distributed Deep Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou and K. Q. Weinberger. Curran Associates, Inc., 2012, pp. 1223–1231.

[29] A. Diamond and J. S. Sekhon. "Genetic Matching for Estimating Causal Effects: A General Multivariate Matching Method for Achieving Balance in Observational Studies". In: *Review of Economics and Statistics* 95.3 (2013), pp. 932–945.

[30] B. Efron, T. Hastie, I. Johnstone and R. Tibshirani. "Least angle regression". In: *The Annals of Statistics* 32.2 (2004), pp. 407–499.

[31] J. Fan and I. Gijbels. "Variable Bandwidth and Local Linear Regression Smoothers". In: *The Annals of Statistics* 20.4 (1992), pp. 2008–2036.

[32] M. Fazel, H. Hindi and S. P. Boyd. "A rank minimization heuristic with application to minimum order system approximation". In: *Proceedings of the American Control Conference*. Vol. 6. 2001, pp. 4734–4739.

[33] D. J. Felleman and D. C. Van Essen. "Distributed hierarchical processing in the primate cerebral cortex". In: *Cerebral Cortex* 1.1 (1991), pp. 1–47.

[34] D. A. Freedman. "On regression adjustments in experiments with several treatments". In: *The Annals of Applied Statistics* 2.1 (2008), pp. 176–196.

[35] D. A. Freedman. "On regression adjustments to experimental data". In: *Advances in Applied Mathematics* 40.2 (Feb. 2008), pp. 180–193.

[36] D. A. Freedman. "Randomization does not justify logistic regression". In: *Statistical Science* 23.2 (2008), pp. 237–249.

[37] J. H. Friedman. "Multivariate Adaptive Regression Splines". In: *The Annals of Statistics* 19.1 (1991), pp. 1–67.

[38] K. Gregor and Y. LeCun. "Learning fast approximations of sparse coding". In: *Proceedings of the 27th International Conference on Machine Learning*. 2010, pp. 399–406.

[39] H. K. Hartline. "The response of single optic nerve fibers of the vertebrate eye to illumination of the retina." In: *American Journal of Physiology* 121 (1938), pp. 400–415.

[40] S. Harvey, D. A. Harrison, M. Singer, J. Ashcroft, C. M. Jones, D. Elbourne, W. Brampton, D. Williams, D. Young, K. Rowan et al. "Assessment of the clinical effectiveness of pulmonary artery catheters in management of patients in intensive care (PAC-Man): a randomised controlled trial". In: *Lancet* 366.9484 (2005), pp. 472–477.

[41] T. Hastie and R. Tibshirani. "Discriminant adaptive nearest neighbor classification". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.6 (June 1996), pp. 607–616.

[42] T. Hastie and C. Loader. "Local Regression: Automatic Kernel Carpentry". In: *Statistical Science* 8.2 (1993), pp. 120–129.

[43] A. E. Hoerl and R. W. Kennard. "Ridge Regression: Biased Estimation for Nonorthogonal Problems". In: *Technometrics* 12.1 (Feb. 1970), pp. 55–67.

[44] P. W. Holland. "Statistics and causal inference". In: *Journal of the American Statistical Association* 81.396 (1986), pp. 945–960.

[45] H. Hotelling. "Relations Between Two Sets of Variates". In: *Biometrika* 28.3/4 (1936), pp. 321–377.

[46] D. H. Hubel and T. N. Wiesel. "Receptive fields and functional architecture of monkey striate cortex". In: *The Journal of Physiology* 195.1 (Mar. 1968), pp. 215–243.

[47] D. H. Hubel and T. N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". In: *The Journal of Physiology* 160.1 (Jan. 1962), pp. 106–154.

[48] P. Indyk and R. Motwani. "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality". In: *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*. 1998, pp. 604–613.

[49] M. Jaggi, V. Smith, M. Takac, J. Terhorst, S. Krishnan, T. Hofmann and M. I. Jordan. "Communication-Efficient Distributed Dual Coordinate Ascent". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence and K. Q. Weinberger. Curran Associates, Inc., 2014, pp. 3068–3076.

[50] M. Kohler. "Universal Consistency of Local Polynomial Kernel Regression Estimates". In: *Annals of the Institute of Statistical Mathematics* 54.4 (2002), pp. 879–899.

[51] J. Lafferty and L. Wasserman. "Rodeo: Sparse, Greedy Nonparametric Regression". In: *The Annals of Statistics* 36.1 (2008), pp. 28–63.

[52] S. Lazebnik, C. Schmid and J. Ponce. "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 2. 2006, pp. 2169–2178.

[53] L. Li, E. Tchetgen Tchetgen, A. van der Vaart and J. M. Robins. "Higher order inference on a treatment effect under low regularity conditions". In: *Statistics & Probability Letters* 81.7 (2011), pp. 821–828.

[54] W. Lin. "Agnostic notes on regression adjustments to experimental data: Reexamining Freedman's critique". In: *The Annals of Applied Statistics* 7.1 (2013), pp. 295–318.

[55] Y. Lin and Y. Jeon. "Random Forests and Adaptive Nearest Neighbors". In: *Journal of the American Statistical Association* 101.474 (2006), pp. 578–590.

[56] H. Liu and B. Yu. "Asymptotic properties of Lasso+mLS and Lasso+Ridge in sparse high-dimensional linear regression". In: *Electronic Journal of Statistics* 7.1 (2013), pp. 3124–3169.

[57] L. W. Mackey, M. I. Jordan and A. Talwalkar. "Divide-and-Conquer Matrix Factorization". In: *Advances in Neural Information Processing Systems 24*. Ed. by J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira and K. Q. Weinberger. Curran Associates, Inc., 2011, pp. 1134–1142.

[58] J. Mairal, F. Bach, J. Ponce and G. Sapiro. "Online Learning for Matrix Factorization and Sparse Coding". In: *The Journal of Machine Learning Research* 11 (Mar. 2010), pp. 19–60.

[59] J. Mairal, Y. Benjamini, B. D. B. Willmore, M. Oliver, J. L. Gallant and B. Yu. "Modeling V4 under Naturalistic Conditions with Invariant Image Representations". In preparation (2016).

[60] P. Massart. "Rates of convergence in the central limit theorem for empirical processes". In: *Geometrical and Statistical Aspects of Probability in Banach Spaces*. Springer, 1986, pp. 73–109.

[61] J. H. R. Maunsell and D. C. Van Essen. "Functional properties of neurons in middle temporal visual area of the macaque monkey. I. Selectivity for stimulus direction, speed, and orientation". In: *Journal of Neurophysiology* 49.5 (May 1983), pp. 1127–1147.

[62] J. A. Mazer, W. E. Vinje, J. McDermott, P. H. Schiller and J. L. Gallant. "Spatial Frequency and Orientation Tuning Dynamics in Area V1". In: *Proceedings of the National Academy of Sciences of the United States of America* 99.3 (2002), pp. 1645–1650.

[63] N. Meinshausen. "Quantile Regression Forests". In: *The Journal of Machine Learning Research* 7 (Dec. 2006), pp. 983–999.

[64] N. Meinshausen. "Relaxed Lasso". In: *Computational Statistics & Data Analysis* 52.1 (Sept. 2007), pp. 374–393.

[65] X. Meng, J. Bradley, B. Yavuz, E. Sparks et al. "MLlib: Machine Learning in Apache Spark". In: *ArXiv e-prints* (May 2015). arXiv: 1505.06807 [cs.LG].

[66] L. W. Miratrix, J. S. Sekhon and B. Yu. "Adjusting treatment effect estimates by post-stratification in randomized experiments". In: *Journal of the Royal Statistical Society. Series B: Statistical Methodology* 75.2 (2013), pp. 369–396.

[67] W. T. Newsome and E. B. Pare. "A selective impairment of motion perception following lesions of the middle temporal visual area (MT)". In: *The Journal of Neuroscience* 8.6 (1988), pp. 2201–2211.

[68] W. T. Newsome, R. H. Wurtz, M. R. Dursteler and A. Mikami. "Deficits in visual motion processing following ibotenic acid lesions of the middle temporal visual area of the macaque monkey". In: *The Journal of Neuroscience* 5.3 (Mar. 1985), pp. 825–840.

[69] S. Nishimoto and J. L. Gallant. "A three-dimensional spatiotemporal receptive field model explains responses of area MT neurons to naturalistic movies". In: *The Journal of Neuroscience* 31.41 (2011), pp. 14551–14564.

[70] S. Nishimoto, A. T. Vu, T. Naselaris, Y. Benjamini, B. Yu and J. L. Gallant. "Reconstructing Visual Experiences from Brain Activity Evoked by Natural Movies". In: *Current Biology* 21.19 (Oct. 2011), pp. 1641–1646.

[71] B. A. Olshausen. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". In: *Nature* 381.6583 (1996), pp. 607–609.

[72] B. A. Olshausen and D. J. Field. "Sparse coding of sensory inputs". In: *Current Opinion in Neurobiology* 14.4 (Aug. 2004), pp. 481–487.

[73] B. A. Olshausen and D. J. Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision research* 37.23 (1997), pp. 3311–3325.

[74] B. A. Olshausen and D. J. Field. "What Is the Other 85% of V1 Doing?" In: *23 Problems in Systems Neuroscience*. Ed. by J. L. van Hemmen and T. J. Sejnowski. Oxford University Press, 2005. Chap. 10, pp. 182–213.

[75] C. C. Pack and R. T. Born. "Temporal dynamics of a neural solution to the aperture problem in visual area MT of macaque brain". In: *Nature* 409.6823 (Feb. 2001), pp. 1040–1042.

[76] B. Panda, J. S. Herbach, S. Basu and R. J. Bayardo. "PLANET: Massively Parallel Learning of Tree Ensembles with MapReduce". In: *Proc. VLDB Endow.* 2.2 (Aug. 2009), pp. 1426–1437.

[77] T. Permutt. "Testing for imbalance of covariates in controlled experiments". In: *Statistics in Medicine* 9.12 (1990), pp. 1455–1462.

[78] G. Raskutti, M. J. Wainwright and B. Yu. "Minimax Rates of Estimation for High-Dimensional Linear Regression Over ℓ_q-Balls". In: *IEEE Transactions on Information Theory* 57.10 (2011), pp. 6976–6994.

[79] B. Recht, C. Re, S. Wright and F. Niu. "HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent". In: *Advances in Neural Information Processing Systems 24*. Ed. by J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira and K. Q. Weinberger. Curran Associates, Inc., 2011, pp. 693–701.

[80] M. Rosenblum, H. Liu and Y. En-Hsu. "Optimal Tests of Treatment Effects for the Overall Population and Two Subpopulations in Randomized Trials, Using Sparse Linear Programming". In: *Journal of the American Statistical Association* 109.507 (2014), pp. 1216–1228.

[81] D. B. Rubin. "Estimating causal effects of treatments in randomized and nonrandomized studies." In: *Journal of Educational Psychology* 66.5 (1974), pp. 688–701.

[82] E. Scornet, G. Biau and J.-P. Vert. "Consistency of random forests". In: *The Annals of Statistics* 43.4 (2015), pp. 1716–1741.

[83] C. S. Sherrington. "Flexion-reflex of the limb, crossed extension-reflex, and reflex stepping and standing". In: *The Journal of Physiology* 40.1-2 (Apr. 1910), pp. 28–121.

[84] E. P. Simoncelli and D. J. Heeger. "A model of neuronal responses in visual area MT". In: *Vision Research* 38.5 (Mar. 1998), pp. 743–761.

[85] G. C. S. Smith and J. P. Pell. "Parachute use to prevent death and major trauma related to gravitational challenge: systematic review of randomised controlled trials". In: *British Medical Journal* 327.7429 (Dec. 2003), pp. 1459–1461.

[86] J. Splawa-Neyman, D. M. Dabrowska and T. P. Speed. "On the Application of Probability Theory to Agricultural Experiments. Essay on Principles. Section 9". In: *Statistical Science* 5.4 (Nov. 1990), pp. 465–472.

[87] C. J. Stone. "Optimal Global Rates of Convergence for Nonparametric Regression". In: *The Annals of Statistics* 10.4 (1982), pp. 1040–1053.

[88] C. J. Stone. "Optimal Rates of Convergence for Nonparametric Estimators". In: *The Annals of Statistics* 8.6 (1980), pp. 1348–1360.

[89] L. Tian, A. Alizadeh, A. Gentles and R. Tibshirani. "A simple method for detecting interactions between a treatment and a large number of covariates". In: *Journal of the American Statistical Association* 109.508 (2014), pp. 1517–1532.

[90] R. Tibshirani. "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288.

[91] J. Touryan and J. A. Mazer. "Linear and non-linear properties of feature selectivity in V4 neurons". In: *Frontiers in Systems Neuroscience* 9 (May 2015), p. 82.

[92] A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer-Verlag New York, 2008.

[93] L. G. Ungerleider and J. V. Haxby. "'What' and 'where' in the human brain". In: *Current Opinion in Neurobiology* 4.2 (1994), pp. 157–165.

[94] D. C. Van Essen and J. H. R. Maunsell. "Hierarchical organization and functional streams in the visual cortex". In: *Trends in Neurosciences* 6 (1983), pp. 370–375.

[95] Q. H. Vuong. "Likelihood Ratio Tests for Model Selection and Non-Nested Hypotheses". In: *Econometrica* 57.2 (1989), pp. 307–333.

[96] S. Wager and S. Athey. "Estimation and Inference of Heterogeneous Treatment Effects using Random Forests". In: *ArXiv e-prints* (Oct. 2015). arXiv: 1510.04342 [stat.ME].

[97] B. D. B. Willmore, J. A. Mazer and J. L. Gallant. "Sparse coding in striate and extrastriate visual cortex". In: *Journal of Neurophysiology* 105.6 (June 2011), pp. 2907–2919.

[98] M. C.-K. Wu, S. V. David and J. L. Gallant. "Complete functional characterization of sensory neurons by system identification". In: *Annual Review of Neuroscience* 29 (2006), pp. 477–505.

[99] R. Xu, D. Nettleton and D. J. Nordman. "Case-Specific Random Forests". In: *Journal of Computational and Graphical Statistics* 25.1 (2016), pp. 49–65.

[100] J. Yang, K. Yu, Y. Gong and T. Huang. "Linear spatial pyramid matching using sparse coding for image classification". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 1794–1801.

[101] F. Ye and C.-H. Zhang. "Rate Minimaxity of the Lasso and Dantzig Selector for the ℓ_q Loss in ℓ_r Balls". In: *The Journal of Machine Learning Research* 11 (2010), pp. 3519–3540.

[102] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker and I. Stoica. "Spark: Cluster Computing with Working Sets". In: *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*. Berkeley, CA, USA: USENIX Association, 2010, p. 10.

[103] C.-H. Zhang and S. S. Zhang. "Confidence intervals for low dimensional parameters in high dimensional linear models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76.1 (2014), pp. 217–242.

[104] Y. Zhang, M. J. Wainwright and J. C. Duchi. "Communication-Efficient Algorithms for Statistical Optimization". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou and K. Q. Weinberger. Curran Associates, Inc., 2012, pp. 1502–1510.

[105] J. Zimmerman and D. Nettleton. "Case-specific random forests for big data prediction". In: *JSM Proceedings, General Methodology*. Alexandria, VA: American Statistical Assocation, 2015, pp. 2537–2543.

[106] M. Zinkevich, M. Weimer, L. Li and A. J. Smola. "Parallelized Stochastic Gradient Descent". In: *Advances in Neural Information Processing Systems 23*. Ed. by J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel and A. Culotta. Curran Associates, Inc., 2010, pp. 2595–2603.