Practical Methods for Scalable Bayesian and Causal Inference with Provable Quality Guarantees

by
Raj Agrawal
B.A., UC Berkeley (2017)
S.M., Massachusetts Institute of Technology (2020)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2021

Author ....
Department of Electrical Engineering and Computer Science
May 13, 2021

Certified by..
Tamara Broderick
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by..
Caroline Uhler
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Practical Methods for Scalable Bayesian and Causal Inference with Provable Quality Guarantees

by

Raj Agrawal

Submitted to the Department of Electrical Engineering and Computer Science
on May 13, 2021, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in Electrical Engineering and Computer Science

Abstract

Many scientific and decision-making tasks require learning complex relationships between a set of p covariates and a target response, from N observed datapoints with $N \ll p$. For example, in genomics and precision medicine, there may be thousands or millions of genetic and environmental covariates but just hundreds or thousands of observed individuals. Researchers would like to (1) identify a small set of factors associated with diseases, (2) quantify these factors' effects, and (3) test for causality. Unfortunately, in this high-dimensional data regime, inference is statistically and computationally challenging due to non-linear interaction effects, unobserved confounders, and the lack of randomized experimental data.

In this thesis, I start by addressing the problems of variable selection and estimation when there are non-linear interactions and fewer datapoints than covariates. Unlike previous methods whose runtimes scale at least quadratically in the number of covariates, my new method (SKIM-FA) uses a kernel trick to perform inference in linear time by exploiting special interaction structure. While SKIM-FA identifies potential risk-factors, not all of these factors need be causal. So next I aim to identify causal factors to aid in decision making. To this end, I show when we can extract causal relationships from observational data, even in the presence of unobserved confounders, non-linear effects, and a lack of randomized controlled data. In the last part of my thesis, I focus on experimental design. Specifically, if the observational data is not adequate, how do we optimally collect new experimental data to test if particular causal relationships of interest exist.

Thesis Supervisor: Tamara Broderick
Title: Associate Professor of Electrical Engineering and Computer Science

Thesis Supervisor: Caroline Uhler
Title: Professor of Electrical Engineering and Computer Science

Dedicated to my mom and sister.

Acknowledgments

I would like to start by thanking my two amazing advisors Tamara Broderick and Caroline Uhler for all of their guidance, support, and feedback over the past four years. I honestly could not ask for better advisors. They have provided me the freedom and flexibility to pursue research topics I find interesting, and have helped me identify impactful research questions. Looking back at my old work when I first started at MIT, I can't believe how much they have helped me in terms of becoming a better scientific communicator and critical thinker. Beyond research, their genuine care for me and everyone else in their groups has led to a very productive and supportive research environment!

Next, I would next like to thank my committee member Devavrat Shah for helping throughout the thesis submission process and for insightful comments. I would also like to thank my undergraduate research advisor Noureddine El Karoui for getting me started with research, and all my mentors I've had throughout my internships.

To all the friends and collaborators in Tamara and Caroline's group, I want to thank you for making research and MIT such a great experience. I would like to give special thanks to Chandler Squires, Lorenzo Masoero, Brian Trippe, Karren Yang, Jonathan Huggins, Trevor Campbell, Neha Prasad, and Uma Roy for working on research projects with me. I would like to thank my other co-authors Karthik Shanmugam, Daria Roithmayr, and Thibaut Horel as well.

Outside of research, I would like to thank Isabel Yang and the whole team at ArbiLex for their support. I would like to thank Pritpal Kanhaiya, Jonathan Lin, Manish Paranjpe, Vibhaalakshmi Sivaraman, and Eric Xu for the late-night Uber Eats and movie nights. I want to thank Christian Lau, Vaikkunth Mugunthan, and Sarath Pattathil for the weekly Friday walks and interesting discussions at the docks. I want to thank Sameed Siddiqui for working out and spotting me at the gym, Chandler Squires for the dinners and helping me put up my glass whiteboard, Lorenzo Masoero for the best TA partner, and Schrasing Tong and Jason Yu for always accompanying me to all-you-can eat at Olivera's. I also would like to thank all my friends from California, and give a special thanks to Kevin Li and Armin Askari for the fun research brainstorming sessions.

Finally, I want to thank my mom and sister who have always been there for me as well as my whole family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Consider a patient who thinks she is at risk for a certain disease. In order to diagnose her, we might use a powerful off-the-shelf machine learning method. However, we must be careful; while machine-learning algorithms have had great success for many predictive tasks, these same algorithms can sometimes give *arbitrarily* bad answers. For example, deep learning models can be fooled to grossly mis-classify images by adding small amounts of noise [Szegedy et al., 2014]; variational Bayesian inference methods can report arbitrarily poor uncertainty estimates and point estimates off by orders of magnitude [Turner and Sahani, 2011, MacKay, 2002, Huggins et al., 2018]; and many black-box machine-learning models are inadvertently biased towards certain groups of individuals [Kusner et al., 2017a]. Hence, in order to provide the patient with a *trustworthy* prediction, we need guarantees that such behavior will not occur, or at least have diagnostics to alert us when our algorithms fail.

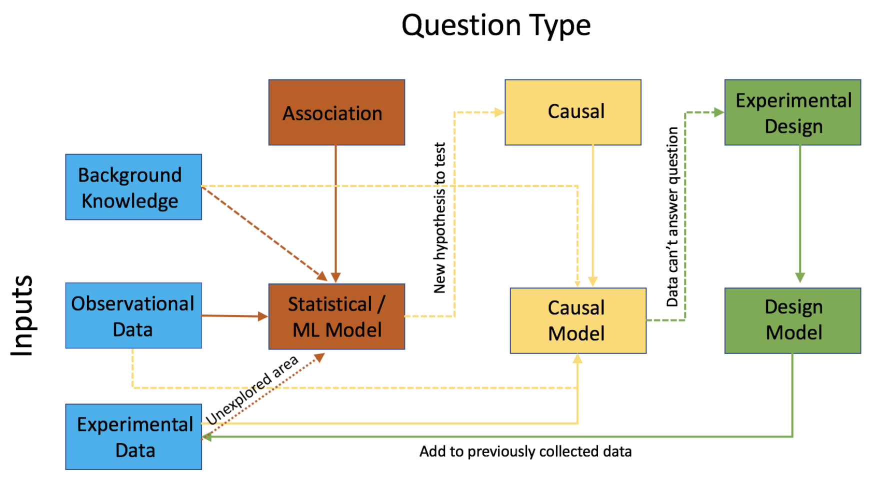


Figure 1-1: Hypothetical data-analysis pipeline broken up into three different stages: association, causality, and experimental design. Dotted arrows represent optional inputs, while solid arrows indicate necessary inputs.

Now, suppose we decide that the patient is likely to develop the disease in the next five years. Then, the natural next question is a preventative one: namely, how can the patient change her lifestyle to prevent the disease from developing? In this case, we have a causal question. Answering it requires predicting the effects of changes or *interventions* to the system (i.e., the patient's lifestyle). In this case, we seek methods that can accurately estimate causal effects, or report when such estimation is not possible from the available data. If it is not possible, then we need powerful *experimental design* methods to seek out new data to collect so that the causal question can be answered with confidence. Such an evolving process of querying and data collection is summarized in Fig. 1-1. Since I strongly believe that each stage in Fig. 1-1 is important for real-world decision-making, my research over the past 4 years has addressed each of these areas. I have demonstrated the feasibility of my methods in published work [Agrawal et al., 2018, 2019c,a,b, Trippe et al., 2019, Agrawal et al., 2021a, Margossian et al., 2020] and several papers ready for submission or under review [Horel et al., Agrawal and Broderick, 2021, Agrawal et al., 2021b].

Research Objective. My overall objective is to create a *scalable* data-analysis pipeline that informs practitioners not only how to answer association and causal questions with *provable* guarantees on quality but also how to move from association to causation through *targeted* experimental design. In what follows, I will discuss completed work in the following three areas that this thesis covers: (1) prediction and association, (2) causality, and (3) experimental design.

1.1 Prediction and Association

Chapter 2: Speeding up kernel methods. A common starting point in many data-analysis tasks is using powerful machine learning methods to make predictions, such as deciding if a patient is at risk for developing a certain disease. Recently, deep learning has led to state-of-the-art performance on many large-scale prediction tasks, most notably in image recognition. But the theoretical statistical guarantees and optimization quality of deep learning methods remain open questions. Kernel methods, on the other hand, form a flexible class of models that offer comparable performance to deep learning for many tasks yet come equipped with strong learning-theoretic guarantees. Unfortunately, they exhibit poor scaling with data size. Given N observations, $O(N^2)$ space is required to store the kernel matrix K and typically $O(N^3)$ time is required to use it for learning, as this often entails inverting K or computing its singular-value decomposition. To overcome poor scaling in N, researchers have devised various approximations to exact kernel methods. A widely-applicable and commonly used tactic is replacing K with a rank-J approximation, which reduces storage and time requirements to, respectively, $O(NJ)$ and $O(NJ^2)$ [Halko et al., 2011]. Thus, if J is sufficently small, only (near-)linear time and space is required in the dataset size. Random feature maps (RFMs) – particularly random Fourier features (RFFs) – form a popular approach to construct low-rank approximations [Kar and Karnick, 2012, Pennington et al., 2015, Daniely et al., 2017, Samo and Roberts, 2015]. The main idea of RFF is that a wide class of kernels used in practice can be written as an expectation under some induced probability distribution Q.

This alternative characterization provides a recipe to approximate the kernel through *random projections*, namely approximating the expectation via a Monte-Carlo average obtained by drawing J points from the induced probability distribution of the kernel function.

Unfortunately, J often needs to be large in RFF in order to approximate the kernel matrix well, which makes the runtime $O(NJ^2)$ intractable for N large [Honorio and Li, 2017, Kar and Karnick, 2012, Rahimi and Recht, 2007, Yang et al., 2012, Huang et al., 2014]. To this end, I reduced the number of features needed by noting that Q picks many *redundant* random features, namely features that do not further improve kernel approximation [Agrawal et al., 2019a]. The key insight of my paper, accepted at AISTATS [Agrawal et al., 2019a], is recognizing that such redundancy can be removed using a *coreset* of random features. Intuitively, a coreset summarizes a large set of points by a small, weighted set. In machine learning, coresets have been used to speed up training time by providing a smaller weighted dataset as input. My contribution was casting the problem of random feature compression as a coreset construction problem, and proving how much compression is possible. In particular, I showed that a set of J random features can be compressed to an *exponentially smaller* set of just $O(\log J)$ features while still achieving the same statistical guarantees as using all J features. Moreover, I empirically demonstrated that such a theoretical exponential rate is realized on real datasets, including one with over fifty *million* observations.

Chapter 3: Fitting Bayesian Linear interaction models in linear time. While kernel methods are successful for fitting rich, non-linear predictive models, sometimes understanding how a set of covariates relate to a target response is more important than the prediction itself. For example, in clinical trials and precision medicine, researchers seek to characterize how individual-level traits impact treatment effects, and in modern genomic studies, researchers seek to identify genetic variants that are risk factors for particular diseases. While linear regression is a default method for these tasks and many others due to its ease of interpretability, its simplicity often comes at the cost of failing to learn more nuanced information from the data. A common way to increase flexibility, while still retaining the interpretability of linear regression, is to augment the covariate space. For instance, two genes together might be highly associated with a disease even though individually they exhibit only moderate association; thus, an analyst might want to consider the multiplicative effect of pairs of covariates co-occurring.

Unfortunately, augmenting the covariate space by including all possible *pairwise interactions* means the number of parameters to analyze grows quadratically with the number of covariates p. This growth leads to many statistical and computational difficulties that are only made worse in the high-dimensional setting, where p is much larger than the number of observations N. To address both the statistical challenges and difficulty of interpreting many parameters, practitioners often enforce a sparsity constraint on the model, reflecting an assumption that only a small subset of all covariates affect the response. The problem of identifying this subset is a central problem in high-dimensional statistics and many different LASSO-based approaches have been proposed to return sparse point estimates. However, these methods do not address how to construct valid confidence intervals or adjust for multiple comparisons.

Fortunately, hierarchical Bayesian methods have a shrinkage effect, naturally handle multiplicity, can provide better statistical power than multiple comparison corrections, and can leverage background knowledge. However, naive approaches to Bayesian inference are computationally intractable for even moderate-dimensional problems. This intractability has two sources. The first source can be seen even in the simple case of conjugate linear regression with a multivariate Gaussian prior. Let $\tilde{X}$ denote the augmented data matrix including all pairwise interactions, Σ the multivariate Gaussian prior covariance on parameters, and σ^2 the noise variance. Given N observations, computing the posterior requires inverting $\Sigma^{-1} + \frac{1}{\sigma^2}\tilde{X}^T\tilde{X}$, which takes $O(p^2N^2 + N^3)$ time. The second source is that reporting on $O(p^2)$ parameters simply has $O(p^2)$ cost.

In work accepted at ICML [Agrawal et al., 2019b], I showed how to speed up inference in Bayesian interaction models by addressing both problems. In the first case, I showed how to represent the original model using a Gaussian process (GP). I used the GP kernel in my *kernel interaction sampler* (KIS) to take advantage of the special structure of interactions and avoid explicitly computing or inverting $\Sigma^{-1} + \frac{1}{\sigma^2}\tilde{X}^T\tilde{X}$. In the second case, I developed a *kernel interaction trick* to compute posterior summaries exactly for main effects and interactions between selected main effects to avoid the full $O(p^2)$ reporting cost. In sum, my method can recover posterior means and variances of non-zero regression coefficients in $O(pN^2 + N^3)$ time, a p-fold speed-up. Empirically, I found that my method leads to (1) improved Type I and Type II error relative to state-of-the-art LASSO-based approaches and (2) improved computational scaling in high dimensions relative to existing Bayesian and LASSO-based approaches.

Chapter 4: Fast High-Dimensional Functional ANOVA Decompositions. While such linear interaction models can provide an adequate first-order approximation to a signal of interest, in other applications this assumption can be problematic for both variable selection and estimation. For example, suppose that one of the covariates has a quadratic effect on the response. Then, from a purely variable selection perspective, a linear method will likely not select this covariate since a quadratic effect has weak (linear) correlation. Another limitation of our previous work was the assumption of strong-hierarchy, namely that an interaction only occurs if both main effects are present. While some problems have strong main effects, in other applications this may not be the case. For example, in genome-wide associate studies, fitting an additive-only model to predict an individual's height from genetics only has an R^2 of about 5% even though height is well-predicted by parents' heights (thought to be between $80\% - 90\%$) [Maher, 2008]. This discrepancy, more generally called the problem of *missing heritability*, remains an open challenge in biology for understanding complex diseases based on genetics. One explanation for missing heritability is not modeling genetic interactions [Maher, 2008, Aschard, 2016, Slim et al., 2018, Greene et al., 2010]. In other words, the main effects might be weak, or in the extreme case some genes might only have interaction effects.

I have extended SKIM to model non-linear effects and remove the strong-hierarchy requirement. In particular, I have shown how to perform sparse, high-dimensional functional ANOVA decompositions in time linear in dimension unlike previous methods that take at least quadratic time. By explicitly accounting for non-linear interactions and removing the strong-hierarchy constraint, our method typically has better performance than competing methods such as the Lasso and tree based methods on both simulated and real datasets.

1.2 Causality

Fast uncertainty quantification for learning causal DAGs. Moving from association to causality is an important next step not only for scientific discovery but also for many decision and policy-making tasks. As a toy example, if we are trying to fight global warming, banning ice cream will not help; even though ice cream sales and temperature are highly correlated, ice cream sales are an *effect*, not a *cause*, of temperature. Hence, *intervening* on ice cream sales, e.g. by closing ice cream shops, will not lead to any substantive change in global temperature levels.

Most real-world systems contain many variables, and we would like to understand how perturbing any part of that system changes the joint distribution. Causal *directed acyclic graphs* (DAGs) are a powerful way to model such causal structure, and enable us to seamlessly quantify the effects of perturbations – i.e., interventions to the system. Each node in a DAG G is associated with a random variable X_i, and X_i is said to be a *cause* of X_j if there exists a directed path from X_i to X_j in G. In applications where N is large relative to p, a point estimate of G suffices from both a practical and theoretical perspective [Chickering, 2002]. However, in many applications of modern interest, the number of samples N is small relative to the number of nodes p. In this case there may be many DAGs that agree with the observed data and it is then desirable to infer a distribution across DAGs instead of outputting just one DAG. Taking a Bayesian approach we can define a *prior* on the space of DAGs, which can encode expert structural knowledge about the underlying DAG – as well as desirable properties such as sparsity. The *posterior* describes the state of knowledge about G after observing the data D, but unfortunately is intractable to compute exactly since it requires summing over a *superexponential* number of DAGs [Koivisto and Sood, 2004]. In many applications, however, we only need summary statistics, namely posterior means and variances of a function $f(G)$ of the underlying causal DAG G. For example, we might set f to indicate what nodes are in the Markov blanket of a particular node. In such cases, it suffices to compute expectations of the function f under the posterior distribution, which can be computed arbitrarily closely as long as we can sample from the posterior. Problematically, previous state-of-the-art MCMC samplers either suffer from poor mixing and/or exponentially slow iteration times with respect to the maximum indegree of the DAG.

In order to achieve fast mixing chains, but remove the *exponential* timing dependence on the maximum indegree, in my paper published at ICML [Agrawal et al., 2018] I instead looked at a (suitably-chosen) reduced subspace of DAGs to average over, namely one DAG for each ordering of the nodes. The main idea was to associate each permutation to a particular type of DAG, a *minimal I-MAP*. This mapping uses the data, namely the sample covariance matrix, to construct the reduced inference space and is well-studied in the frequentist setting. A potential concern is that the data is used twice, namely to construct the inference space and then to do inference over it. However, I provided a theoretical bound for the error, i.e. the differences between the exact and approximate posterior means and variances. In particular, I showed that the error in the expectation of any functional from the approximation is *exponentially* decreasing as a function of the number of samples.

Chapter 5: Non-Linear Causal Discovery in the Presence of Hidden Variables. Currently, minimal I-MAP MCMC assumes that there are no unmeasured confounders. However, this assumption is too restrictive for many applications. The unobserved variables are problematic for structure learning because they can create spurious edges between variables in the causal graphs. For example, without conditioning on the stress-level of an individual, the posterior probability of an edge between drinking coffee and having cancer might increase. Unfortunately, in the presence of latent confounding, recovering casual relationships from observational data alone is an ill-posed problem without additional assumptions. Fortunately, in practice, we sometimes expect additional structure about the confounders. For example, in gene-expression data, batch effects can lead to incorrect associations between genes [Leek and Storey, 2007]. In genome-wide association studies, ancestry differences between case and controls can create spurious correlations in disease studies [Price et al., 2006]. In finance, the latent "market" and sector variables can explain much of the variation in stocks [Chandrasekaran et al., 2012, Fan et al., 2013]. Such confounding patterns are more generally known as *pervasive confounders*, and represent variables that have an effect on many observed variables, similar to latent factor models; see [Frot et al., 2019, Wang and Blei, 2019, Shah et al., 2020, Chandrasekaran et al., 2012] for other real-world examples of this pervasive assumption.

In Agrawal et al. [2021b], I provide a proof and specific method to estimate causal relationships in the non-linear, pervasive confounding setting. The heart of our procedure relies on the ability to actually estimate the pervasive confounding variation through a simple spectral decomposition of the data matrix. I derive a particular DAG score function based on this insight, and empirically compare this proposed method (the "DeCAMFounder") to existing procedures. By explicitly accounting for confounders and non-linear effects, the DeCAMFounder typically has better performance than competing methods on both simulated and real datasets.

1.3 Targeted Experimental Design

Chapter 6: Experimental design for learning causal DAGs. So far we have considered the case where the data is given. But in many domains, such as biology or running ads on Bing, the practitioner has control over the data being collected. In genomics, for instance, genome editing technologies have enabled the collection of batches of large-scale interventional gene expression data [Dixit et al., 2016]. An imminent problem is understanding how to optimally select a batch of interventions and allocate samples across these interventions, over multiple experimental rounds in a computationally tractable manner. One of the most popular ways is simply uniformly randomly sampling experiments to conduct. However, uniform sampling might be suboptimal by sampling similar data (redundancy) and/or data that does not help answer the target question.

Previous works showed that experimental design can improve structure recovery in causal DAG models. However, these methods assume a basic framework in which experiments are performed one sample at a time. In practice, experimenters often perform a batch of interventions and collect samples over multiple rounds of experiments; and they must also factor in budget and feasibility constraints, such as on the number of unique interventions that can be performed in a single experiment, the number of experimental rounds, and the total number of samples to be collected. Generalizing the frameworks of previous methods, I assume the experimenter is interested in learning some function $f(G)$ of the unknown graph G in my paper which was published at AISTATS [Agrawal et al., 2019c]. Returning to gene regulation, one might set $f(G)$ to indicate whether some gene X is downstream of some gene Y, i.e. if X is a *descendant* of Y in G. Using targeted experimental design, all statistical power is placed in learning the target function rather than being agnostic to recovering all features in the graph. In addition, I also explicitly took into account that only finitely many samples are allowed in each round and worked under various budget constraints such as a limit on the number of rounds and the number of unique interventions.

Since I considered the batched, finite-sample setting, picking experiments optimally turns out to be computationally intractable. To understand why, consider the problem of allocating N total samples across B batches such that each batch can have at most K unique experiments. These constraints lead to a difficult combinatorial problem. To still have optimization quality guarantees when selecting experiments to conduct, I proposed an information-based score function and proved it is *submodular*, which allows experiments to be greedily selected while having $(1 - \frac{1}{e})$ guarantees on optimization quality. Finally, I demonstrated empirically that experimental design using this method leads to significant boosts over previous methods.

Chapter 2

Data-Dependent Compression of Random Features for Large-Scale Kernel Approximation

Abstract

Kernel methods offer the flexibility to learn complex relationships in modern, large data sets while enjoying strong theoretical guarantees on quality. Unfortunately, these methods typically require cubic running time in the data set size, a prohibitive cost in the large-data setting. Random feature maps (RFMs) and the Nyström method both consider low-rank approximations to the kernel matrix as a potential solution. But, in order to achieve desirable theoretical guarantees, the former may require a prohibitively large number of features J_+, and the latter may be prohibitively expensive for high-dimensional problems. We propose to combine the simplicity and generality of RFMs with a data-dependent feature selection scheme to achieve desirable theoretical approximation properties of Nyström with just $O(\log J_+)$ features. Our key insight is to begin with a large set of random features, then reduce them to a small number of weighted features in a data-dependent, computationally efficient way, while preserving the statistical guarantees of using the original large set of features. We demonstrate the efficacy of our method with theory and experiments—including on a data set with over 50 million observations. In particular, we show that our method achieves small kernel matrix approximation error and better test set accuracy with provably fewer random features than state-of-the-art methods.

2.1 Introduction

Kernel methods are essential to the machine learning and statistics toolkit because of their modeling flexibility, ease-of-use, and widespread applicability to problems including regression, classification, clustering, dimensionality reduction, and one and two-sample testing [Hofmann et al., 2008, Schölkopf and Smola, 2001, Chwialkowski et al., 2016, Gretton et al., 2012]. In addition to good empirical performance, kernel-based methods come equipped with strong statistical and learning-theoretic guarantees [Vapnik, 1998, Mendelson, 2003, Balcan et al., 2008, Boser et al., 1992, Vapnik et al., 1997, Sriperumbudur et al., 2010]. Because kernel methods are nonparametric, they are particularly attractive for large-scale problems, where they make it possible to learn complex, highly non-linear structure from data. Unfortunately, their time and memory costs scale poorly with data size. Given N observations, storing the kernel matrix K requires $O(N^2)$ space. Using K for learning typically requires $O(N^3)$ time, as this often entails inverting K or computing its singular value decomposition.

To overcome poor scaling in N, researchers have devised various approximations to exact kernel methods. A widely-applicable and commonly used tactic is to replace K with a rank-J approximation, which reduces storage requirements to $O(NJ)$ and computational complexity of inversion or singular value decomposition to $O(NJ^2)$ [Halko et al., 2011]. Thus, if J can be chosen to be constant or slowly increasing in N, only (near-)linear time and space is required in the dataset size. Two popular approaches to constructing low-rank approximations are random feature maps (RFMs) [Kar and Karnick, 2012, Pennington et al., 2015, Daniely et al., 2017, Samo and Roberts, 2015]—particularly random Fourier features (RFFs) [Rahimi and Recht, 2007]—and Nyström-type approximations [Drineas and Mahoney, 2005]. The Nyström method is based on using J randomly sampled columns from K, and thus is data-dependent. The data-dependent nature of Nyström methods can provide statistical guarantees even when $J \ll N$, but these results either apply only to kernel ridge regression [El Alaoui and Mahoney, 2015, Yang et al., 2017, Rudi et al., 2015] or require burdensome recursive sampling schemes [Musco and Musco, 2017, Lim et al., 2018]. Random features, on the other hand, are simple to implement and use J random features that are data-independent. For problems with both large N and number of covariates p, an extension of random features called *Fast Food RFM* has been successfully applied at a fraction of the computational time required by Nyström-type approximations, which are *exponentially* more costly in terms of p [Le et al., 2013]. The price for this simplicity and data-independence is that a large number of random features is often needed to approximate the kernel matrix well [Honorio and Li, 2017, Kar and Karnick, 2012, Rahimi and Recht, 2007, Yang et al., 2012, Huang et al., 2014].

The question naturally arises, then, as to whether we can combine the simplicity of random features and the ability to scale to large-p problems with the appealing approximation and statistical properties of Nyström-type approaches. We provide one possible solution by making random features data-dependent, and we show promising theoretical and empirical results. Our key insight is to begin with a large set of random features, then reduce them to a small set of weighted features in a data-dependent, computationally efficient way, while preserving the statistical guarantees of using the original large set. We frame the task of finding this small set of features as an optimization problem, which we solve using ideas from the coreset literature [Campbell and Broderick, 2019, 2018]. Using greedy optimization schemes such as the Frank–Wolfe algorithm, we show that a large set of J_+ random features can be compressed to an *exponentially smaller* set of just $O(\log J_+)$ features while still achieving the same statistical guarantees as using all J_+ features. We demonstrate that our method achieves superior performance to existing approaches on a range of real datasets—including one with over 50 million observations—in terms of kernel matrix approximation and classification accuracy.

2.2 Preliminaries and related work

Suppose we observe data $\{(x_n, y_n)\}_{n=1}^N$ with predictors $x_n \in \mathbb{R}^p$ and responses $y_n \in \mathbb{R}$. In a supervised learning task, we aim to find a model $f : \mathbb{R}^p \to \mathbb{R}$ among a set of candidates $\mathcal{F}$ that predicts the response well for new predictors. Modern data sets of interest often reach N in the tens of millions or higher, allowing analysts to learn particularly complex relationships in data. Nonparametric kernel methods [Schölkopf and Smola, 2001] offer a flexible option in this setting; by taking $\mathcal{F}$ to be a reproducing kernel Hilbert space with positive-definite kernel $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$, they enable learning more nuanced details of the model f as more data are obtained. As a result, kernel methods are widespread not just in regression and classification but also in dimensionality reduction, conditional independence testing, one and two-sample testing, and more [Schölkopf et al., 1997, Zhang et al., 2011, Gretton et al., 2008, 2012, Chwialkowski et al., 2016].

The problem, however, is that kernel methods become computationally intractable for large N. We consider kernel ridge regression as a prototypical example [Saunders et al., 1998]. Let $K \in \mathbb{R}^{N \times N}$ be the kernel matrix consisting of entries $K_{nm} := k(x_n, x_m)$. Collect the responses into the vector $y \in \mathbb{R}^N$. Then kernel ridge regression requires solving

$$\min_{\alpha \in \mathbb{R}^N} -\frac{1}{2}\alpha^T(K + \lambda I)\alpha + \alpha^T y,$$

where $\lambda > 0$ is a regularization parameter. Computing and storing K alone has $O(N^2)$ complexity, while computing the solution $\alpha^\star = (K + \lambda I)^{-1} y$ further requires solving a linear system, with cost $O(N^3)$. Many other kernel methods have $O(N^3)$ dependence; see Table 2.1.

To make kernel methods tractable on large datasets, a common practice is to replace the kernel matrix K with an approximate low-rank factorization $\hat{K} := ZZ^T \approx K$, where $Z \in \mathbb{R}^{N \times J}$ and $J \ll N$. This factorization can be viewed as replacing the kernel function k with a finite-dimensional inner product $k(x_n, x_m) \approx z(x_n)^T z(x_m)$ between features generated by a *feature map* $z : \mathbb{R}^p \to \mathbb{R}^J$. Using this type of approximation significantly reduces downstream training time, as shown in the second column of Table 2.1. Previous results show that as long as ZZ^T is close to K in the Frobenius norm, the optimal model f using $\hat{K}$ is uniformly close to the one using K [Cortes et al., 2010]; see the rightmost column of Table 2.1.

However, finding a good feature map is a nontrivial task. One popular method, known as *random Fourier features* (RFF) [Rahimi and Recht, 2007], is based on Bochner's Theorem:

Theorem 2.2.1 ([Rudin, 1994, p. 19]). *A continuous, stationary kernel $k(x, y) = \phi(x - y)$ for $x, y \in \mathbb{R}^p$ is positive definite with $\phi(0) = 1$ if and only if there exists a*

Table 2.1: A comparison of training time for PCA, SVM, and ridge regression using the exact kernel matrix K versus a low-rank approximation $\hat{K} = ZZ^T$, where Z has J columns. Exact training requires either inverting or computing the SVD of the true kernel matrix K at a cost of $O(N^3)$ time, as shown in the first column. The second column refers to training the methods using a low-rank factorization Z. For ridge regression and PCA, the low-rank training cost reflects the time to compute and invert the feature covariance matrix $Z^T Z$. For SVM, the time refers to fitting a linear SVM on Z using *dual-coordinate descent* with optimization tolerance ρ [Hsieh et al., 2008]. The third column quantifies the uniform error between the function fit using K and the function fit using Z. For specific details of how the bounds were derived, see Appendix A.4.

Method	Exact Training Cost	Low-Rank Training Cost	Approximation Error
PCA	$O(N^3)$	$\Theta(NJ^2)$	$O\left((1-\frac{\ell}{N})\|\hat{K}-K\|_F\right)$
SVM	$O(N^3)$	$\Theta(NJ\log\frac{1}{\rho})$	$O\left(\|\hat{K}-K\|_F^{\frac{1}{2}}\right)$
Ridge Regression	$O(N^3)$	$\Theta(NJ^2)$	$O\left(\frac{1}{N}\|\hat{K}-K\|_F\right)$

probability measure Q such that

$$\begin{aligned}\phi(x-y) &= \int_{\mathbb{R}^p} e^{i\omega^T(x-y)}\mathrm{d}Q(\omega) \\ &= \mathbb{E}_Q[\psi_\omega(x)\psi_\omega(y)^*], \quad \psi_\omega(x) := e^{i\omega^T x}.\end{aligned} \tag{2.1}$$

Theorem 2.2.1 implies that $z_{\text{complex}}(x) := (1/\sqrt{J})[\psi_{\omega_1}(x), \cdots, \psi_{\omega_J}(x)]^T$, where $\omega_i \overset{\text{i.i.d.}}{\sim} Q$, provides a Monte-Carlo approximation of the true kernel function. As noted by Rahimi and Recht [2008], the real-valued feature map $z(x) := (1/\sqrt{J})[\cos(\omega_1^T x + b_1), \cdots, \cos(\omega_J^T x + b_J)]^T$, $b_j \overset{\text{unif.}}{\sim} [0, 2\pi]$ also yields an unbiased estimator of the kernel function; we use this feature map in what follows unless otherwise stated. The resulting $N \times J$ feature matrix Z yields estimates of the true kernel function with standard Monte-Carlo error rates of $O\left(1/\sqrt{J}\right)$ uniformly on compact sets [Rahimi and Recht, 2007, Sutherland and Schneider, 2015]. The RFF methodology also applies quite broadly. There are well-known techniques for obtaining samples from Q for a variety of popular kernels such as the squared exponential, Laplace, and Cauchy [Rahimi and Recht, 2007], as well as extensions to more general *random feature maps* (RFMs), which apply to many types of non-stationary kernels [Kar and Karnick, 2012, Pennington et al., 2015, Daniely et al., 2017].

The major drawback of RFMs is the $O(NJp)$ time and $O(NJ)$ memory costs associated with generating the feature matrix Z.[1] Although these are linear in N as

[1] *Fast Food RFM* can reduce the computational cost of generating the feature matrix to $O(NJ\log p)$ by exploiting techniques from sparse linear algebra. For simplicity, we focus on RFM here, but we note that our method can also be used on top of Fast Food RFM in cases when p is large.

desired, recent empirical evidence [Huang et al., 2014] suggests that J needs to be quite large to provide competitive performance with other data analysis techniques. Recent work addressing this drawback has broadly involved two approaches: *variance reduction* and *feature compression.* Variance reduction techniques involve modifying the standard Monte-Carlo estimate of k, e.g. with control variates, quasi-Monte-Carlo techniques, or importance sampling [Avron et al., 2016, Chang et al., 2017, Shen et al., 2017, Yu et al., 2016, Avron et al., 2017]. These approaches either depend poorly on the data dimension p (in terms of statistical generalization error), or, for a fixed approximation error, reduce the number of features J compared to RFM only by a constant. Feature compression techniques, on the other hand, involve two steps: (1) "up-projection," in which the basic RFM methodology generates a large number J_+ of features—followed by (2) "compression," in which those features are used to find a smaller number J of features while ideally retaining the kernel approximation error of the original J_+ features. Compact random feature maps [Hamid et al., 2014] represent an instance of this technique in which compression is achieved using the Johnson–Lindenstrauss (JL) algorithm [Johnson et al., 1986]. However, not only is the generation and storage of J_+ features prohibitively expensive for large datasets, JL compression is *data-independent* and leads to only a constant reduction in J_+ as we show in Appendix A.3 (see summary in Table 2.2).

2.3 Random feature compression via coresets

In this section, we present an algorithm for approximating a kernel matrix $K \in \mathbb{R}^{N\times N}$ with a low-rank approximation $K \approx \hat{K} = ZZ^T$ obtained using a novel feature compression technique. In the up-projection step we generate J_+ random features, but only compute their values for a small, randomly-selected subset of $S \ll N^2$ datapoint pairs. In the compression step, we select a sparse, weighted subset of J of the original J_+ features in a sequential greedy fashion. We use the feature values on the size-S subset of all possible data pairs to decide, at each step, which feature to include and its weight. Once this process is complete, we compute the resulting weighted subset of J features on the whole dataset. We use this low-rank approximation of the kernel in our original learning problem. Since we use a sparse weighted feature subset for compression—as opposed to a general linear combination as in previous work—we do not need to compute all J_+ features for the whole dataset. This circumvents the expensive $O(NJ_+p)$ up-projection computation typical of past feature compression methods. In addition, we show that our greedy compression algorithm needs to output only $J = O(\log J_+)$ features—as opposed to past work, where $J = O(J_+)$ was required—while maintaining the same kernel approximation error provided by RFM with J_+ features. These results are summarized in Table 2.2 and discussed in detail in Section 2.3.2.

2.3.1 Algorithm derivation

Let $Z_+ \in \mathbb{R}^{N \times J_+}$, $J_+ > J$, be a fixed up-projection feature matrix generated by RFM. Our goal is to use Z_+ to find a compressed low-rank approximation $\hat{K} = ZZ^T \approx K$, $Z \in \mathbb{R}^{N \times J}$. Our approach is motivated by the fact that spectral 2-norm bounds on $K - \hat{K}$ provide uniform bounds on the difference between learned models using K and $\hat{K}$ [Cortes et al., 2010], as well as the fact that the Frobenius norm bounds the 2-norm. So we aim to find a Z that minimizes the Frobenius norm error $\|K - ZZ^T\|_F$. By the triangle inequality,

$$\begin{aligned}&\|K - ZZ^T\|_F \\ &\leq \|K - Z_+{Z_+}^T\|_F + \|Z_+{Z_+}^T - ZZ^T\|_F,\end{aligned} \tag{2.2}$$

so constructing a good feature compression down to J features amounts to picking Z such that $Z_+{Z_+}^T \approx ZZ^T$ in Frobenius norm. Let $Z_{+j} \in \mathbb{R}^N$ denote the jth column of Z_+. Then we would ideally like to solve the optimization problem

$$\begin{aligned}&\underset{w \in \mathbb{R}_+^{J_+}}{\operatorname{argmin}} \quad \frac{1}{N^2}\|Z_+{Z_+}^T - Z(w)Z(w)^T\|_F^2 \\ &\text{s.t.} \quad Z(w) := \left[\begin{array}{ccc} \sqrt{w_1}Z_{+1} & \cdots & \sqrt{w_{J_+}}Z_{+J_+} \end{array}\right] \\ &\qquad\qquad \|w\|_0 \leq J.\end{aligned} \tag{2.3}$$

This problem is intractable to solve exactly for two main reasons. First, computing the objective function requires computing Z_+, which itself takes $\Omega(NJ_+p)$ time. But it is not uncommon for all three of N, J_+, and p to be large, making this computation expensive. Second, the cardinality, or "0-norm," constraint on w yields a difficult combinatorial optimization. In order to address these issues, first note that

$$\begin{aligned}&\frac{1}{N^2}\|Z_+{Z_+}^T - Z(w)Z(w)^T\|_F^2 = \\ &\mathbb{E}_{i,j \overset{\text{i.i.d.}}{\sim} \pi}\left[({z_+}_i^T z_{+j} - z_i(w)^T z_j(w))^2\right],\end{aligned}$$

where π is the uniform distribution on the integers $\{1, \ldots, N\}$, and $z_{+i}, z_i(w) \in \mathbb{R}^{J_+}$ are the ith rows of Z_+, $Z(w)$, respectively. Therefore, we can generate a Monte-Carlo estimate of the optimization objective by sampling S pairs $i_s, j_s \overset{\text{i.i.d.}}{\sim} \pi$:

$$\begin{aligned}&\frac{S}{N^2}\|Z_+{Z_+}^T - Z(w)Z(w)^T\|_F^2 \\ &\approx \sum_{s=1}^{S}({z_+}_{i_s}^T z_{+j_s} - z_{i_s}(w)^T z_{j_s}(w))^2 \\ &= (1-w)^T RR^T(1-w) \text{ s.t.}\end{aligned} \tag{2.4}$$

$$R := \left[\begin{array}{ccc} z_{+i_1} \circ z_{+j_1}, & \cdots, & z_{+i_S} \circ z_{+j_S} \end{array}\right] \in \mathbb{R}^{J_+ \times S},$$

where $\circ$ indicates a component-wise product. Denoting the jth row of R by $R_j \in \mathbb{R}^S$ and the sum of the rows by $r = \sum_{j=1}^{J_+} R_j$, we can rewrite the Monte Carlo approximation of the original optimization problem in Eq. (2.3) as

$$\begin{aligned} \underset{w \in \mathbb{R}_+^{J_+}}{\operatorname{argmin}} \quad & \|r - r(w)\|_2^2 \\ \text{s.t.} \quad & \|w\|_0 \leq J, \end{aligned} \tag{2.5}$$

where $r(w) := \sum_{j=1}^{J_+} w_j R_j$. Note that the s^{th} component $r_s = z_{+i_s}^T z_{+j_s}$ of r is the Monte-Carlo approximation of $k(x_{i_s}, x_{j_s})$ using all J_+ features, while $r(w)_s = (\sqrt{w} \circ z_{+i_s})^T (\sqrt{w} \circ z_{+j_s})$ is the sparse Monte-Carlo approximation using weights $w \in \mathbb{R}_+^{J_+}$. In other words, the difference between the full optimization in Eq. (2.3) and the reformulated optimization in Eq. (2.5) is that the former attempts to find a sparse, weighted set of features that approximates the full J_+-dimensional feature inner products for all data pairs, while the latter attempts to do so only for the subset of pairs i_s, j_s, $s \in \{1, \dots, S\}$. Since a kernel matrix is symmetric and $k(x_n, x_n) = 1$ for any datapoint x_n, we only need to sample (i, j) above the diagonal of the $N \times N$ matrix (see Algorithm 1).

The reformulated optimization problem in Eq. (2.5)—i.e., approximating the sum r of a collection $(R_j)_{j=1}^{J_+}$ of vectors in $\mathbb{R}^S$ with a sparse weighted linear combination—is precisely the *Hilbert coreset construction problem* studied in previous work [Campbell and Broderick, 2019, 2018]. There exist a number of efficient algorithms to solve this problem approximately; in particular, the Frank–Wolfe-based method of Campbell and Broderick [2019] and "greedy iterative geodesic ascent" (GIGA) [Campbell and Broderick, 2018] both provide an exponentially decreasing objective value as a function of the compressed number of features J. Note that it is also possible to apply other more general-purpose methods for cardinality-constrained convex optimization [Chen et al., 1998, Candes and Tao, 2007, Tibshirani, 1994], but these techniques are often too computationally expensive in the large-dataset setting. Our overall algorithm for feature compression is shown in Algorithm 1.

2.3.2 Theoretical results

In order to employ Algorithm 1, we must choose the number S of data pairs, the up-projected feature dimension J_+, and compressed feature dimension J. Selecting these three quantities involves a tradeoff between the computational cost of using Algorithm 1 and the resulting low-rank kernel approximation Frobenius error, but it is not immediately clear how to perform that tradeoff. Theorem 2.3.2 and Corollary 2.3.3 provide a remarkable resolution to this issue: roughly, if we fix J_+ such that the basic random features method provides kernel approximation error $\epsilon > 0$ with high probability, then choosing $S = \Omega(J_+^2 (\log J_+)^2)$ and $J = \Omega(\log J_+)$ suffices to guarantee that the compressed feature kernel approximation error is also $O(\epsilon)$ with high probability. In contrast, previous feature compression methods required $J = \Omega(J_+)$ to achieve the same result; see Table 2.2. Note that Theorem 2.3.2 assumes that the

Algorithm 1 Random Feature Maps Compression (RFM-FW / RFM-GIGA)

```
    Input: Data $(x_n)_{n=1}^N$ in $\mathbb{R}^p$, RFM distribution $Q$, number of starting random features $J_+$, number of compressed features $J$, number of data pairs $S$
    Output: Weights $w \in \mathbb{R}^{J_+}$ with at most $J$ non-zero entries
 1: $(i_s, j_s)_{s=1}^S \overset{\text{i.i.d.}}{\sim} \text{Unif}\left(\{(i,j) : i < j, 2 \leq j \leq N\}\right)$.
 2: Sample $(\omega_j)_{j=1}^{J_+} \overset{\text{i.i.d.}}{\sim} Q$
 3: Sample $b_j \overset{\text{unif.}}{\sim} [0, 2\pi], 1 \leq j \leq J_+$
 4: for $s = 1 : S$ do
 5:     Compute $z_{+i_s} \leftarrow (1/\sqrt{J_+})[\cos(\omega_1^T x_{i_s} + b_1), \cdots, \cos(\omega_{J_+}^T x_{i_s} + b_{J_+})]^T$; same for $z_{+j_s}$
 6: Compute $R \leftarrow \left[z_{+i_1} \circ z_{+j_1}, \ \cdots, \ z_{+i_S} \circ z_{+j_S}\right]$
 7: $R_j \leftarrow$ row $j$ of $R$; $r \leftarrow \sum_{j=1}^{J_+} R_j$
 8: $w \leftarrow$ solution to Eq. (2.5) with FW [Campbell and Broderick, 2019] or GIGA [Campbell and Broderick, 2018]
 9: $Z(w) = \left[ \ \sqrt{w_1} Z_{+1} \ \cdots \ \sqrt{w_{J_+}} Z_{+J_+} \ \right]$
10: return $Z(w)$
```

compression step in Algorithm 1 is completed using the Frank–Wolfe-based method from Campbell and Broderick [2019]. However, this choice was made solely to simplify the theory; as GIGA [Campbell and Broderick, 2018] provides stronger performance both theoretically and empirically, we expect a stronger result than Theorem 2.3.2 and Corollary 2.3.3 to hold when using GIGA. The proof of Theorem 2.3.2 is given in Appendix A.2 and depends on the following assumptions.

Assumption 2.3.1. (a) The cardinality of the set of vectors $\{x_i - x_j, x_i + x_j\}_{1 \leq i < j \leq N}$ is $\frac{N(N-1)}{2}$, i.e., all vectors $x_i - x_j, x_i + x_j, 1 \leq i < j \leq N$ are distinct.

(b) $Q(\omega)$ for $\omega \in \mathbb{R}^p$ has strictly positive density on all of $\mathbb{R}^p$, where Q is the measure induced by the kernel k; see Theorem 2.2.1.

Assumption 2.3.1(a-b) are sufficient to guarantee that the compression coefficient ν_{J_+} provided in Theorem 2.3.2 does not go to 1. If $\nu_{J_+} \to 1$ as $J_+ \to \infty$, the amount of compression could go to zero asymptotically. When the x_j's contain continuous (noisy) measurements, Assumption 2.3.1(a) is very mild since the difference or sum between two datapoints is unlikely to equal the difference or sum between two other datapoints. Assumption 2.3.1(b) is satisfied by most kernels used in practice (e.g. radial basis function, Laplace kernel, etc.).

We obtain the exponential compression in Theorem 2.3.2 for the following reason: Frank-Wolfe and GIGA converge linearly when the minimizer of Eq. (2.5) belongs to the relative interior of the feasible set of solutions [Marguerite and Philip, 1956], which turns out to occur in our case. With linear convergence, we need to run only a *logarithmic* number of iterations (which upper bounds the sparsity of w) to approximate r by $r(w)$ for a given level of approximation error. For fixed J_+, Lemma

Table 2.2: A comparison of the computational cost of basic random feature maps (RFM), RFM with JL compression (RFM-JL), and RFM with our proposed compression using FW (RFM-FW) for N datapoints and $J_+ = {}^1\!/\!_\epsilon \log {}^1\!/\!_\epsilon$ up-projection features. The first column specifies the number of compressed features J needed to retain the $O(\epsilon)$ high probability kernel approximation error guarantee of RFM. The second and third columns list the complexity for computing the compressed features and using them for PCA or ridge regression, respectively. Theoretically, the number of datapoint pairs S should be set to $\Omega({J_+}^2(\log J_+)^2)$ in Algorithm 1 (see Theorem 2.3.2) but empirically we find in Section 6.5 that S can be set much smaller. See Appendix A.3 for derivations.

Method	# Compressed Features J	Cost of Computing Z	PCA/Ridge Reg. Cost
RFM	$O(J_+)$	$O(NJ_+)$	$O\left(NJ_+^2\right)$
RFM-JL	$O(J_+)$	$O(NJ_+ \log J_+)$	$O\left(NJ_+^2\right)$
RFM-FW	$O(\log J_+)$	$O(SJ_+ \log J_+ + N \log J_+)$	$O\left(N(\log J_+)^2\right)$

A.5 from Campbell and Broderick [2019] immediately implies that the minimizer belongs to the relative interior. As $J_+ \to \infty$ (that is, as we represent the kernel function exactly), we show that the minimizer asymptotically belongs to the relative interior, and we provide a lower bound on its distance to the boundary of the feasible set. This distance lower bound is key to the asymptotic worst-case bound on the compression coefficient given in Theorem 2.3.2 and Theorem 2.3.4.

Theorem 2.3.2. *Fix $\epsilon > 0$, $\delta \in (0,1)$, and $J_+ \in \mathbb{N}$. Then there are constants $\nu_{J_+} \in (0,1)$, which depends only on J_+, and $0 \le c^*_\delta < \infty$, which depends only on δ, such that if*

$$J = \Omega\left(-\frac{\log J_+}{\log \nu_{J_+}}\right) \text{ and } S = \Omega\left(\frac{c^*_\delta}{\epsilon^2}\left[\frac{\log \frac{1}{\epsilon}}{\log \nu_{J_+}}\right]^4 \log J_+\right),$$

then with probability at least $1-\delta$, the output Z of Algorithm 1 satisfies

$$\frac{1}{N^2}\|Z_+{Z_+}^T - ZZ^T\|_F^2 \le \epsilon.$$

Furthermore, the compression coefficient is asymptotically bounded away from 1. That is,

$$0 < \limsup_{J_+ \to \infty} \nu_{J_+} < 1. \tag{2.6}$$

Corollary 2.3.3. *In the setting of Theorem 2.3.2, if we let $J_+ = \Omega({}^1\!/\!_\epsilon \log {}^1\!/\!_\epsilon)$, then*

$$\frac{1}{N^2}\|K - ZZ^T\|_F^2 = O(\epsilon).$$

Proof. Claim 1 of Rahimi and Recht [2007] implies that $\frac{1}{N^2}\|K - Z_+{Z_+}^T\|_F^2 = O(\epsilon)$ if we set $J_+ = \Omega(1/\epsilon \log 1/\epsilon)$. The result follows by combining Theorem 2.3.2 and Eq. (2.2). □

Table 2.2 builds on the results of Theorem 2.3.2 and Corollary 2.3.3 to illustrate the benefit of our proposed feature compression technique in the settings of kernel principal component analysis (PCA) and ridge regression. Since random features and random features with JL compression both have $J = \Omega(J_+)$, the $O(NJ_+^2)$ cost of computing the feature covariance matrix $Z^T Z$ dominates when training PCA or ridge regression. In contrast, the dominant cost of random features with our proposed algorithm is the compression step; each iteration of Frank-Wolfe has cost $O(J_+ S)$, and we run it for $O(\log J_+)$ iterations.

While Corollary 2.3.3 says how large S must be for a given J_+, it does not say how to pick J_+, or equivalently how to choose the level of precision ϵ. As one would expect, the amount of precision needed depends on the downstream application. For example, recent theoretical work suggests that both kernel PCA and kernel ridge regression require J_+ to scale only sublinearly with the number of datapoints N to achieve the same statistical guarantees as an exact kernel machine trained on all N datapoints [Sriperumbudur and Sterge, 2017, Avron et al., 2017, Rudi and Rosasco, 2017]. For kernel support vector machines (SVMs), on the other hand, Sutherland and Schneider [2015] suggest that J_+ needs to be larger than N. Such a choice of J_+ would make random features *slower* than training an exact kernel SVM. However, since Sutherland and Schneider [2015] do not provide a lower bound, it is still an open theoretical question how J_+ must scale with N for kernel SVMs.

For J_+ even moderately large, setting $S = \Omega(J_+^2(\log J_+)^2))$ to satisfy Theorem 2.3.2 will be prohibitively expensive. Fortunately, in practice, we find $S \ll J_+^2$ suffices to provide significant practical computational gains without adversely affecting approximation error; see the results in Section 6.5. We conjecture that we see this behavior since we expect even a small number of data pairs S to be enough to guide feature compression in a data-dependent manner. We empirically verify this intuition in Fig. 2-4 of Section 6.5.

Finally, we provide an asymptotic upper bound for the compression coefficient ν_{J_+}. We achieve greater compression when $\nu_{J_+} \downarrow 0$. Hence, the upper bound below shows the asymptotic worst-case rate of compression.

Theorem 2.3.4. *Suppose all $\{(i,j) : 1 \le i < j \le N\}$ are sampled in Algorithm 1. Then,*

$$0 < \limsup_{J_+ \to \infty} \nu_{J_+} < 1 - \frac{\left(1 - \frac{\|K\|_F}{c_Q}\right)^2}{2} < 1, \tag{2.7}$$

where K is the exact kernel matrix and

$$\begin{aligned} c_Q &:= \frac{1}{N}\mathbb{E}_{\omega\sim Q, b\sim \mathit{Unif}[0,2\pi]}\|u(\omega,b)\|_2, \text{ with} \\ u(\omega,b) &:= (\cos(\omega^T x_i + b)\cos(\omega^T x_j + b))_{i,j\in[N]}. \end{aligned} \tag{2.8}$$

By Theorem 2.2.1, $\|K\|_F = \frac{1}{N}\|\mathbb{E}_{\omega,b} u(\omega,b)\|_2$, so $\|K\|_F \le c_Q$ by Jensen's inequality.

In Appendix A.1, we show this inequality holds strictly. Hence the term squared in Eq. (2.7) lies in $(0, 1]$. Recall $\|K\|_F^2 = \sum_{i=1}^N \lambda_i$, for λ_i the eigenvalues of K. With these observations, Theorem 2.3.4 says that the asymptotic worst-case rate of compression improves if K's eigenvalue sum is smaller. As rough intuition: If the sum is small, then K may be nearly low-rank and thus easier to approximate via a low-rank approximation. Since we subsample only S of all pairs in Theorem 2.3.2, the upper bound in Theorem 2.3.4 does not necessarily apply. Nonetheless, for S moderately large, this upper bound roughly characterizes the worst-case compression rate for Algorithm 1.

2.4 Experiments

In this section we provide an empirical comparison of basic random feature maps (RFM) [Rahimi and Recht, 2007], RFM with Johnson-Lindenstrauss compression (RFM-JL) [Hamid et al., 2014], and our proposed algorithm with compression via greedy iterative geodesic ascent [Campbell and Broderick, 2018] (RFM-GIGA). We note that there are many other random feature methods, such as Quasi-Monte-Carlo random features [Avron et al., 2016], that one might consider besides RFM-JL. A strength of our method is that it can be used as an additional compression step with these methods and is thus complementary with them; we discuss this idea and demonstrate the resulting improvements in Appendix A.5. In this section, we focus on Johnson-Lindenstrauss as the current state-of-the-art random features compression method.

We compare performance on the task of kernel SVM classification [Vapnik et al., 1997]. We consider five real, large-scale datasets, summarized in Table 2.3. We assess performance via two quality metrics—Frobenius error of the kernel approximation and test set classification error. We also measure overall computation time—including both random feature projection and SVM training. We use the radial basis kernel $k(x, y) = e^{-\gamma\|x-y\|^2}$; we pick both γ and the SVM regularization strength for each dataset by randomly sampling 10,000 datapoints, training an exact kernel SVM on those datapoints, and using 5-fold cross-validation. For both RFM-JL and RFM-GIGA we set $J_+ = 5{,}000$, and for RFM-GIGA we set $S = 20{,}000$.

Figs. 2-1 and 2-2 show the relative kernel matrix approximation error $\|ZZ^T -$

Table 2.3: All datasets are taken from `LIBSVM`.

Dataset	# Samples	Dimension	# Classes
Adult	48,842	123	2
Human	10,299	561	6
MNIST	70,000	780	10
Sensorless	58,000	9	11
Criteo	51,882,752	1,000,000	2

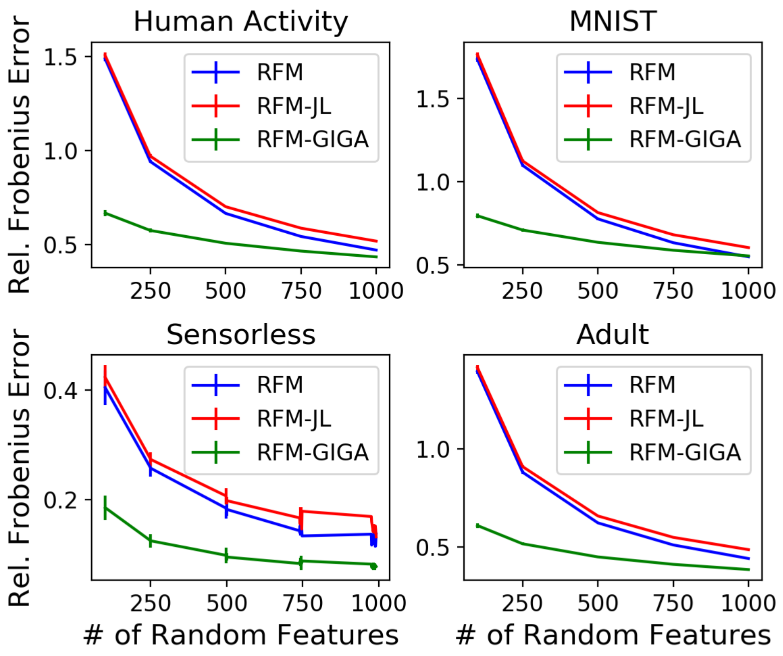


Figure 2-1: Kernel matrix approximation error. Lower is better. Points average 20 runs; error bar is one standard deviation.

$K\|_F/\|K\|_F$ and test classification accuracy, respectively, as a function of the number of compressed features J. Note that, since we cannot actually compute K, we approximate the relative Frobenius norm error by randomly sampling 10^4 datapoints. We ran each experiment 20 times; the results in Figs. 2-1 and 2-2 show the mean across these trials with one standard deviation denoted with error bars. RFM-GIGA outperforms RFM and RFM-JL across all the datasets, on both metrics, for the full range of number of compressed features that we tested. This empirical result corroborates the theoretical results presented earlier in Section 2.3.2; in practice, RFM-GIGA requires approximately an order of magnitude fewer features than either RFM or RFM-JL.

To demonstrate the computational scalability of RFM-GIGA, we also plot the relative kernel matrix approximation error versus computation time for the Criteo dataset, which consists of over 50 million data points. Before random feature projection and training, we used sparse random projections [Li et al., 2006] to reduce the input dimensionality to 250 dimensions (due to memory constraints). We set $J_+ = 5000$ and $S = 2 \times 10^4$ as before, and let J vary between 10^2 and 10^3. The results of this experiment in Fig. 2-3 suggest that RFM-GIGA provides a significant improvement in performance over both RFM and RFM-JL. Note that RFM-JL is very expensive in this setting—the up-projection step requires computing a 5×10^8 by 5×10^3 feature matrix—explaining its large computation time relative to RFM and RFM-GIGA. For test-set classification, all the methods performed the same for all choices of J (accuracy of 0.74 ± 0.001), so we do not provide the runtime vs. classification accuracy plot. This result is likely due to our compressing the 10^6-dimensional feature space to 250 dimensions, making it hard for the SVM classifier to properly learn.

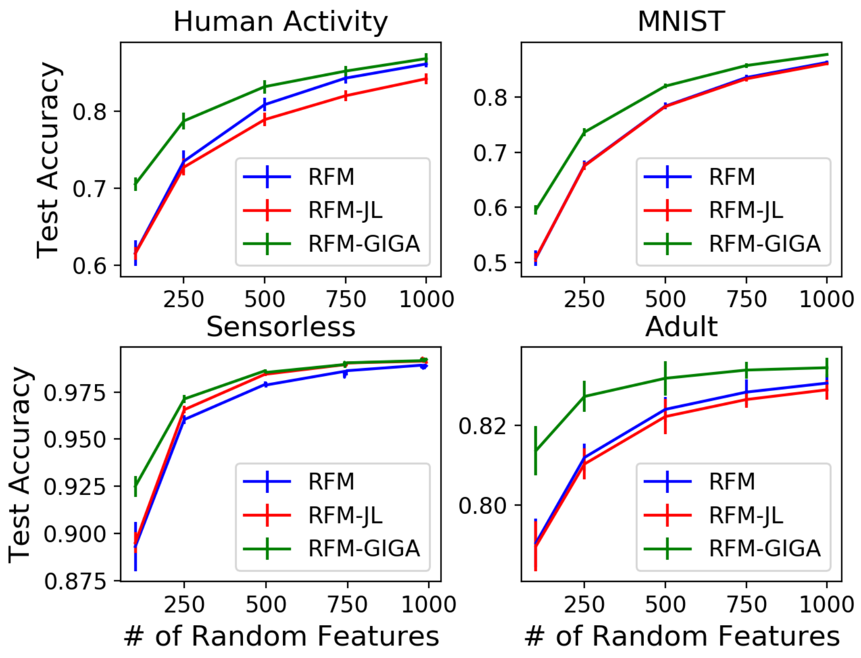


Figure 2-2: Classification accuracy. Higher is better. Points average 20 runs; error bar is one standard deviation.

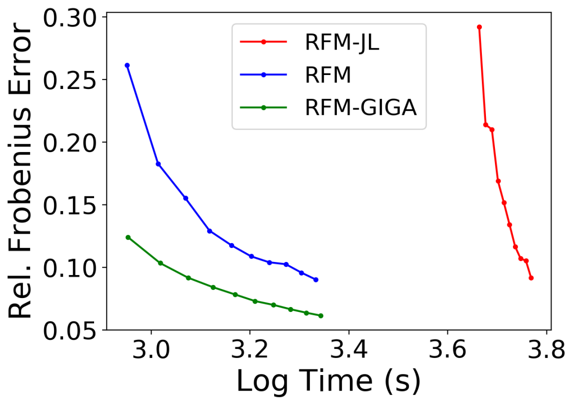


Figure 2-3: Log clock time vs. kernel matrix approximation quality on the Criteo data. Lower is better.

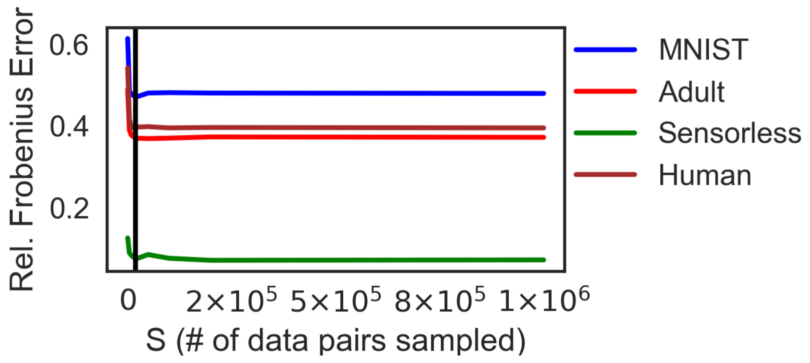


Figure 2-4: We plot the relative Frobenius norm error against S for J_+ fixed at 5,000. The solid black line corresponds to the results found in Fig. 2-1.

Given the empirical advantage of our proposed method, we next focus on understanding (1) if S can be set much smaller than $\Omega(J_+^2(\log J_+)^2))$ in practice and (2) if we can get an exponential compression of J_+ in practice as Theorem 2.3.2 and Theorem 2.3.4 guarantee.

To test the impact of S on performance, we fixed $J_+ = 5{,}000$, and we let S vary between 10^2 and 10^6. Figure 2-4 shows what the results in Fig. 2-1 would have looked like had we chosen a different S. We clearly see that after around only $S = 10{,}000$ there is a phase transition such that increasing S does not further improve performance.

To better understand if we actually see an exponential compression in J_+ in practice, as our theory suggests, we set $J_+ = 10^5$ (i.e. very large) and fixed $S = 20{,}000$ as before. We examined the HIGGS dataset consisting of 1.1×10^7 samples, and let J (the number of compressed features) vary between 500 and 10^4. Since GIGA can select the same random feature at different iterations (i.e. give a feature higher weight), J reached 8,600 after 10^4 iterations in Fig. 2-5. Fig. 2-5 shows that for $J \approx 2 \times 10^3$, increasing J further has negligible impact on kernel approximation performance—only 0.001 difference in relative error. Fig. 2-5 shows that we are able to compress J_+ by around two orders of magnitude.

Finally, since our proofs of Theorem 2.3.2 and Theorem 2.3.4 assume Step 8 of Algorithm 1 is run using Frank-Wolfe instead of GIGA, we compare in Fig. 2-6 how the results in Fig. 2-1 change by using Frank-Wolfe instead. Fig. 2-6 shows that for J small, GIGA has better approximation quality than FW but for larger J, the two perform nearly the same. This behavior agrees with the theory and empirical results of Campbell and Broderick [2018], where GIGA is motivated specifically for the case of high compression.

2.5 Conclusion

This work presents a new algorithm for scalable kernel matrix approximation. We first generate a low-rank approximation. We then find a sparse, weighted subset of the columns of the low-rank factor that minimizes the Frobenius norm error relative to the original low-rank approximation. Theoretical and empirical results suggest that our

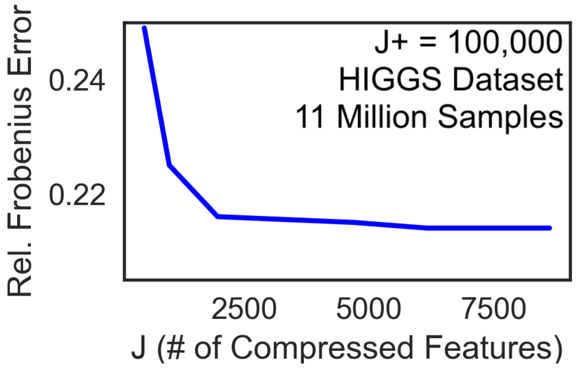


Figure 2-5: Let $S = 20{,}000$, $J_+ = 10^5$. We plot the relative Frobenius norm error vs. J from 500 to 10^4.

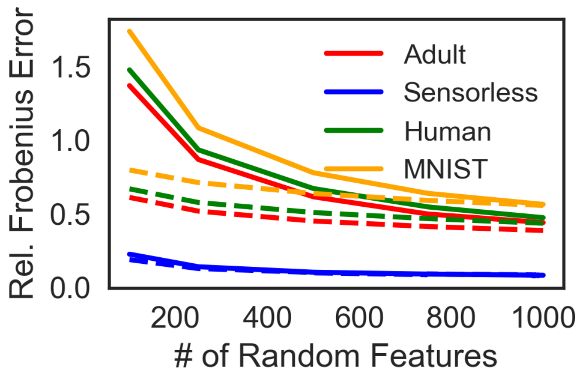


Figure 2-6: The performance of GIGA versus Frank-Wolfe for the experiment described in Fig. 2-1. Solid lines correspond to Frank-Wolfe and dashed with GIGA.

method provides a substantial improvement in scalability and approximation quality over past techniques. Directions for future work include investigating the effects of variance reduction techniques for the up-projection, using a similar compression technique on features generated by the Nyström method [Williams and Seeger, 2001], and transfer learning of feature weights for multiple related datasets.

Chapter 3

The Kernel Interaction Trick: Fast Bayesian Discovery of Pairwise Interactions in High Dimensions

Abstract

Discovering interaction effects on a response of interest is a fundamental problem faced in biology, medicine, economics, and many other scientific disciplines. In theory, Bayesian methods for discovering pairwise interactions enjoy many benefits such as coherent uncertainty quantification, the ability to incorporate background knowledge, and desirable shrinkage properties. In practice, however, Bayesian methods are often computationally intractable for even moderate-dimensional problems. Our key insight is that many hierarchical models of practical interest admit a particular Gaussian process (GP) representation; the GP allows us to capture the posterior with a vector of $O(p)$ kernel hyper-parameters rather than $O(p^2)$ interactions and main effects. With the implicit representation, we can run Markov chain Monte Carlo (MCMC) over model hyper-parameters in time and memory *linear* in p per iteration. We focus on sparsity-inducing models and show on datasets with a variety of covariate behaviors that our method: (1) reduces runtime by orders of magnitude over naive applications of MCMC, (2) provides lower Type I and Type II error relative to state-of-the-art LASSO-based approaches, and (3) offers improved computational scaling in high dimensions relative to existing Bayesian and LASSO-based approaches.

3.1 Introduction

Many decision-making and scientific tasks require understanding how a set of covariates relate to a target response. For example, in clinical trials and precision medicine, researchers seek to characterize how individual-level traits impact treatment effects, and in modern genomic studies, researchers seek to identify genetic variants that are risk factors for particular diseases. While linear regression is a default method for these tasks and many others due to its ease of interpretability, its simplicity often comes at the cost of failing to learn more nuanced information from the data. A common way to increase flexibility, while still retaining the interpretability of linear regression, is to augment the covariate space. For instance, two genes together might be highly associated with a disease even though individually they exhibit only moderate association; thus, an analyst might want to consider the multiplicative effect of pairs of covariates co-occurring.

Unfortunately, augmenting the covariate space by including all possible *pairwise interactions* means the number of parameters to analyze grows quadratically with the number of covariates p. This growth leads to many statistical and computational difficulties that are only made worse in the high-dimensional setting, where p is much larger than the number of observations N. And $p \gg N$ is often exactly the case of interest in genomic and medical applications. To address the statistical challenges, practitioners often enforce a sparsity constraint on the model, reflecting an assumption that only a small subset of all covariates affect the response. The problem of identifying this subset is a central problem in high-dimensional statistics and many different LASSO-based approaches have been proposed to return sparse point estimates. However, these methods do not address how to construct valid confidence intervals or adjust for multiple comparisons[1] [Bien et al., 2013, Lim and Hastie, 2015, Wu et al., 2009, Nakagawa et al., 2016, Shah, 2016].

Fortunately, hierarchical Bayesian methods have a shrinkage effect, naturally handle multiplicity, can provide better statistical power than multiple comparison corrections Gelman et al. [2012], and can leverage background knowledge. However, naive approaches to Bayesian inference are computationally intractable for even moderate-dimensional problems. This intractability has two sources. The first source can be seen even in the simple case of conjugate linear regression with a multivariate Gaussian prior. Let $\tilde{X}$ denote the augmented data matrix including all pairwise interactions, Σ the multivariate Gaussian prior covariance on parameters, and σ^2 the noise variance. Given N observations, computing the posterior requires inverting $\Sigma^{-1} + \frac{1}{\sigma^2}\tilde{X}^T\tilde{X}$, which takes $O(p^2N^2 + N^3)$ time. The second source is that reporting on $O(p^2)$ parameters simply has $O(p^2)$ cost.

We propose to speed up inference in Bayesian linear regression with pairwise interactions by addressing both problems. In the first case, we show how to represent the original model using a Gaussian process (GP). We use the GP kernel in our *kernel interaction sampler* to take advantage of the special structure of interactions and avoid explicitly computing or inverting $\Sigma^{-1} + \frac{1}{\sigma^2}\tilde{X}^T\tilde{X}$. In the second case, we develop a *kernel interaction trick* to compute posterior summaries exactly for main effects and interactions between selected main effects to avoid the full $O(p^2)$ reporting cost. In sum, we show that we can recover posterior means and variances of regression coefficients in $O(pN^2 + N^3)$ time, a p-fold speed-up. We demonstrate the utility and efficacy of our general-purpose computational tools for the *sparse kernel interaction model* (SKIM), which we propose in Section 3.6 for identifying sparse interactions.

[1]While the knockoff filter introduced in Barber and Candès [2015] is a promising way to control the false discovery rate, such a method has not been evaluated theoretically or empirically for interaction models.

In Section 6.5 we empirically show (1) improved Type I and Type II error relative to state-of-the-art LASSO-based approaches and (2) improved computational scaling in high dimensions relative to existing Bayesian and LASSO-based approaches. Our methods extend naturally beyond pairwise interactions to higher-order *multi-way* interactions, as detailed in Appendix B.1.

3.2 Preliminaries and Related Work

Suppose we observe data $D = \{(x^{(n)}, y^{(n)})\}_{n=1}^N$ with covariates $x^{(n)} \in \mathbb{R}^p$ and responses $y^{(n)} \in \mathbb{R}$. Let $X \in \mathbb{R}^{N \times p}$ denote the design matrix and $Y \in \mathbb{R}^N$ denote the vector of responses. Linear models assume that each $y^{(n)}$ is a (noisy) linear function of the covariates $x^{(n)}$. A common strategy to increase the expressivity of linear models is to augment the original covariates $x^{(n)}$ with their pairwise interactions

$$\Phi_2^T(x) \coloneqq [1, x_1, \cdots, x_p, x_1x_2, \cdots, x_{p-1}x_p, x_1^2, \cdots, x_p^2].$$

That is, for a parameter $\theta \in \mathbb{R}^{p(p+1)/2}$ and zero-mean i.i.d. errors $\epsilon^{(n)}$, we assume the data are generated according to

$$y^{(n)} = \theta^T \Phi_2(x^{(n)}) + \epsilon^{(n)}. \tag{3.1}$$

Our goal is to identify which interaction terms have a significant effect on the response. Detecting such interactions is important for many applications. For example, in genomics, two-way interaction terms are needed to detect possible epistasis between genes [Aschard, 2016, Slim et al., 2018] and to appropriately account for the site- and sample-specific effects of GC content on genomic and other types of sequencing data [Benjamini and Speed, 2012, Risso et al., 2011]. In economics and clinical trials, pairwise interactions between covariates and treatment are used to estimate the heterogeneous effect a treatment has across different subgroups [Lipkovich et al., 2017, Section 6]. Unfortunately, having $O(p^2)$ parameters creates statistical and computational challenges when p is large.

To address the statistical issues, practitioners often assume that θ is *sparse* (i.e., contains only a few non-zero values), and that θ satisfies *strong hierarchy.* That is, an interaction effect $\theta_{x_ix_j}$ is present only if both of the main effects θ_{x_i} and θ_{x_j} are present, where $\theta_{x_ix_j}$ and θ_{x_i} are the regression coefficients of the variables x_ix_j and x_i respectively [Bien et al., 2013, Lim and Hastie, 2015, Wu et al., 2009, Nakagawa et al., 2016, Chipman, 1996]. By assuming such low-dimensional structure, inference tasks such as parameter estimation and variable selection become more tractable statistically. However, sparsity constraints create computational difficulties. For example, finding the maximum-likelihood estimator (MLE) subject to $\|\theta\|_0 \leq s$ requires searching over $\Theta(p^{2s})$ active parameter subsets. To avoid the combinatorial issues resulting from an L_0 penalty, recent works [Bien et al., 2013, Lim and Hastie, 2015] have instead used L_1 penalties to encourage parameter sparsity for interaction models; L_1 penalties have a long history in high-dimensional linear regression [Chen et al., 1998, Candes and Tao, 2007, Tibshirani, 1994], Maximizing the likelihood with an added L_1 penalty is a convex problem. But each iteration of a state-of-the-art solver for methods given by Bien et al. [2013] and Lim and Hastie [2015] still takes $O(Np^2)$ time. To handle larger p, Wu et al. [2009], Nakagawa et al. [2016], Shah [2016] have proposed various pruning heuristics for finding locally optimal solutions. However, since these methods do not provide an exact solution to the optimization problem, any statistical guarantees (such as the statistical rate at which these estimators converge to the true parameter as a function of N and p) are weaker than those for exact methods.

L_1-based methods face a number of additional challenges: constructing valid confidence intervals, incorporating background knowledge, and controlling for the issue of multiple comparisons when testing many parameters for statistical significance. In many applications such as genome-wide association studies, controlling for multiplicity is critical to prevent wasting resources on false discoveries. Moreover, since $\dim(\Phi_2) = p(p+1)/2$, θ can be very high dimensional even when p is moderately large. Hence, there will typically be nontrivial uncertainty when attempting to estimate θ. Fortunately, hierarchical Bayesian methods have (1) a natural shrinkage or regularization effect such that multiple testing corrections are no longer necessary, (2) better statistical power than using multiple comparison correction terms such as Bonferroni [Gelman et al., 2012], and (3) naturally provide calibrated uncertainties. Bayesian methods can also incorporate expert information.

Though they offer desirable statistical properties, Bayesian approaches are computationally expensive. Previous efforts [Griffin and Brown, 2017, Chipman, 1996] have focused on developing *hierarchical sparsity priors* that promote strong hierarchy, analogous to the LASSO-based approaches [Bien et al., 2013, Lim and Hastie, 2015, Wu et al., 2009, Nakagawa et al., 2016]. But these methods do not address the computational intractability of inference for even moderate-dimensional problems.

We address the computational challenges of inference by developing the *kernel interaction trick* (Section 3.5), which allows us to access posterior marginals of θ without ever representing θ explicitly. Note that while some previous works have used a degree-two polynomial kernel to implicitly generate all pairwise interactions [Morota and Gianola, 2014, Weissbrod et al., 2016, Su et al., 2012], those works have focused on prediction or estimating the cumulative proportion of variance explained by interactions rather than our present focus on posterior inference.

3.3 Bayesian Models with Interactions

Our goal is to estimate and provide uncertainties for the parameter $\theta \in \mathbb{R}^{\dim(\Phi_2)}$. To take a Bayesian approach, we encode the state of knowledge before observing the data D in a *prior* $\pi_0(\theta)$. We express the *likelihood* as $\mathcal{L}(Y \mid \theta, X) = \prod_{n=1}^{N} \mathcal{L}(y^{(n)} \mid \theta, x^{(n)})$. Applying Bayes' theorem yields the *posterior* distribution $\pi(\theta \mid D) \propto \mathcal{L}(Y \mid \theta, X)\pi_0(\theta)$, which describes the state of knowledge about θ after observing the data D. For a function f of interest, we wish to compute the posterior expectation

$$\mathbb{E}_{\pi(\theta|D)}[f(\theta)] = \int f(\theta)\pi(\theta \mid D)d\theta. \tag{3.2}$$

Table 3.1: Per-iteration MCMC runtime and memory scaling of methods for sampling two-way interactions. NAIVE refers to explicitly factorizing $\Sigma_{N,\tau}$ to compute $p(D \mid \tau, \sigma^2)$, WOODBURY refers to using the Woodbury identity and matrix determinant lemma to compute $p(D \mid \tau, \sigma^2)$, and FULL refers to jointly sampling θ and τ. The third column provides the number of parameters sampled.

Method	Time	Memory	#
Our Method	$O(pN^2 + N^3)$	$O(pN + N^2)$	$O(p)$
Naive	$O(p^6 + p^4 N)$	$O(p^4 + p^2 N)$	$O(p)$
Woodbury	$O(p^2 N^2 + N^3)$	$O(p^2 N + N^2)$	$O(p)$
Full	$O(p^2 N)$	$O(p^2 N)$	$\Theta(p^2)$

Typically, $f(\theta) = \theta_j$ or $f(\theta) = \theta_j^2$, which together allow us to compute the posterior mean and variance of each θ_j.

Generative model. Going forward, we model θ as being drawn from a *Gaussian scale mixture* prior to encode desirable properties such as sparsity and strong hierarchy [cf. Griffin and Brown, 2017, Chipman, 1996, George and McCulloch, 1993, Carvalho et al., 2009, Piironen and Vehtari, 2017]. These priors have also been used beyond sparse Bayesian regression [cf. Hamdan et al., 2005, Wainwright and Simoncelli, 1999, Choy and Chan, 2003]. A Gaussian scale mixture is equivalent to assuming that there exists an auxiliary random variable $\tau \sim p(\tau)$ such that θ is conditionally Gaussian given τ. Let Σ_τ denote the covariance matrix for $p(\theta \mid \tau)$. Also, let σ^2 be the latent noise variance in the likelihood; since it is typically unknown, we treat it as random and put a prior on it as well. Hence, the full generative model can be written

$$\begin{aligned} \tau &\sim p(\tau) \\ \sigma^2 &\sim p(\sigma^2) \\ \theta \mid \tau &\sim \mathcal{N}(0, \Sigma_\tau) \\ y^{(n)} \mid x^{(n)}, \theta, \sigma^2 &\sim \mathcal{N}(\theta^T \Phi_2(x^{(n)}), \sigma^2). \end{aligned} \tag{3.3}$$

Computational challenges of inference. Again, our main goal is to tractably compute expectations of functions under the posterior $\pi(\theta \mid D) \propto \mathcal{L}(Y \mid \theta, X)\pi_0(\theta)$. Since there are $\Theta(p^2)$ parameter components, direct numerical integration over each of these components is feasible only when p is at most 3 or 4. As a result we turn to Monte Carlo integration. Two natural Monte Carlo estimators one might use to approximate $\mathbb{E}_{\pi(\theta|D)}[f(\theta)]$ are

1. $\frac{1}{T}\sum_{t=1}^{T} f(\theta^{(t)})$ with $\theta^{(t)} \overset{\text{iid}}{\sim} \pi(\theta \mid D)$ or

2. $\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\pi(\theta|D,\tau^{(t)})}[f(\theta)]$ with $\tau^{(t)} \overset{\text{iid}}{\sim} p(\tau \mid D)$.

For the first estimator, we can use Markov chain Monte Carlo (MCMC) techniques to sample each $\theta^{(t)}$ approximately independently from $\pi(\theta \mid D)$ since the posterior is available up to a multiplicative normalizing constant. Computing the prior $p(\theta)$, however, may be analytically intractable because it requires marginalizing out τ. We could instead additionally sample τ. To use gradient-based MCMC samplers, sampling τ would require computing the pdfs (and gradients) of the likelihood terms $\mathcal{L}(y^{(n)} \mid x^{(n)}, \theta, \sigma^2)$ and the prior terms $p(\theta \mid \tau)$ and $p(\tau)$. So the cost would be $O(p^2 N + \dim(\tau))$ time per iteration. Even for p moderately large, the $\Theta(p^2 + \dim(\tau))$ number of parameters might require many MCMC iterations to properly explore such a large space [MacKay, 1998, Pillai et al., 2012, Beskos et al., 2013]; see also Fig. 3-2 for an empirical demonstration.

To explore a smaller space, and hence potentially reduce the number of MCMC iterations required for the chains to mix, we might take the second approach: sampling from $p(\tau \mid D)$ by marginalizing out the high-dimensional parameter θ. Sampling each τ requires computing

$$p(D \mid \tau, \sigma^2) = \int p(D \mid \theta, \sigma^2) dp(\theta \mid \tau). \tag{3.4}$$

Since $p(\theta \mid \tau)$ is a multivariate Gaussian density function, $p(D \mid \tau, \sigma^2)$ equals

$$\frac{(1/\sqrt{2\pi\sigma^2})^N \det(2\pi\Sigma_{N,\tau})^{\frac{1}{2}} \exp\left(-\frac{1}{2\sigma^2} Y^T Y\right)}{\det(2\pi\Sigma_\tau)^{\frac{1}{2}} \exp\left(-\frac{1}{2\sigma^4} Y^T \Phi_2(X) \Sigma_{N,\tau} \Phi_2(X)^T Y\right)} \tag{3.5}$$

where $\Sigma_{N,\tau}^{-1} \coloneqq \Sigma_\tau^{-1} + \frac{1}{\sigma^2}\Phi_2(X)^T\Phi_2(X)$. Unfortunately, computing Eq. (3.5) naively takes prohibitive $O(p^6 + p^4 N)$ time – or $O(p^2 N^2 + N^3)$ time when using linear algebra identities; see Table 3.1 and Appendix B.6 for details.

3.4 The Kernel Interaction Sampler

Our *kernel interaction sampler* (KIS) provides a recipe for efficiently sampling from $p(\tau, \sigma^2 \mid D)$ using MCMC. Recall from the last section that the computational bottleneck for sampling τ was computing $p(D \mid \tau, \sigma^2)$, so we focus on that problem here. We achieve large computational gains (Table 3.1) by using the special model structure and a kernel trick to avoid the factorization of $\Sigma_{N,\tau}$ in Eq. (3.5). To that end, KIS has three main parts: (1) we re-parameterize the generative model given in Eq. (3.3) using a Gaussian process (GP); (2) we show how to cheaply compute the GP kernel; and (3) we show how these steps translate into computation of $p(D \mid \tau, \sigma^2)$ in time linear in p. In Appendix B.1 we extend to the case of higher-order interactions.

For the moment, suppose that we could construct a covariance function k_τ such that the generative model in Eq. (3.3) could be re-parameterized as:

$$\begin{aligned} \tau &\sim p(\tau) \\ g \mid \tau &\sim GP(0, k_\tau) \\ \sigma^2 &\sim p(\sigma^2) \\ y^{(n)} \mid g, x^{(n)}, \sigma^2 &\sim \mathcal{N}(g(x^{(n)}), \sigma^2), \end{aligned} \tag{3.6}$$

where $GP(0, k_\tau)$ denotes a Gaussian process (GP) with mean function zero. Defining the kernel matrix $(K_\tau)_{ij} := k_\tau(x^{(i)}, x^{(j)})$, we can conclude that [see Rasmussen and Williams, 2006, Eq. 2.30]

$$\log p(D \mid \tau, \sigma^2) = -\frac{1}{2} Y^T L^{-1} Y - \frac{1}{2} \log |L| - \frac{N}{2} \log 2\pi, \tag{3.7}$$

where L equals $K_\tau + \sigma^2 I_N$. Let T_k denote the time it takes to evaluate k_τ on a pair of points. The computational bottleneck of Eq. (5.10) is computing and factorizing K_τ, which take $O(N^2 T_k)$ and $O(N^3)$ time, respectively. Hence, as long as T_k is $O(p)$, we can compute $p(D \mid \tau, \sigma^2)$ in time *linear* in p. To achieve this scaling, we first show (in the next result) that *any* generative model in the form of Eq. (3.3) can be rewritten in the form of Eq. (3.6). We then show how k_τ can be evaluated in $O(p)$ time for the models of interest.

Proposition 3.4.1. *(Gaussian process representation) Let Y and $\tilde{Y}$ be response vectors generated according to the models in Eq.* (3.3) *and Eq.* (3.6) *respectively for design matrix $X \in \mathbb{R}^{N \times p}$. Let $k_\tau(x^{(i)}, x^{(j)}) = \Phi_2(x^{(i)})^\top \Sigma_\tau \Phi_2(x^{(j)})$. Then, $Y \mid X \overset{d}{=} \tilde{Y} \mid X$, where $\overset{d}{=}$ denotes equality in distribution. Moreover, for every draw $g \mid \tau \sim \mathcal{N}(0, k_\tau)$, there exists some $\theta \in \mathbb{R}^{\dim(\Phi_2)}$ such that $g(\cdot) = \theta^T \Phi_2(\cdot)$.*

The proof follows directly by considering the weight-space view of a GP [Rasmussen and Williams, 2006, Chapter 2]; see Appendix D.1 for details.

Next, we need to show that k_τ can be evaluated in $O(p)$ time for models of interest. This fact is not obvious; computing k_τ on a pair of points naively still requires explicitly computing the high-dimensional feature maps Φ_2 and prior covariance matrix Σ_τ. To compute k_τ efficiently, we rewrite it as a weighted sum of polynomial kernels of the form

$$k^c_{\text{poly},d}(x, \tilde{x}) := \left(x^T \tilde{x} + c\right)^d,$$

which each take $O(p)$ time to compute. Below we define *two-way interaction kernels* as particular linear combinations of these polynomial kernels. Then we provide a result motivating this class; namely, we show that any diagonal Σ_τ prior can be written as a two-way interaction kernel. Fortunately, to the best of our knowledge, all previous high-dimensional Bayesian regression models assume Σ_τ is diagonal [cf. Griffin and Brown, 2017, Chipman, 1996, George and McCulloch, 1993, Carvalho et al., 2009, Piironen and Vehtari, 2017]. Hence, this restriction on Σ_τ is mild.

Definition 3.4.2. (Two-way interaction kernel) We call the kernel k a *two-way interaction kernel* if for some choice of $M_1, M_2 \in \mathbb{N}$, $\alpha, \psi, \lambda^{(m)} \in \mathbb{R}_+^p$ $(m = 1, \ldots, M_1)$, $\nu^{(m)} \in \mathbb{R}_+$ $(m = 1, \ldots, M_2)$, $1 \leq i_m < j_m \leq p$ $(m = 1, \ldots, M_2)$, and $A \in \mathbb{R}$, the kernel $k(x, \tilde{x})$ is equal to

$$\begin{aligned}&\sum_{m=1}^{M_1} k^1_{poly,2}(\lambda^{(m)} \odot x, \lambda^{(m)} \odot \tilde{x}) + \sum_{m=1}^{M_2} \nu^{(m)} x_{i_m} x_{j_m} \tilde{x}_{i_m} \tilde{x}_{j_m} \\ &+ k^A_{poly,1}(\alpha \odot x, \alpha \odot \tilde{x}) + k^0_{poly,1}(\psi \odot x \odot x, \psi \odot \tilde{x} \odot \tilde{x}),\end{aligned}$$

where $\odot$ is the entrywise product.

Theorem 3.4.3 (1-to-1 correspondence with diagonal Σ_τ)**.** *Suppose k is a two-way interaction kernel. Then*

$$k(x, \tilde{x}) = \Phi_2(x)^\top S \Phi_2(\tilde{x}), \tag{3.8}$$

where the induced prior covariance matrix S is diagonal. The entries of S are given by

$$\begin{aligned}\operatorname{diag}(S)_{(i)} &= \alpha_i^2 + 2\sum_{m=1}^{M_1} \left[\lambda_i^{(m)}\right]^2 \\ \operatorname{diag}(S)_{(ij)} &= 2\sum_{m=1}^{M_1} \left[\lambda_i^{(m)} \lambda_j^{(m)}\right]^2 + \sum_{k: i_k = i, j_k = j} \nu^{(m)} \\ \operatorname{diag}(S)_{(ii)} &= \psi_i^2 + \sum_{m=1}^{M_1} \left[\lambda_i^{(m)}\right]^4 \\ \operatorname{diag}(S)_{(0)} &= M_1 + A,\end{aligned}$$

where $\operatorname{diag}(S)_{(i)}$, $\operatorname{diag}(S)_{(ij)}$, $\operatorname{diag}(S)_{(ii)}$, *and* $\operatorname{diag}(S)_{(0)}$ *denote the prior variances of the main effect θ_{x_i}, interaction effect $\theta_{x_i x_j}$, quadratic effect $\theta_{x_i^2}$, and intercept θ_0, respectively. Furthermore, for any diagonal covariance matrix $S \in \mathbb{R}^{\dim(\Phi_2) \times \dim(\Phi_2)}$, there exists a two-way interaction kernel that induces S as a prior covariance matrix.*

Theorem 3.4.3 (proof in Appendix B.2.2) and Proposition 3.4.1 imply that two-way interaction kernels induce a space of models in 1-to-1 correspondence with models in the form of Eq. (3.3) when Σ_τ is constrained to be diagonal. Since most models of practical interest have Σ_τ diagonal, we can readily construct the two-way interaction kernel corresponding to Σ_τ by solving the system of equations

$$\begin{aligned}\operatorname{diag}(S)_{(i)} &= \operatorname{diag}(\Sigma_\tau)_{(i)} \quad \operatorname{diag}(S)_{(ij)} = \operatorname{diag}(\Sigma_\tau)_{(ij)} \\ \operatorname{diag}(S)_{(ii)} &= \operatorname{diag}(\Sigma_\tau)_{(ii)} \quad \operatorname{diag}(S)_{(0)} = \operatorname{diag}(\Sigma_\tau)_{(0)}\end{aligned} \tag{3.9}$$

Each of the $M_1 + 2$ polynomial kernels takes $O(p)$ time to compute, and each of the M_2 product terms takes $O(1)$ time. Therefore, we want to select M_1 and M_2 small so that k_τ can be computed quickly. Since there are more degrees of freedom (i.e., free variables) available to solve Eq. (3.9) as M_1 and M_2 increase, eventually a solution will exist as we show in Appendix B.2.2. But Theorem 3.4.3 does not tell us how large

M_1 and M_2 have to be for an arbitrary model. In Appendix B.3, we solve Eq. (3.9) for a variety of models of practical interest and show that in these cases, M_1 and M_2 can be set very small (between one and three). Thus k_τ can be computed in $O(p)$ time, and so the kernel matrix K_τ can be computed in $O(N^2p)$ time. Finally, then, we may compute the likelihood $p(D \mid \tau, \sigma^2)$ in $O(N^2p + N^3)$ time.

3.5 The Kernel Interaction Trick: Recovering Posterior Marginals

Even if we are able to sample τ much faster using KIS, the problem of computing $\mathbb{E}_{p(\theta|D,\tau)}[f(\theta)]$ remains unresolved. In this section, we show that, given K_τ, any such expectation can be recovered in $O(1)$ time by evaluating the GP posterior at certain test points.

To provide the main intuition for our solution, suppose we would like to compute the posterior mean of the main effect θ_{x_i}. Let $e_i \in \mathbb{R}^p$ denote the ith unit vector. Since $g = \theta^T \Phi_2$ by Proposition 3.4.1, we have

$$\begin{aligned} g(e_i) &= \theta_{x_i} + \theta_{x_i^2} \\ g(-e_i) &= -\theta_{x_i} + \theta_{x_i^2} \\ \frac{g(e_i) - g(-e_i)}{2} &= \theta_{x_i}. \end{aligned} \tag{3.10}$$

Since g is a Gaussian process, the distribution of $Z_g \coloneqq (g(e_i), g(-e_i)) \mid D, \tau$ is multivariate Gaussian and can be computed in closed form by appropriate matrix multiplications of the kernel matrix K_τ; see Theorem 3.5.1 below for details. Then, by consulting Eq. (3.10), one can recover $\theta_{x_i} \mid D, \tau$ as the linear combination $[1/2, -1/2]^T Z_g \mid D, \tau$, which is univariate Gaussian. While we have focused on a particular instance here, this example provides the main insight for the general formula to compute $\mathbb{E}_{p(\theta|D,\tau)}[f(\theta)]$ from K_τ.

Theorem 3.5.1. *(The kernel interaction trick) Let $H_\tau \coloneqq (K_\tau + \sigma^2 I_N)^{-1}$ and*

$$A_{ij} \coloneqq [e_i, -e_i, e_j, e_i + e_j]^T \in \mathbb{R}^{4\times p}.$$

Let $K_\tau(A_{ij}, X) = K_\tau(X, A_{ij})^T$ be the $4 \times N$ matrix formed by taking the kernel between each row of A_{ij} with each row of X. For a row-vector $a \in \mathbb{R}^4$, define the scalars $\mu_a \coloneqq a K_\tau(A_{ij}, X) H_\tau Y$ and

$$\sigma_a^2 \coloneqq a \left[K_\tau(A_{ij}, A_{ij}) - K_\tau(A_{ij}, X) H_\tau K_\tau(X, A_{ij})\right] a^T.$$

Then the distributions of $\theta_{x_i} \mid \tau, D$, $\theta_{x_i x_j} \mid \tau, D$, and $\theta_{x_i^2} \mid \tau, D$ are given by $\mathcal{N}(\mu_a, \sigma_a^2)$ with, respectively, $a = (1/2, -1/2, 0, 0)$, $a = (-1/2, 1/2, -1, 1)$, and $a = (1/2, 1/2, 0, 0)$.

Corollary 3.5.2. *Given K_τ, the distributions of θ_{x_i}, $\theta_{x_i x_j}$, and $\theta_{x_i^2}$ take $O(1)$ additional time and memory to compute.*

We prove Theorem 3.5.1 and Corollary 3.5.2 in Appendix D.1. In Appendix B.2.5, we generalize Theorem 3.5.1 by showing how to obtain the joint posterior distribution of any subset of parameters contained in θ. Hence, we can compute $\mathbb{E}_{p(\theta|D,\tau)}[f(\theta)]$ for an arbitrary f using the kernel interaction trick.

Note that if we would like to obtain the posterior mean of all $\Theta(p^2)$ parameters, then clearly a linear time algorithm in p is impossible. Instead, we can adopt a lazy evaluation strategy where we compute the posterior of one of the $\Theta(p^2)$ parameters only when it is needed. This approach is effective in the many applications where we do not need to look at all the interactions. In particular, we might first find the top k main effects. After selecting these variables, we could examine their interactions. The number of interactions among the main effects (which is $\Theta(k^2)$) is much smaller than the total number of possible interactions (which is $\Theta(p^2)$) if $k \ll p$. Such a strategy is natural if we believe that θ satisfies the (commonly assumed) strong hierarchy restriction.

3.6 SKIM: <u>S</u>parse <u>K</u>ernel <u>I</u>nteraction <u>M</u>odel

To demonstrate the utility and efficacy of the kernel interaction sampler and kernel interaction trick, we choose a particular model that we call the *sparse kernel interaction model (SKIM).* In what follows, we first detail SKIM, which we will see promotes sparsity and strong hierarchy. Then, by observing that SKIM is a special case of the general model in Eq. (3.3), we can show that SKIM induces a two-way interaction kernel via Theorem 3.4.3 and Eq. (3.9). We will see that this kernel has only 3 components and thus takes only $O(p)$ time to evaluate. By Corollary B.2.3, we can compute the distribution of interaction terms from SKIM in $O(1)$ time once we have computed the kernel matrix. Hence, the final computation time for discovering main effects and interaction effects with SKIM will be $O(N^2p + N^3)$ by the discussion at the end of Section 3.5.

SKIM[2] is given in full detail, together with discussion of hyperparameter selection and intepretation, in Appendix B.4.1. It is a particular instance of a general class of hierarchical sparsity priors [cf. Griffin and Brown, 2017, Chipman, 1996, George and McCulloch, 1993] that have the following form:

$$\begin{gathered}
\kappa \sim p(\kappa) \quad \eta \sim p(\eta) \quad c^2 \sim p(c^2) \\
\theta_{x_i} \mid \kappa, \eta \sim \mathcal{N}(0, \eta_1^2 \kappa_i^2) \\
\theta_{x_i x_j} \mid \kappa, \eta \sim \mathcal{N}(0, \eta_2^2 \kappa_i^2 \kappa_j^2) \\
\theta_{x_i^2} \mid \kappa, \eta \sim \mathcal{N}(0, \eta_3^2 \kappa_i^4) \\
\theta_0 \mid c^2 \sim \mathcal{N}(0, c^2),
\end{gathered} \tag{3.11}$$

where θ_0 is the intercept term and every η_i or κ_j is a scalar.

We next show that any prior in the form of Eq. (3.11) induces a $O(p)$ two-way interaction kernel. The proof is in Appendix B.2.6.

[2]See `https://github.com/agrawalraj/skim` for the code.

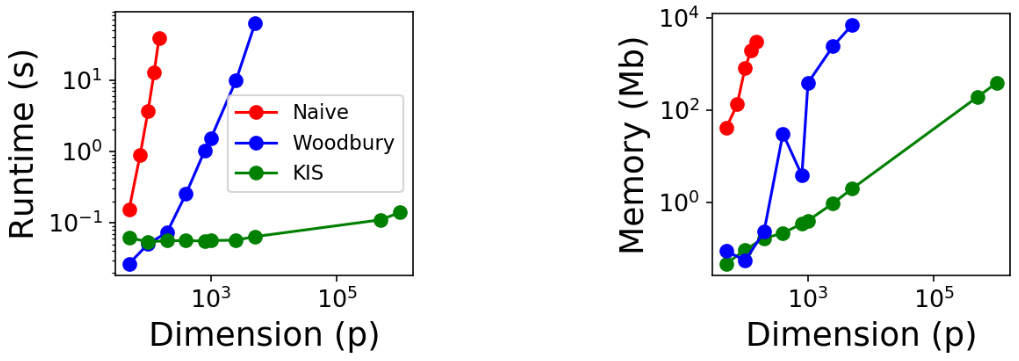


Figure 3-1: Empirical evaluation of (a) time and (b) memory scaling with dimension of marginal likelihood computation. Woodbury and Naive refer to the baselines in Section 5.2.

Proposition 3.6.1. *Taking* $\tau := (\eta, \kappa, c^2)$*, the generative model in Eq.* (3.11) *is equivalent to using the following kernel in Eq.* (3.6)*:*

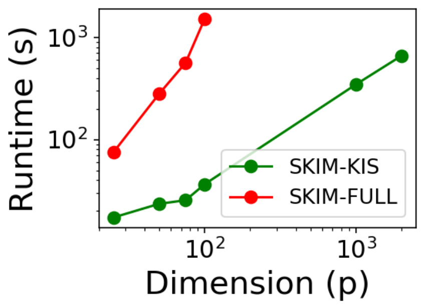
$$\begin{aligned} k_\tau(x, \tilde{x}) = \frac{\eta_2^2}{2} k^1_{poly,2}(\kappa \odot x, \kappa \odot \tilde{x}) \\ (\eta_3^2 - \frac{\eta_2^2}{2}) k^0_{poly,1}(\kappa \odot x \odot x, \kappa \odot \tilde{x} \odot \tilde{x}) \\ + \left(\eta_1^2 - \eta_2^2\right) k^0_{poly,1}(\kappa \odot x, \kappa \odot \tilde{x}) + c^2 - \frac{\eta_2^2}{2}. \end{aligned}$$

3.7 Experiments

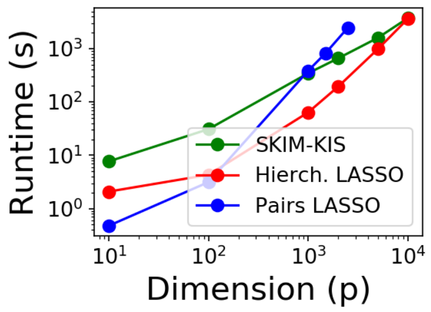
Time and memory cost versus Bayesian baselines. We first assess the computational advantages of our kernel interaction sampler (KIS) by comparing it with each baseline Bayesian method in Table 3.1. We start by profiling the time and memory cost of computing $p(D \mid \tau, \sigma^2)$, which we have seen is a computational bottleneck for sampler option 2 in Section 5.2. In Fig. 3-1, we depict the time and memory cost of $p(D \mid \tau, \sigma^2)$ computation for conjugate linear regression with an isotropic Gaussian prior on synthetic datasets with $N = 50$. We vary p but not N because we are interested primarily in the high-dimensional case when p is large relative to N. Fig. 3-1 shows that KIS yields orders-of-magnitude speed and memory improvements over the baseline methods for computing $p(D \mid \tau, \sigma^2)$.

We next compare inference for SKIM using KIS, which marginalizes out θ and samples τ, to jointly sampling (θ, τ) (denoted FULL).[3] We implemented KIS and FULL in `Stan` Carpenter et al. [2019] and used the NUTS algorithm Hoffman and Gelman [2014] for sampling (4 chains with 1,000 iterations per chain). As shown in Fig. 3-2(a), KIS is orders of magnitude faster even for lower values of p. In Section 5.2 we remarked that since FULL explores a much higher-dimensional space, there might

[3]See the discussion of sampler option 1 in Section 5.2.



(a) NUTS Runtimes



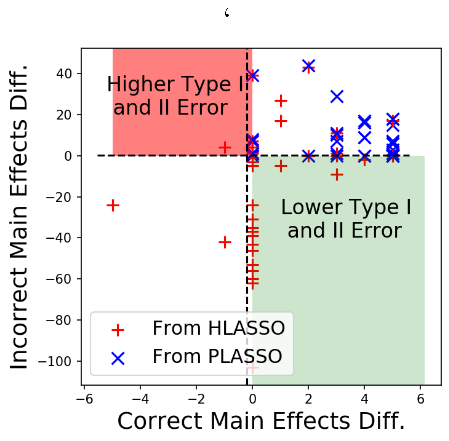
(b) LASSO runtime comparisons

Figure 3-2: The left-hand figure indicates the time to complete four parallel chains of 1000 iterations of NUTS for the SKIM model proposed in Section 3.6 using KIS (denoted as SKIM-KIS) and FULL. For each point, KIS had $\hat{R} < 1.05$ while FULL always had $\hat{R} > 1.05$. The right-hand figure compares the runtime of inference for SKIM-KIS versus fitting LASSO-based methods.

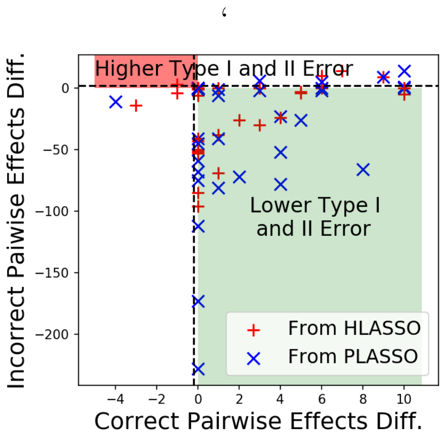
be issues with mixing. To explore this possibility empirically, we check the Gelman–Rubin statistic ($\hat{R}$) values of the output from both KIS and FULL. We found that, for FULL, the $\hat{R}$ values were greater than 1.05, with some reaching as high as 1.5 (indicating poor mixing), while for KIS all $\hat{R}$ values were less than 1.05 (suggesting good mixing).

Comparison to LASSO: synthetic data. Having demonstrated the considerable computational savings over baseline Bayesian approaches, we next demonstrate the advantage of our method over frequentist approaches such as the LASSO. In particular, we consider the common case when the true high-dimensional parameter θ is assumed to be sparse and satisfies the requirement of strong hierarchy. To the best of our knowledge, there has not been an extensive empirical comparison between sparse Bayesian interaction models and sparse frequentist interaction models. The likely reason is that each MCMC iteration for sampling τ takes $O(N^2p^2 + N^3)$ time using the Woodbury matrix method. The per-iteration cost of the iterative optimization solver for the LASSO and the hierarchical LASSO, on the other hand, is $O(Np^2)$, which is much faster when N is even moderately large. Fortunately, SKIM admits a cheap-to-compute kernel function such that each MCMC iteration takes $O(N^2p + N^3)$ time, which is *faster* than the LASSO-style approaches in cases when p is large relative to N.

We benchmark SKIM against generating all pairwise interactions and running the LASSO (denoted pairs LASSO) and the hierarchical LASSO Lim and Hastie [2015], which constrains the fitted parameters to satisfy strong hierarchy. We generate 36 different synthetic datasets, which differ in the number of observations, dimension, and signal-to-noise ratio. The covariates X are drawn from $\mathcal{N}(0, \lambda^2 I_p)$ for different choices of λ. Here, λ controls the signal-to-noise ratio; when λ is larger, the signal



(a) Main effects differences

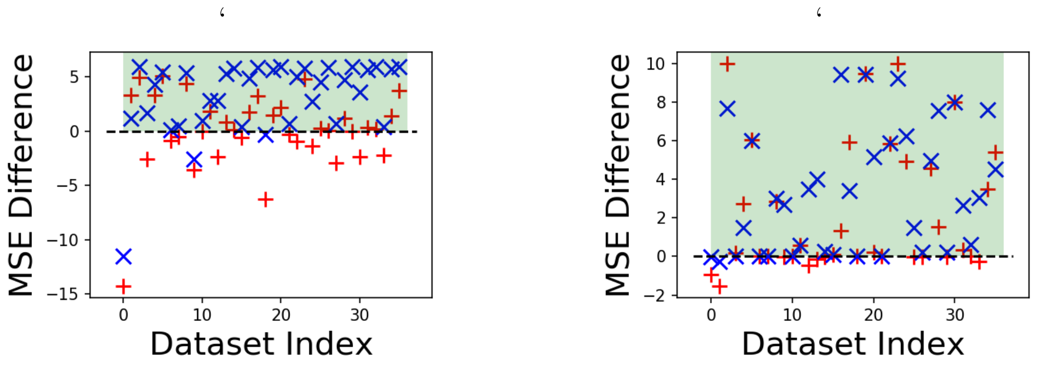


(b) Pairwise effects differences

Figure 3-3: Variable selection performance of each method for the 36 synthetic datasets. Each point in each plot indicates one of these datasets for a particular method. The green regions in the second and last plot indicate where our method in strictly better than the other two in terms of variable selection, while the red region indicates the datasets for which our method is strictly worse. In the first and last figures, better performance occurs when moving right and/or down.

is stronger. We consider $N \in \{50, 100, 200\}$ observations, $p \in \{50, 100, 200, 500\}$ dimensions (which translates into between roughly 1.25×10^3 and 1.25×10^5 total interaction parameters), and $\lambda \in \{1, 2, 5\}$. In each dataset, we select five variables (and their pairwise interactions) to affect y, and we allow the rest of the variables to lead to spurious correlations with the response y. We set the magnitudes of all non-zero effects to 1. Finally, $y \mid x, \theta^* \sim \mathcal{N}(0, \sigma^2)$, where the noise variance σ^2 equals the largest λ^2 value, namely 25, to mimic the realistic case when the noise variance is large relative to the signal. We compare each method in terms of variable selection quality and mean-squared error (MSE) between the fitted and true parameter. For variable selection, we select a parameter only if the posterior mean of that parameter is farther than 2.59 times its (average) posterior standard deviation from zero; see Appendix B.5 for details. For the hierarchical LASSO and pairs LASSO, the variables selected are those with non-zero coefficients, and we use 5-fold cross-validation to find the strength of the L_1 penalties. We fit the hierarchical LASSO using the `glinternet` package in `R` and pairs LASSO using `sklearn` in `python`. We implemented KIS is `Stan` (4 chains with 1,000 iterations each). The $\hat{R}$ values for each dataset were less than 1.05.

First, we examine how well each method selects main effects and pairwise effects. Each point in Fig. 3-3(a) shows the number of main effects selected and number of incorrect main effects selected for a given synthetic dataset. In this plot, it is clear that our method has better false discovery rate (FDR) control over the other two methods on average. Fig. 3-3(c) shows the FDR performance for pairwise effects. To compare the methods at the dataset level, in Fig. 3-3(b,d) we consider the difference in the number of correct and incorrect main effects selected by our method and the LASSO



(a) MSE difference (main) (b) MSE difference (pairwise)

Figure 3-4: Each red cross denotes the difference in MSE of the hierarchical LASSO and KIS from the true main effects (left) and pairwise effects (right) for a given synthetic dataset. When the MSE difference is larger than 0 (i.e., the green shaded region), then our method is closer to the true effect sizes in terms of Euclidean distance. Similarly, each blue x equals the difference in MSE of all-pairs LASSO and our method.

methods for each dataset. The green shaded regions indicate the datasets for which our method simultaneously selects more correct main effects and has fewer incorrect main effects, i.e., is strictly better than the other two methods for any variable selection metric. Finally, in Fig. 3-4 we look at the difference in MSE to θ^*, broken down in terms of the error for estimating main and pairwise effects. Again, we see for the great majority of the datasets, KIS outperforms the LASSO based approaches. In Fig. 3-2(b) we see that SKIM-KIS has competitive runtimes relative to pairs LASSO and hierarchical LASSO.

Comparison to LASSO: synthetic data, real covariates. To understand the impact of the geometry of the covariates on performance, we took the Residential Building Data Set from the UCI Machine Learning Repository and simulated responses as in our previous synthetic experimental setup. In particular, we randomly chose 5 variables and their 10 pairwise interactions to have non-zero effects. In this case, the covariates are highly correlated (the first 20 out of 105 principal components capture over 99% of the variance in the data). In Table 3.2, we see that SKIM significantly outperforms the LASSO-based methods for recovering main and pairwise effects.

Comparison to LASSO: cars miles per gallon dataset. We conclude by comparing the methods on the Auto MPG dataset, from the UCI Machine Learning Repository, which contains $N = 398$ samples and $p = 8$ variables. We consider only the 6 numerical variables (cylinders, displacement, horsepower, weight, acceleration, model year) and standardize the data by subtracting the mean and dividing by the standard deviation. To compare the methods, we first fit SKIM and the LASSO-based methods (via 5-fold cross-validation) on these 6 features. Our method selects three main effects (weight, horsepower, acceleration) and one interaction (weight $\times$

Table 3.2: Building dataset results. MAIN (PAIR) MSE refers to total error in estimating main (pairwise) effects. The main and pairwise MSE added together yield the total MSE. The second and fourth columns show (# of effects correctly selected) : (# of incorrect effects selected) for main and pairwise effects, respectively. Larger green values are better while larger purple values are worse.

Method	Main MSE	# Main	Pair MSE	# Pair
SKIM	0.1	**3 : 0**	7.0	**3 : 0**
PLASSO	5.0	**2 : 5**	9.3	**3 : 21**
HLASSO	1.5	**3 : 19**	7.8	**3 : 18**

Table 3.3: Auto MPG dataset results. Each column represents the (# of original effects selected) : (# of fake effects selected). A selected main (pairwise) effect is an "original" effect if it corresponds to one of the original 6 features (15 interactions). Main100 (Pairwise100) and Main200 (Pairwise200) denote when 100 and 200 random noise covariates are added to the original 6 features, respectively. Larger purple values are worse. Higher green values are not necessarily better since there are no ground truth interactions.

Method	Main100	Main200	Pair100	Pair200
SKIM	**3 : 0**	**3 : 0**	**1 : 0**	**1 : 0**
PLASSO	**4 : 1**	**4 : 0**	**4 : 99**	**2 : 78**
HLASSO	**5 : 4**	**6 : 46**	**5 : 2**	**4 : 38**

horsepower). The hierarchical LASSO selects all six main effects and 8 out of the 15 possible pairwise interactions. Pairs LASSO selects 5 main effects and 8 interactions.

Since there is no ground truth, and all of the main and pairwise interactions could a priori affect miles per gallon, it is difficult to compare the methods. To better assess the methods, we instead append random noise covariates and refit each model. In particular, we draw additional covariates from a $\mathcal{N}(0, I_m)$, for $m = 100, 200$ and add these noise variables to the original 6 features. The total number of main and pairwise regression coefficients grows to 5,671 and 21,321 for $m = 100, 200$ respectively, making the regression task very high-dimensional. The results are summarized in Table 3.3. All methods are able to pick up some main effects and pairwise effects from the original dataset. Beyond that observation, we cannot compare which main and interaction effects from the original data are real. However, we do know that all noise effects are fake. We see that even with more noise directions, our method selects the same main effects and pairwise effects as the noiseless covariate case; that is, it does not pick up any fake effects. The two LASSO-based methods, on the other hand, incorrectly select many noise variables as interactions.

Conclusion. Through our kernel interaction sampler we have demonstrated that Bayesian interaction models can offer both competitive computational scaling relative to LASSO-based methods and improved Type I and II error rates. While our method runs in time linear in p per iteration, the cubic dependence on N still makes inference computationally challenging. Fortunately, there is a wide GP literature that deals precisely with reducing this cubic timing dependence through inducing points [Titsias, 2009, Quiñonero Candela and Rasmussen, 2005] or novel conjugate-gradient techniques [Gardner et al., 2018]. An interesting future direction will be to empirically and theoretically understand the statistical penalty of using these inducing point methods to scale SKIM to the setting of both large N *and* large p.

Chapter 4

The SKIM-FA Kernel: High-Dimensional Variable Selection and Non-Linear Interaction Discovery in Linear Time

Abstract

Many scientific problems require identifying a small set of covariates that are associated with a target response and estimating their effects. Often, these effects are non-linear and include interactions, so linear and additive methods can lead to poor estimation and variable selection. The Bayesian framework makes it straightforward to simultaneously express sparsity, non-linearity, and interactions in a hierarchical model. But, as for the few other methods that handle this trifecta, inference is computationally intractable — with runtime at least quadratic in the number of covariates, and often worse. In the present work, we solve this computational bottleneck. We first show that suitable Bayesian models can be represented as Gaussian processes (GPs). We then demonstrate how a kernel trick can reduce computation with these GPs to $O(\#\text{ covariates})$ time for both variable selection and estimation. Our resulting fit corresponds to a sparse orthogonal decomposition of the regression function in a Hilbert space (i.e., a functional ANOVA decomposition), where interaction effects represent all variation that cannot be explained by lower-order effects. On a variety of synthetic and real datasets, our approach outperforms existing methods used for large, high-dimensional datasets while remaining competitive (or being orders of magnitude faster) in runtime.

4.1 Introduction

Many scientific and decision-making tasks require learning complex relationships between a set of p covariates and target response from $N \ll p$ observed datapoints. For example, in genomics and precision medicine, researchers would like to identify a small set of genetic and environmental factors (out of the potentially thousands or millions) associated with diseases and quantify their effects [Maher, 2008, Aschard, 2016, Slim et al., 2018, Greene et al., 2010]. Estimating these effects can be challenging, however, without sufficiently flexible models. Blood sugar levels, for example, could depend non-linearly on an individual's age; we might expect that younger people have a lower chance of developing high blood sugar levels than older people. In other instances, effects can be challenging to estimate due to multiplicative interactions. A particular drug could help individuals with certain genetic characteristics but harm others. To learn such nuances in our data, we need statistical methods that can model non-linear and interaction effects. We also need computationally efficient methods that can scale to large p settings. Unfortunately, as we detail below, existing methods suffer in at least one of these three categories.

Sparse linear regression methods (e.g., the Lasso) are typically fast but do not have the flexibility to learn non-linear or interaction effects [Chen et al., 1998, Candes and Tao, 2007, Nakagawa et al., 2016]. SpAM extends the Lasso to model non-linear effects but assumes additive effects [Liu et al., 2008]. Conversely, the hierarchical Lasso models interactions but assumes linearity, and its runtime scales quadratically with dimension [Bien et al., 2013]. Recently, Agrawal et al. [2019d] developed a kernel trick to learn interactions in time linear in dimension but this method assumes linear effects. Black-box approaches, such as neural networks and random forests, learn interactions and non-linear effects for the purposes of prediction. However, it is not clear how to actually access the effects from the fitted prediction model.

The *hierarchical functional ANOVA* [Stone, 1994], which includes many of the models described above as special cases, provides a powerful framework to jointly model interactions and non-linear effects. Unfortunately, existing functional ANOVA methods, which are primarily kernel regression based, do not scale well with dimension [Gu and Wahba, 1993, Lin and Zhang, 2006, Gunn and Kandola, 2004]; these methods use kernels that take $O(p^Q)$ time to evaluate, where Q equals the size of the highest order interaction. Hence, running kernel ridge regression for inference takes $O(p^Q N^2 + N^3)$ time.

Contributions. We propose SKIM-FA kernels to model non-linear and interaction effects. We show how to compute SKIM-FA kernels in $O(pQ)$ time by exploiting special low-dimensional structure. We motivate this structure from the perspective of hierarchical Bayesian modeling. Then, we use equivalences between kernel ridge regression, Gaussian process, and conjugate Bayesian linear regression to develop our efficient inference procedure.

Outline. The rest of the paper is outlined as follows. We start by describing how to model non-linear interaction effects and encode sparsity using hierarchical Bayesian modeling in Section 4.2. In Section 4.3, we develop two kernel tricks to perform inference more efficiently when the covariates are independent. Then, we extend our procedure to the general covariate case in Section 4.4. We defer implementation details of our final algorithm to Section 4.5. We conclude by discussing related work in Section 4.6 and benchmarking our method against other methods often used to model high-dimensional data in Section 6.5.

4.2 A framework for non-linear interactions and sparsity

4.2.1 Problem Statement

Suppose we collect data $D = \{(x^{(n)}, y^{(n)})\}_{n=1}^{N}$ with covariates $x^{(n)} \in \mathbb{R}^p$ and continuous scalar responses $y^{(n)} = f^*(x^{(n)}) + \epsilon^{(n)}$, where $\epsilon^{(n)} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2_{\text{noise}})$, $x^{(n)} \overset{\text{iid}}{\sim} \mu$, and $f^* \in \mathcal{H}$. We would like to identify what covariates f^* depends on (i.e., perform variable selection) and recover interactions effects using only noisy realizations of f^*. To perform such inference, we use penalized regression:

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \sum_{n=1}^{N} \mathcal{L}(y^{(n)}, f(x^{(n)})) + J(f), \tag{4.1}$$

where $\mathcal{L}(\cdot, \cdot)$ and $J(f)$ denote some loss function and penalty on model complexity, respectively. This paper focuses on four subproblems resulting from Eq. (4.1): (P1) picking $\mathcal{H}$ to model interactions (P2) selecting $\mathcal{L}(\cdot, \cdot)$ and $J(f)$ to induce sparsity, (P3) tractably solving Eq. (4.1) for our choice of sparsity-inducing $J(f)$, and (P4) efficiently reporting effects in $\hat{f}$.

4.2.2 Our contributions: an overview

We describe, at a high-level, our solution to subproblems P1 through P4, and what parts of our solution are new.

P1 - Constructing $\mathcal{H}$. Our construction of $\mathcal{H}$ in Section 4.2.3 is based on Huang [1998]. We use the hierarchical functional ANOVA introduced in Stone [1994] to make recovering interaction effects a well-defined inference task (i.e., statistically identifiable).

P2 - Selecting the loss and penalty. We select the loss and penalty from a hierarchical Bayesian modeling point-of-view in Section 4.2.4, where the loss corresponds to the negative log-likelihood of the data, and the penalty $J(f)$ corresponds to the negative log prior of f. Existing sparse Bayesian interaction methods do not work at our level of generality. Nevertheless, our proposed class of priors are heavily influenced by existing sparse Bayesian interaction models. Our solutions to P3 and P4 are our core contributions.

P3 - Solving Eq. (4.1). We solve Eq. (4.1) in time linear in p by using two kernel tricks. The first kernel trick, described in Section 4.3, is based on the seminal smoothing spline ANOVA (SS-ANOVA) work by Gu and Wahba [1993]. To make this connection to SS-ANOVA, we show that there exists a duality between the class of hierarchical models proposed in Section 4.2.4 and reproducing kernel Hilbert spaces induced by what we call *model selection kernels*. Model selection kernels generalize the kernels used in Gu and Wahba [1993] by removing the requirement that all covariates be independent. Our second kernel trick is unique to our new model, the *sparse kernel interaction model for functional ANOVA* (SKIM-FA), which belongs to the class of models in Section 4.2.4.

P4 - Reporting effects. For the case of independent covariates, we report effects using the procedure in Gu and Wahba [1993]. Our new contribution, provided in Section 4.4, is developing an efficient algorithm to report effects for the non-independent case.

4.2.3 Interactions and identifiability for nonlinear functions

To model interactions of order up to Q (i.e., effects that only depend on at most Q unique covariates), we "glue" function spaces of different interaction orders together to construct $\mathcal{H}$. Then, we use the hierarchical functional ANOVA to make inference over $\mathcal{H}$ well-defined (i.e., so that each function in $\mathcal{H}$ has a unique expansion).

Modeling Interactions. Let $\mathcal{H} = \mathcal{H}_Q \coloneqq \bigoplus_{V:|V|\leq Q} \mathcal{H}_V$, where $\mathcal{H}_V$ belongs to the space of all square-integrable functions of x_V (with respect to the probability measure μ) and $V \subset [p] \coloneqq \{1, \cdots, p\}$. Then,

$$\begin{aligned}\bigoplus_{V:|V|\leq Q} \mathcal{H}_V &= \{f : f = \sum_{V:|V|\leq Q} f_V(x_V),\ f_V \in \mathcal{H}_V\} \\ &= \{f : f = f_\emptyset + \sum_{i=1}^{p} f_{\{i\}}(x_i) + \sum_{i<j}^{p} f_{\{i,j\}}(x_i, x_j) + \cdots + f_{\{1,\cdots,\ p\}}(x_1, \cdots, x_p)\},\end{aligned} \tag{4.2}$$

where $f_\emptyset$ belongs to the space of constant functions $\mathcal{H}_\emptyset = \{\theta : \theta \in \mathbb{R}\}$. Similar to additive models, $f_{\{i\}}(x_i)$ has the interpretation as the main or marginal effect of covariate x_i on y. Similarly, $f_{\{i,j\}}(x_i, x_j)$ has the interpretation as the two-way or pairwise effect of x_i and x_j on y. Unfortunately, the components in Eq. (4.2) are not identifiable without further constraints. For example, if $f^*(x) = f_1(x_1) + f_2(x_2) + f_{12}(x_1, x_2)$, then f^* also decomposes as $f_1(x_1) + [f_2(x_2) + 5] + [f_{12}(x_1, x_2) - 5]$.

Identifiability with the Functional ANOVA. To resolve identifiability issues, we construct a smaller space of functions $\mathcal{H}_V^o \subset \mathcal{H}_V$, where $\mathcal{H}_V^o$ only includes functions whose variation cannot be explained by lower-order effects of x_V:

$$\mathcal{H}_V^o = \{f_V \in \mathcal{H}_V : \forall A \subsetneq V,\ \forall f_A \in \mathcal{H}_A,\ \langle f_V, f_A\rangle_\mu = 0\}, \tag{4.3}$$

where $\langle \cdot, \cdot \rangle_\mu$ measures similarity between functions through their covariance. That is, for $f_A \in \mathcal{H}_A$ and $f_B \in \mathcal{H}_B$, $\langle f_A, f_B\rangle_\mu = \mathbb{E}_{x\sim\mu}[f_A(x_A) f_B(x_B)]$.

Theorem 4.2.1. *[Stone, 1994, Huang, 1998] Suppose $f \in \mathcal{H}_Q$ and μ is absolutely continuous with respect to Lebesgue measure. Further, suppose that the domain of* *functions in $\mathcal{H}_Q$ range over a compact set $\mathcal{X}$ of $\mathbb{R}^p$. Then, there exists (μ-almost everywhere) unique functions $f_V \in \mathcal{H}_V^o$ such that $f = \sum_{V:|V|\leq Q} f_V$.*

Definition 4.2.2. Suppose $f = \sum_{V:|V|\leq Q} f_V$ where $f_V \in \mathcal{H}_V^o$. Then, $\sum_{V:|V|\leq Q} f_V$ is called the *functional ANOVA decomposition* of f with respect to μ.

By the orthogonality constraints in Eq. (4.3), $f_{\{i,j\}}(x_i, x_j)$ in the functional ANOVA decomposition, for example, represents the variation that cannot be explained by 1D functions of x_i and x_j and an intercept. When the covariates are independent, then the signal variance further decomposes as

$$\operatorname{var}(f) = \operatorname{var}(f_{\{\emptyset\}}) + \sum_i \operatorname{var}(f_{\{i\}}) + \sum_{i,j} \operatorname{var}(f_{\{i,j\}}) + \cdots \operatorname{var}(f_{\{1,2,\cdots,p\}}(x_1, \cdots, x_p)), \tag{4.4}$$

where $\operatorname{var}(f) = \langle f, f \rangle_\mu$. Hence, Eq. (4.4) allows us to *analyze* how the *variance* of the *function* is distributed across the interactions of different orders. Hence, the name functional analysis of variance or ANOVA.

4.2.4 How to achieve sparsity for nonlinear functions

We motivate our choice of loss and penalty in Eq. (4.1) from a Bayesian point-of-view. That is, we view $\mathcal{L}(\cdot, \cdot)$ as the negative log-likelihood function, $J(f)$ as the negative log prior on f, and $\hat{f}$ as the maximum a priori (MAP) estimate under our proposed Bayesian model.

Our Loss. Since the noise terms are Gaussian, the log-likelihood of the data given f is Gaussian. Hence, we pick $\mathcal{L}$ to equal squared-error loss.

Our Penalty. Since we are primarily interested in the case when f^* is sparse, i.e., when many of the effects f_V equal zero, we would like $J(f)$ to exploit such sparsity. To this end, we take a basis expansion of each component space, and then place a sparsity prior on the basis coefficients. We assume that for all $V \subset [p]$ and $1 \leq |V| \leq Q$, there exists a $B_V \in \mathbb{N} \cup \{\infty\}$ and feature map $\Phi_V : \mathbb{R}^{|V|} \mapsto \mathbb{R}^{B_V}$ such that the components of Φ_V form a basis of $\mathcal{H}_V^o$. Then, for any $f_V \in \mathcal{H}_V^o$, there exists $\Theta_V \in \mathbb{R}^{B_V}$ such that $f_V(x_V) = \Theta_V^T \Phi_V(x_V)$. Hence, if we can estimate Θ_V, we can estimate the functional ANOVA decomposition of ${}^*f$ by Theorem 4.2.1.

To obtain a MAP estimate of Θ_V, we draw each $\Theta_V \sim \mathcal{N}(0, \theta_V \cdot I_{B_V} \times I_{B_V})$, where θ_V is a non-negative auxiliary parameter drawn from a sparsity prior (e.g., a Laplace prior); see Section 4.5 for our particular choice of sparsity prior. If $\theta := \{\theta_V\}$ is sparse, then we claim that $\{f_V\}$ is sparse. To understand why, suppose $\theta_V = 0$. Then, the prior variance of f_V equals 0. Hence, f_V will equal 0.

Since Φ_V is a basis of $\mathcal{H}_V^o$ and our prior on Θ_V has full support on $\mathbb{R}^{B_V}$, our choice of likelihood and prior allows us to model any $f \in \mathcal{H}_Q$. We summarize our full hierarchical Bayesian model below:

$$\begin{aligned} \Theta_V \mid \theta_V &\sim \mathcal{N}(0, \theta_V \cdot I_{B_V} \times I_{B_V}),\ V \subset [p],\ |V| \leq Q, \theta_V > 0 \\ f &= \sum_{V:|V|\leq Q} \Theta_V^T \Phi_V(\cdot) \\ y^{(n)} \mid x^{(n)}, \Theta, \sigma^2_{\text{noise}} &\sim \mathcal{N}(f(x^{(n)}), \sigma^2_{\text{noise}}),\ n \in [N], \end{aligned} \tag{4.5}$$

where the first equation corresponds to $\exp(-J(f))$ and the last equation corresponds to $\exp(-\mathcal{L}(y^{(n)}, f(x^{(n)})))$. While there certainty exist other likelihoods and priors to model interactions and sparsity, many existing sparse Bayesian models are in the form of Eq. (4.5). Importantly, such models have been shown to have desirable statistical properties (theoretical or empirical); see, for example, Wei et al. [2019], Curtis et al. [2014], Griffin and Brown [2017], Agrawal et al. [2019d], Chipman [1996], George and McCulloch [1993]. As we show in the next section, the Gaussian likelihood and prior will allow us to leverage computational tricks to speed up inference.

4.3 Using two kernel tricks to reduce computation cost

Conditional on θ, Eq. (4.5) reduces to conjugate Bayesian regression. Hence, we can analytically compute the MAP estimate of Θ_V (and hence solve Eq. (4.1) in closed-form). Unfortunately, as we show below, it is computationally intractable to compute this closed-form solution in general. To remedy this computational intractability, we show how to make inference scale linearly with p by exploiting additional special model structure in Section 4.3.1 and Section 4.3.2.

Intractability of Conjugate Bayesian Regression. Our model in Eq. (4.5) has $B_Q := \sum_{V:|V|\leq Q} B_V$ parameters. In general, computing the MAP estimate of these B_Q parameters requires inverting a $B_Q \times B_Q$ covariance matrix. Hence, the computational cost of MAP inference scales as $O(B_Q^3 + NB_Q^2)$. B_Q may be prohibitively large for two reasons. The first problem arises if any of the basis expansion sizes (i.e., a particular B_V) is large. If $\mathcal{H}_V$ is infinite-dimensional, for example, then $B_V = \infty$. Even if all the $\mathcal{H}_V$ are finite-dimensional, B_V typically grows exponentially with the size of $|V|$; see, for example, Huang [1998]. The second issue arises from the combinatorial sum over interactions; even if all of the B_V equal 1, B_Q still has on the order of $O(p^Q)$ terms. Hence, without additional structure, the computation time for conjugate Bayesian regression is lower bounded by $\Omega(p^{3Q} + p^{2Q}N)$. Fortunately, due to unique structure in our problem, we show how to avoid the cost of explicitly generating the basis expansion ("Trick one" in Section 4.3.1), and summing over all $O(p^Q)$ interactions ("Trick two" in Section 4.3.2).

In what follows, we assume θ is fixed. Then, we to estimate θ in Section 4.5.

4.3.1 Trick one: How to represent and access sparsity without incurring the cost of a basis expansion

We show how to remove the computational dependence on the size of B_V through a kernel trick. We also show how to compute the functional ANOVA decomposition of $\hat{f}$ using this kernel trick. Our kernel generalizes the one used in Gu and Wahba [1993], which assumes independent covariates, to the case of general covariate distributions. In order to prove the existence of a kernel trick, we make the following assumption:

Assumption 4.3.1. Each $\mathcal{H}_V$ is a reproducing kernel Hilbert space (RKHS).

While there exists a reproducing kernel function that induces $\mathcal{H}_V$ by Assumption 4.3.1, the non-trivial part is showing the existence of a kernel to induce $\mathcal{H}_V^o$, which is not immediate due to the orthogonality constraints in Eq. (4.3).

Proposition 4.3.2. *(existence of a kernel trick) Under Assumption 4.3.1, there exists a positive-definite kernel k_V such that $k_V(x, \tilde{x}) = \langle \Phi_V(x), \Phi_V(\tilde{x}) \rangle$, where the components of $\Phi_V \in \mathbb{R}^{B_V}$ form a countable basis of $\mathcal{H}_V^o$.*

In Section 4.3.2, we show how to efficiently evaluate k_V without explicitly computing the feature maps. In light of Proposition 4.3.2, we introduce *model selection kernels* to rewrite the model in Eq. (4.5) as a Gaussian process. We then show how this reparametrization allows us to perform inference more efficiently.

Definition 4.3.3. A kernel k_θ is a *model selection kernel* if it can be written as $\sum_{V:|V|\leq Q} \theta_V k_V$, where k_V is the reproducing kernel for $\mathcal{H}_V^o$ and $k_\emptyset(x, \tilde{x}) = 1$ (i.e., the kernel that induces the space of constant functions $\mathcal{H}_\emptyset$).

Lemma 4.3.4. *Suppose k_θ is a model selection kernel. Then, the distribution of $Y \mid x$ in Eq.* (4.5) *has the same distribution as instead placing a zero-mean Gaussian process prior with covariance kernel k_θ on f.*

Based on the reparametrization in Lemma 4.3.4, $J(f)$ equals the penalty induced by the kernel k_θ. Hence, the solution to Eq. (4.1) reduces to kernel ridge-regression (i.e., the posterior predictive mean of the Gaussian process) by Rasmussen and Williams [2006, Chapter 2]:

$$\hat{f}(x) = \bar{f}_\theta(x) := \sum_{n=1}^{N} \hat{\alpha}_n k_\theta(x_n, x), \quad \hat{\alpha} = (K_\theta + \sigma^2_{\text{noise}} I_{N\times N})^{-1} Y, \tag{4.6}$$

where $Y_n = y^{(n)}$ and $[K_\theta]_{nm} = k_\theta(x^{(n)}, x^{(m)})$.

Unlike the weight-space view in Section 4.2.4 where $f_V = \Theta_V^T \Phi_V(\cdot)$, it is not clear how to actually recover f_V from $\bar{f}_\theta(x)$. For general kernels, accessing f_V (and consequently computing the functional ANOVA of $\bar{f}_\theta(x)$) lacks an analytical form. Fortunately, model selection kernels allow us to immediately recover each component in the functional ANOVA decomposition of $\bar{f}_\theta(x)$ as follows:

Lemma 4.3.5. *Suppose k_θ is a model selection kernel and $f^{(M)}(x) = \sum_{m=1}^{M} \alpha_m k_\theta(x_m, x)$ for $\alpha_m \in \mathbb{R}$ and $x_m \in \mathbb{R}^p$. Then, $f^{(M)}(x) = \sum_{V:|V|\leq Q} f_V$, where $f_V = \theta_V \sum_{m=1}^{M} \alpha_m k_V(x_m, x) \in \mathcal{H}_V^o$.*

As a consequence of Lemma 4.3.5, model selection kernels make it easy to perform variable selection[1] as well.

Corollary 4.3.6. *(non-linear variable selection) Suppose $f^{(M)}(x) = \sum_{m=1}^{M} \alpha_m k_\theta(x_m, x)$. Then, $f^{(M)}(x)$ functionally depends on the set of covariates $\{i : \exists V \subset [p], i \in V \text{ s.t. } \theta_V \neq 0\}$.*

While we have avoided the cost of generating the basis expansion to solve Eq. (4.1), computing Eq. (4.6) is still computationally intractable; k_θ sums over $O(p^Q)$ kernels. Hence, the cost to compute the kernel matrix K_θ and invert $(K_\theta + \lambda I_{N\times N})$ takes $O(N^2 p^Q)$ and $O(N^3)$ time, respectively.

4.3.2 Trick two: A recursion to avoid a combinatorially large summation over interactions in the presence of covariate independence

We show how to compute k_θ in $O(pQ)$ time (and hence solve Eq. (4.6) in $O(pQN^2 + N^3)$ time) for a particular subset of model selection kernels that we call *SKIM-FA* kernels. In what follows, we start by motivating SKIM-FA kernels from the hierarchical Bayesian characterization of model selection kernels in Eq. (4.5). Then, we show that when the covariates are independent, we can compute SKIM-FA kernels much more efficiently using a second kernel trick. In Section 4.4, we generalize to the non-independent covariate case by building on top of the procedure described in this section.

The Sparse Kernel Interaction Model for Functional ANOVA (SKIM-FA). In Eq. (4.5), $\{\theta_V\}$ does not necessarily have any special structure. SKIM-FA, on the other hand, assumes that $\theta_V = \prod_{i\in V} \eta_{|V|}^2 \kappa_i^2$ for some non-negative random vectors $\kappa \in \mathbb{R}_+^p$ and $\eta \in \mathbb{R}_+^{Q+1}$. Then, the prior on Θ_V in Eq. (4.5) simplifies to

$$\Theta_V \mid \eta, \kappa \sim \mathcal{N}\left(0, \eta_{|V|}^2 \prod_{i\in V} \kappa_i^2 \cdot I_{B_V} \times I_{B_V}\right), \tag{4.7}$$

for SKIM-FA, which generalizes the prior used in Agrawal et al. [2019d] for linear pairwise interaction models.

SKIM-FA Interpretation. In Eq. (4.7), $\eta_{|V|}^2$ quantifies the overall strength of $|V|$-way interactions by modifying the prior variance of all effects of order $|V|$. κ_i plays the

[1]For general non-linear regression functions, performing variable selection can be challenging; in principle, showing that a fitted regression function does not depend on x_i requires checking that the fitted function is constant in x_i across the entire domain.

role of a "variable importance" measure for covariate x_i by affecting the prior variance of all effects involving covariate x_i. Hence, if it turns out an effect involving x_i is strong, the posterior of κ_i will place high-probability at large values (i.e., indicating that covariate x_i has high "importance"). Notice that if $\kappa_i = 0$, then the prior variance of Θ_V equals 0 whenever $i \in V$. Consequently, all effects involving x_i will equal 0. Hence, we can perform variable selection in $O(p)$ time by just examining the sparsity pattern of κ instead of in $O(p^Q)$ time using Corollary 4.3.6. In Section 4.5, we show how we select our sparsity prior on κ. Finally, note that while we added more structure to the prior, we have not lost modeling flexibility; as long as $\mathbb{P}^*(\kappa_i)$ does not equal 0 with probability one, then the prior variance of Θ_V will be non-zero. Hence, our prior will have support on all of $\mathcal{H}_Q$.

Definition 4.3.7. A *SKIM-FA kernel* is a model selection kernel that can be written as

$$k_{\text{SKIM-FA}}(x, \tilde{x}) = \sum_{V:|V|\leq Q} \left[\eta_{|V|}^2 \prod_{i\in V} \kappa_i^2\right] k_V(x, \tilde{x}).$$

for some $\kappa \in \mathbb{R}^p$ and $\eta \in \mathbb{R}^{Q+1}$.

Proposition 4.3.8. *For a SKIM-FA kernel, Eq.* (4.5) *can be replaced by Eq.* (4.7) *in Lemma 4.3.4.*

Proof. Set $\theta_V = \eta_{|V|}^2 \prod_{i\in V} \kappa_i^2$ in Lemma 4.3.4. □

Efficient Evaluation of SKIM-FA Kernels. Recall that k_i is the reproducing kernel for $\mathcal{H}_i^o$. Suppose, for the moment, that the reproducing kernel k_V for $\mathcal{H}_V^o$ equals $\prod_{i\in V} k_i$ (we will shortly show that this condition holds when the covariates are independent). Then, by Theorem 4.3.9 and Corollary 4.3.10 below, we can compute SKIM-FA kernels orders of magnitude faster by not explicitly summing over all $O(p^Q)$ interactions in Definition 4.3.7.

Theorem 4.3.9. *Suppose $k_V(x, \tilde{x}) = \prod_{i\in V} k_i(x_i, \tilde{x}_i)$. Then,*

$$\begin{aligned}
&k_{\text{SKIM}-\text{FA}}(x, \tilde{x}) = \sum_{q=1}^{Q} \eta_q^2 \bar{k}_q(x, \tilde{x}) \quad \text{s.t.} \\
&\bar{k}_q(x, \tilde{x}) = \frac{1}{q}\sum_{s=1}^{q} (-1)^{s+1} \bar{k}_{q-s}(x, \tilde{x}) k^s(x, \tilde{x}), \quad \bar{k}_0(x, \tilde{x}) = 1, \qquad (4.8) \\
&k^s(x, \tilde{x}) = \sum_{i=1}^{p} \kappa_i^{2s} [k_i(x_i, \tilde{x}_i)]^s.
\end{aligned}$$

As we show in Appendix D.1, the key to proving Theorem 4.3.9 is an old recursive kernel formula provided in Vapnik [1995, pg. 199]. From Theorem 4.3.9, we have two corollaries. The first requires a short inductive argument; see Appendix D.1. The second follows immediately by setting $Q = 2$ into Eq. (4.8).

Corollary 4.3.10. *$k_{\mathrm{SKIM-FA}}(x, \tilde{x})$ takes $O(pQ)$ time to evaluate on a pair of points.*

Corollary 4.3.11. *Suppose $Q = 2$. Then, $k_{SKIM\text{-}FA}(x, \tilde{x})$ equals*

$$.5\eta_2^2 \left[\left(\sum_{i=1}^{p} \kappa_i^2 k_i(x_i, \tilde{x}_i) \right)^2 - \sum_{i=1}^{p} \kappa_i^4 [k_i(x_i, \tilde{x}_i)]^2 \right] + \eta_1^2 \sum_{i=1}^{p} \kappa_i^2 k_i(x_i, \tilde{x}_i) + \eta_0^2 \tag{4.9}$$

To see how Eq. (4.8) acts as another kernel trick, consider the linear interaction case when $\mathcal{H}_Q$ consists of interactions of the form $\prod_{i \in V} x_i$. Suppose further that κ and η are equal to the ones vector. Then, $k_{\mathrm{SKIM-FA}}(x, \tilde{x}) = \sum_{V:|V| \leq Q} \prod_{i \in V} x_i \tilde{x}_i$, which explicitly generates and sums over the interactions $\prod_{i \in V} x_i$. However, it is well known that polynomial kernels implicitly generate interactions, and hence can be used instead to avoid summing over all interactions. The core idea in Eq. (4.8) is similar; the kernels k^s raised to the s power in Eq. (4.8) implicitly generate interactions of order equal to s just like a polynomial kernel. However, instead of generating interactions of the form $\prod_{i \in V} x_i$, k^s operates on one-dimensional kernels k_i to generate interactions of the form $\prod_{i \in V} k_i$. Since $k_V = \prod_{i \in V} k_i$ by assumption, these "interactions" of kernels span $\mathcal{H}_V^o$ by the product property of kernels.

We conclude by providing sufficient conditions for when k_V actually simplifies as $\prod_{i \in V} k_i$ by using a result from Gu and Wahba [1993]. We leave our construction of k_i to Appendix C.3.

Assumption 4.3.12. (Tensor product space) For all $V \subset [p]$ and $1 \leq |V| \leq Q$, $\mathcal{H}_V = \bigotimes_{i \in V} \mathcal{H}_i$.

Proposition 4.3.13. *[Gu and Wahba, 1993] Suppose $\mu = \mu_{\otimes}$, where $\mu_{\otimes}(x) := \mu_1(x_1) \otimes \mu_2(x_2) \cdots \otimes \mu_p(x_p)$ and μ_j is the marginal distribution of x_j. Then, under Assumption 4.3.1 and Assumption 4.3.12, $k_V = \prod_{i \in V} k_i$.*

Since any Hilbert space of square-integrable functions of x_V can be approximated arbitrarily well by taking tensor products of one-dimensional Hilbert spaces by Stone [1994], Huang [1998], Assumption 4.3.12 is a mild assumption. The more problematic assumption is that all covariates are independent (i.e., that $\mu = \mu_{\otimes}$).

4.4 How to get sparsity, interactions, and fast inference when covariates are dependent

Here we extend to the general μ case. We start by motivating why this extension is important in Section 4.4.1. Then, in Section 4.4.2, we develop a change-of-basis formula to take the functional ANOVA decomposition of $\bar{f}_\theta$ with respect to $\mu_{\otimes}$ to one with respect to μ. While our formula assumes $Q = 2$, we expect that our formula naturally extends beyond the pairwise case.

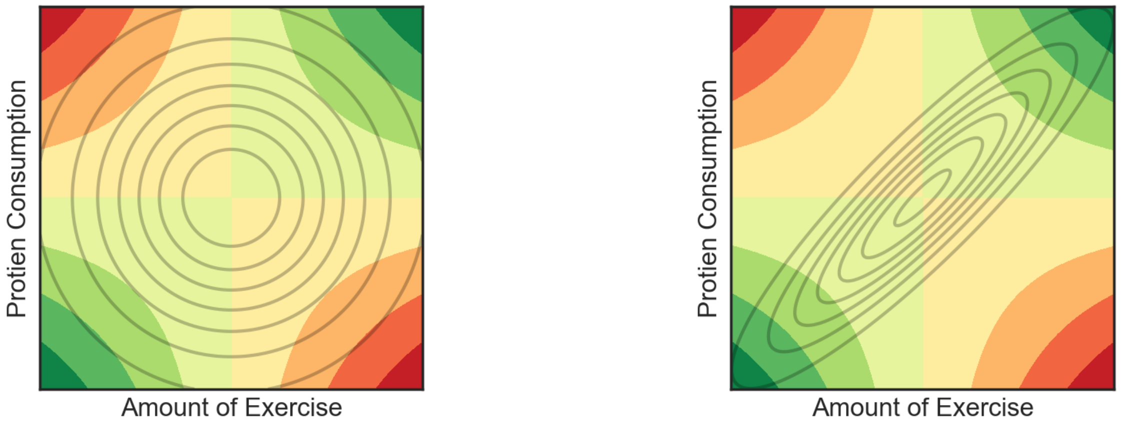


Figure 4-1: The colors denote the contour plot of the function $f^*(x_1, x_2) = x_1 x_2$. Darker green indicates more larger values while darker red indicates stronger negative values. The gray solid lines in the left and right hand figures represent the density contours of $\mu_\otimes$ and μ in Example 4.4.2, respectively.

4.4.1 Practical problems that arise when assuming independent covariates

Even though $\mu_\otimes$ has the same 1D marginal distributions as μ, we prove that the functional ANOVA decomposition of an $f \in \mathcal{H}$ can be *arbitrarily* different depending on the choice of measure. We prove this claim by showing something stronger, namely that the intercepts between two functional ANOVA decompositions can be arbitrarily far apart.

Proposition 4.4.1. *For any $\Delta > 0$, there exists a probability measure μ and square-integrable f such that the relative difference*

$$\frac{|f_\emptyset^\mu - f_\emptyset^{\mu_\otimes}|}{|f_\emptyset^\mu|} > \Delta, \quad \textit{where} \quad f_\emptyset^\mu = \mathbb{E}_\mu f(X) \quad \textit{and} \quad f_\emptyset^{\mu_\otimes} = \mathbb{E}_{\mu_\otimes} f(X).$$

To build intuition for Proposition 4.4.1, and motivate using μ instead of $\mu_\otimes$ to compute the functional ANOVA decomposition of $\bar{f}_\theta$, consider the following toy example.

Example 4.4.2. Suppose $f^*(x_1, x_2) = x_1 x_2$, where x_1 could represent exercise, x_2 protein consumption, and $f^*(x_1, x_2)$ the expected cardiovascular health of an individual who consumes x_1 grams of protein and exercises x_2 times per week. Suppose exercise and protein consumption are positively correlated and that μ equals a multivariate Gaussian distribution with mean zero, unit covariance, and correlation equal to .9. Then, $\mu_\otimes$ equals a multivariate Gaussian distribution with mean zero, unit covariance but correlation equal to 0. Suppose we report the intercept $f_\emptyset$ to summarize the typical cardiovascular health in the population. In the functional ANOVA decomposition of f^* with respect to μ, $f_\emptyset = \mathbb{E}_\mu[f^*] = \mathbb{E}_\mu[f^* + \epsilon] = \mathbb{E}_\mu[Y] \approx .89$. Hence, this intercept has the interpretation as the average cardiovascular health in the population. If we instead use $\mu_\otimes$, then $f_\emptyset = \mathbb{E}_{\mu_\otimes} f^* = 0 \neq \mathbb{E}_\mu[f^*]$. In this case, it is not clear how to interpret this intercept; $\mu_\otimes$ averages the regression surface f^* over individuals who rarely occur in the actual population (e.g., those who exercise very frequently but not not consume much protein); see also Fig. 4-1 for a visualization.

4.4.2 A change of basis to handle covariate dependence

We generalize to the non-independent case through a change-of-basis formula provided in Theorem 4.4.5. Our formula allows us to re-expresses the effects estimated using the kernel in Section 4.3.2, which assumes independent covariates, to one with respect to the actual distribution μ. Our idea is similar to ideas in numerical linear algebra; we use one parameterization of a vector space, in our case the space of functions $\mathcal{H}_Q$, that makes computation "nice." Once we finish computation in the "nice" parameterization, we use a change-of-basis formula to report the actual quantity we care about in the original parameterization of the space, namely reporting the functional ANOVA decomposition of our fit $\bar{f}_\theta$ with respect to μ.

To make this idea mathematically precise, suppose we can write $\mathcal{H}_Q$ using two different parameterizations, one that uses $\mu_\otimes$ in Eq. (4.3) (denoted as $\mathcal{H}^o_{V,\mu_\otimes}$) and the other that uses μ in Eq. (4.3) (denoted as $\mathcal{H}^o_{V,\mu}$). Then,

$$\mathcal{H}_Q = \bigoplus_{V:|V|\leq Q} \mathcal{H}_V \tag{4.10}$$

$$= \bigoplus_{V:|V|\leq Q} \mathcal{H}^o_{V,\mu_\otimes} \tag{4.11}$$

$$= \bigoplus_{V:|V|\leq Q} \mathcal{H}^o_{V,\mu}. \tag{4.12}$$

If these equalities indeed hold, then we can use Theorem 4.3.9 to estimate f^* in $O(pQN^2 + N^3)$ time. Hence, it suffices to show how to take this estimate of f^* and compute its functional ANOVA decomposition with respect to μ instead of $\mu_\otimes$ (i.e., move from the parameterization in Eq. (4.11) to the one in Eq. (4.12)). We show how to compute this change-of-basis when all the $\mathcal{H}_i$ are finite-dimensional.

Assumption 4.4.3. For all $i \in [p]$ there exists a $B_i < \infty$ and linearly independent set of continuous functions $\{\phi_{ib}\}_{i=1}^{B_i}$ such that $\mathcal{H}_i = \text{span}\{1, \phi_{i1}, \cdots, \phi_{iB_i}\}$ and $\Phi_i = [\phi_{i1}, \cdots, \phi_{iB_i}]^T$.

Assumption 4.4.3 is a mild condition since we can approximate any function arbitrarily well by setting B_i sufficiently large since $\mathcal{H}_i$ is seperable; see, Huang [1998], for rates of convergence for different finite-basis approximations. Under this assumption, Lemma 4.4.4 implies that $\mathcal{H}^o_{V,\mu_\otimes} = \mathcal{H}^o_{V,\mu}$. Hence, a change-of-basis formula exists. We provide the change-of-basis formula for $Q = 2$ in Theorem 4.4.5.

Lemma 4.4.4. *Under Assumption 4.3.12 and Assumption 4.4.3, any $f \in \mathcal{H}$ is square-integrable with respect to any probability measure.*

Theorem 4.4.5. *Suppose $Q = 2$ and that Assumption 4.3.12 and Assumption 4.4.3 hold. For $f \in \mathcal{H}$, let*

$$\begin{aligned} f &= f_{\emptyset}^{\mu_\otimes} + \sum_{i=1}^{p} f_{\{i\}}^{\mu_\otimes} + \sum_{i,j=1}^{p} f_{\{i,j\}}^{\mu_\otimes} \\ &= f_{\emptyset}^{\mu} + \sum_{i=1}^{p} f_{\{i\}}^{\mu} + \sum_{i,j=1}^{p} f_{\{i,j\}}^{\mu} \end{aligned}$$

be the functional ANOVA decomposition of f with respect to $\mu_\otimes$ and μ, respectively. Then, there exists unique coefficients, $\Psi_{ij}^{i} \in \mathbb{R}^{1\times B_i}$, $\Psi_{ij}^{j} \in \mathbb{R}^{1\times B_j}$, $\Psi_{ij}^{0} \in \mathbb{R}$, such that

$$\begin{aligned} f_{\{i,j\}}^{\mu}(x_i, x_j) &= f_{\{i,j\}}^{\mu_\otimes}(x_i, x_j) - [\Psi_{ij}^{i}\Phi_i(x_i) + \Psi_{ij}^{j}\Phi_j(x_j) + \Psi_{ij}^{0}] \\ f_{\{i\}}^{\mu}(x_i) &= f_{\{i\}}^{\mu_\otimes}(x_i) + \sum_{j>i} \Psi_{ij}^{i}\Phi_i(x_i) + \sum_{j<i} \Psi_{ji}^{i}\Phi_i(x_i) \\ f_{\emptyset}^{\mu} &= f_{\emptyset}^{\mu_\otimes} + \sum_{i<j} \Psi_{ij}^{0}, \end{aligned} \tag{4.13}$$

where Φ_i denotes the (finite-dimensional) feature map in Definition 4.3.3.

We prove Theorem 4.4.5 in Appendix D.1. By Corollary 4.3.10, we can estimate $f_{\{i\}}^{\mu_\otimes}(x_i)$ and $f_{\{i,j\}}^{\mu_\otimes}(x_i, x_j)$ in time linear in p. Hence, it remains to show how we can actually compute each Ψ_{ij}^{i} in Theorem 4.4.5. In Section 4.5, we show how to estimate Ψ_{ij}^{i} arbitrarily well using a Monte Carlo approach.

4.5 Final algorithm and implementation details

We start by describing and motivating our choice of sparsity prior on κ. Then, we show how we fit κ and η_q using cross-validation and our computational tricks in Section 4.3. We conclude by showing how we compute Ψ_{ij}^{i} in Theorem 4.4.5 via Monte Carlo.

Our Sparsity Inducing Prior on κ. To induce sparsity in κ for variable selection, we pick a prior on κ_i that equals the mixture of a discrete point mass at 0 and a Uniform(0, 1) random variable. Similar to a *spike-and-slab* prior [George and McCulloch, 1993], the point mass at 0 allows us to achieve exact sparsity. Unlike a spike-and-slab prior, however, we construct our prior so that we can still take gradients (and hence use continuous optimization techniques like gradient descent); see Algorithm 2 for details. Our construction involves introducing another random variable U_i so that

$$\kappa_i = \frac{1}{1-c}\max(U_i - c, 0),\ U_i \sim \text{Uniform}(0, 1). \tag{4.14}$$

Then, $\mathbb{P}^*(\kappa_i = 0) = c$. Otherwise, with probability $1 - c$, $\kappa_i \sim \text{Uniform}(0, 1)$. Hence, c plays a similar role as a prior inclusion probability in a spike-and-slab prior. Since κ_i

is a deterministic function of U_i, it suffices to estimate U_i instead as we detail below.

Cross-Validation Loss and Optimization. Given the empirical success of cross-validation and its use in other functional ANOVA methods (e.g., as in Gu and Wahba [1993], Lin and Zhang [2006]), we also use cross-validation to fit the SKIM-FA kernel hyperparameters κ and η. Specifically, we would like to pick $U, \eta, \sigma^2_{\text{noise}}$ (where $\kappa_i = \frac{1}{1-c}\max(U_i - c, 0)$) by minimizing a leave-$M$-out cross validation loss:

$$\begin{aligned} L(U, \eta, \sigma^2_{\text{noise}}) &= \frac{1}{\binom{N}{M}} \sum_{\substack{A: A \subset [N] \\ |A| = N-M}} \left[\frac{1}{M} \sum_{m \in A} (y^{(m)} - \bar{f}_A(x^{(m)}))^2 \right], \\ &= \mathbb{E}_{A \sim \pi} \left[\frac{1}{M} \sum_{m \in A} (y^{(m)} - \bar{f}_A(x^{(m)}))^2 \right] \end{aligned} \tag{4.15}$$

where $\bar{f}_A$ equals the kernel ridge regression fit in Eq. (4.6) using the subset of datapoints in A and π equals the uniform distribution over all $N - M$ sized partitions of $[N]$.

Since the gradient of $L(U, \eta, \sigma^2_{\text{noise}})$ exists[2], we can minimize Eq. (4.15) using gradient descent. However, this loss is computationally intensive; we need to solve Eq. (4.6) $\binom{N}{M}$ times in order to take a single gradient descent step. Instead, we approximate Eq. (4.15) by using Monte Carlo cross validation. Specifically, we randomly draw a single A from π in Eq. (4.15) and use the mean-squared prediction error of $\bar{f}_A$ to estimate Eq. (4.15). Then, this estimate leads to an unbiased estimate of Eq. (4.15), and hence the gradient of $L(U, \eta, \sigma^2_{\text{noise}})$. We summarize our full procedure in Algorithm 2. Note that in Algorithm 2 we do not minimize over U but instead over $\tilde{U}$, where $U_i = \frac{\tilde{U}_i^2}{\tilde{U}_i^2 + 1}$. Since the range of $\frac{\tilde{U}_i^2}{\tilde{U}_i^2 + 1}$ equals $(0, 1)$ when $\tilde{U}_i$ varies over all of $\mathbb{R}$, we can optimize $\tilde{U}_i$ over an unconstrained domain.

Proposition 4.5.1. *Suppose $\kappa_i^{(t)} = 0$ at some iteration t in Algorithm 2. Then, for all subsequent iterations $t' \geq t$, $\kappa_i^{(t')} = 0$.*

Based on Proposition 4.5.1, we may view Algorithm 2 as a gradient-based analogue of backward stepwise regression; we start with the full saturated model by initializing all $\tilde{U}_i = 1$ (and consequently all $\kappa_i > 0$). Then, we keep pruning off covariates the longer we run gradient descent. We demonstrate empirically in Section 6.5 that the actual data-generating covariates remain while the irrelevant covariates get pruned off. Once we have found the kernel hyperparameters from Algorithm 2, Algorithm 3 and Algorithm 4 show how to perform variable selection and recover the effects, respectively. Both Algorithm 3 and Algorithm 4 follow directly from Corollary 4.3.6 and Lemma 4.3.5.

[2]Although $\frac{1}{1-c}\max(U_i - c, 0)$ is not differentiable at c, we may instead take the sub-gradient. To compute gradients, we use the automatic differentiation library `PyTorch`.

Algorithm 2 Learn SKIM-FA Kernel Hyperparameters and Kernel Ridge Weights

```
1: procedure LearnHyperParams($M$, $\gamma$, $T$)
2:     For all $i \in [p]$, set $\tilde{U}_i^{(0)} = 1$
3:     For all $q \in [Q]$, set $\eta_q^{(0)} = 1$
4:     $\sigma_{\text{noise}}^{(0)} = \sqrt{.5\text{var}(Y)}$ ▷ Initialize noise variance as half of the response variance
5:     $\tau^{(0)} = (\tilde{U}, \eta_q^{(0)}, \sigma_{\text{noise}}^{(0)})$
6:     for $t \in 1 : T$ do                ▷ Update parameters via gradient descent
7:         Randomly select $N - M$ datapoints and collect covariates and responses in $X_A$, $Y_A$
8:         $U_i^{(t)} = \frac{[\tilde{U}_i^{(t)}]^2}{[\tilde{U}_i^{(t)}]^2+1}, i \in [p]$
9:         $\kappa_i^{(t)} = \max(U_i^{(t)} - c, 0), i \in [p]$
10:        Compute kernel matrix $K_\tau^A$, where $[K_\tau^A]_{ij} = k_{\text{SKIM-FA}}(X_i^A, X_j^A)$ via Eq. (4.8)
11:        Let $f_A$ equal the solution of Eq. (4.6) with $\lambda = [\sigma_{\text{noise}}^{(t)}]^2$, $K = K_\tau^A$, $Y = Y_A$
12:        $L = \frac{1}{M}\sum_{n\in A^c}(y^{(n)} - f_A(x^{(n)}))^2$
13:        $\tau^{(t+1)} = \tau^{(t)} - \gamma\nabla_{\tau^{(t)}} L$    ▷ Compute gradients via an automatic differentiation library
14:    Compute $\alpha^{(T)}$, the kernel ridge regression weights found by solving Eq. (4.6) using all $N$ datapoints with SKIM-FA hyperparameters equal to $\kappa^{(T)}, \eta^{(T)}, \sigma_{\text{noise}}^{(T)}$
15:    return $\kappa^{(T)}, \eta^{(T)}, \sigma_{\text{noise}}^{(T)}, \alpha^{(T)}$
```

Algorithm 3 SKIM-FA Variable Selection

```
1: procedure VarSelect($\kappa$)
2:     return $\{i : \kappa_i \neq 0\}$
```

Algorithm 4 Estimated functional ANOVA effect $\bar{f}_V$ of $\bar{f}_\theta$ with respect to $\mu_\otimes$

```
1: procedure OrthEffects($V$, $\alpha$, $\kappa$, $\eta$, $\alpha$)
2:     $\theta_V = \eta_{|V|}^2 \prod_{i\in V}\kappa_i^2$
3:     return $\bar{f}_V(\cdot) = \theta_V \sum_{n=1}^N \alpha_n k_V(x^{(n)}, \cdot)$
```

Estimating Ψ^i_{ij} for Change-of-Basis Formula in Theorem 4.4.5. To estimate Ψ^i_{ij}, we use a Monte Carlo procedure in Algorithm 5.

Algorithm 5 Change of Basis Formula for Finite Dimensional Model Selection Kernels

```
1: procedure ReExpressEffect($\alpha$, $k_\theta$, $W$, $\mu$)
2:     Compute $f^{\mu\otimes}_{\{i,j\}}, f^{\mu\otimes}_{\{i\}}, f^{\mu\otimes}_{\emptyset}$ using Algorithm 4
3:     For $1 \le w \le W$ randomly sample $x^{(w)} \overset{\text{iid}}{\sim} \mu$
4:     Compute $X_{ij} = [\Phi_i(x_i^{(1)}) \cdots \Phi_i(x_i^{(W)})\ \Phi_j(x_j^{(1)}) \cdots \Phi_j(x_j^{(W)})]^T$
5:     Compute $f^{\mu\otimes}_{ij,W} = [f^{\mu\otimes}_{\{i,j\}}(x_i^{(1)}, x_j^{(1)}) \cdots f^{\mu\otimes}_{\{i,j\}}(x_i^{(W)}, x_j^{(W)})]^T$
6:     Compute $[\hat{\Psi}^i_{ij}\ \hat{\Psi}^j_{ij}]^T = (X_{ij}^T X_{ij})^{-1} X_{ij}^T f^{\mu\otimes}_{ij,W}$          ▷ Least-squares projection
7:     Compute $\hat{f}^{\mu}_{\{i,j\}} = f^{\mu\otimes}_{\{i,j\}} - [\hat{\Psi}^i_{ij}\Phi_i^T(\cdot) + \hat{\Psi}^j_{ij}\Phi_j^T(\cdot) + \Psi^0_{ij}]$
8:     Compute $\hat{f}^{\mu}_{\{i\}} = f^{\mu\otimes}_{\{i\}} + \sum_{j>i} \hat{\Psi}^i_{ij}\Phi_i(\cdot) + \sum_{j<i} \hat{\Psi}^i_{ji}\Phi_i(\cdot)$
9:     Compute $\hat{f}^{\mu}_{\emptyset} = f^{\mu\otimes}_{\emptyset} + \sum_{i<j} \hat{\Psi}^0_{ij}$
10:    return $\hat{f}^{\mu}_{\{i,j\}}, \hat{f}^{\mu}_{\{i\}}, \hat{f}^{\mu}_{\emptyset}$
```

Proposition 4.5.2. *Let $W \to \infty$ in Algorithm 5. Then, the components returned from Algorithm 5 converge to the decomposition in Eq.* (4.13).

4.6 Related Work

The seminal work by Gu and Wahba [1993] used a type of model selection kernel to estimate the functional ANOVA decomposition of f^* with splines. Since the method in Gu and Wahba [1993] does not lead to sparsity, Gunn and Kandola [2004], Lin and Zhang [2006] put an L_1 penalty on θ to achieve sparsity, similar to *multiple kernel learning* techniques [Lanckriet et al., 2004]. Adding an L_1 penalty does not lead to an analytical solution nor a convex optimization problem. Hence, Gunn and Kandola [2004], Lin and Zhang [2006] alternate between minimizing θ and recomputing $\bar{f}_\theta$, similar to Algorithm 2. Other approaches use cross-validation and gradient descent to iteratively select θ [Gu and Wahba, 1993]. In either case, the computational bottleneck is computing and inverting $(K_\theta + \sigma^2_{\text{noise}} I_{N\times N})^{-1}Y$: k_θ takes $O(p^Q)$ time to compute on a pair of points. Hence, computing and inverting $K_\theta + \sigma^2_{\text{noise}} I_{N\times N}$ takes $O(p^Q N^2)$ time and $O(N^3)$ time, respectively.

Many existing functional ANOVA techniques assume that all covariates are independent, i.e., that μ equals *product measure*; see, for example, Gunn and Kandola [2004], Lin and Zhang [2006], Gu and Wahba [1993], Durrande et al. [2013]. Hooker [2007] highlighted pathologies that arise when using $\mu_\otimes$ instead of μ. Specifically, he empirically showed on synthetic and real data that the functional ANOVA decomposition of an $f \in \mathcal{H}$ with respect to μ can be significantly different than the decomposition with respect to $\mu_\otimes$. This discrepancy arises because $\mu_\otimes$ can place high probability in regions where the actual covariate distribution μ has low probability; see also Section 4.4.1.

4.7 Experiments

In this section, we compare our inference methods in Section 4.5 against existing procedures in terms of variable selection and estimation performance. We describe these benchmark methods and performance metrics in Section 4.7.1 and Section 4.7.2, respectively. In Section 4.7.3, we evaluate each method on simulated data so that we have ground truth effects. In Section 4.7.4, we evaluate methods on a real dataset (the bike-sharing dataset from the UCI machine-learning repository) using a similar procedure as in Agrawal et al. [2019d] to construct a synthetic ground truth. To highlight the importance of Algorithm 5, we conclude by showing the sensitivity of functional ANOVA decompositions with respect to the measure chosen in Section 4.7.5. All results can be re-generated using the data and code provided in https://github.com/agrawalraj/banova.

4.7.1 Benchmark Methods

We compare our method against other methods used to model high-dimensional data and interactions. We focus on the $Q = 2$ case throughout since (1) existing methods typically only work for the pairwise interaction case and (2) higher order interactions are often difficult to interpret and estimate. Even when $Q = 2$, the functional ANOVA methods outlined in Section 4.7.1 take $O(p^2N^2 + N^3)$ time, making them computationally intractable for even moderate p and N settings. Instead, we focus on methods that can model interactions and actually scale to moderate-to-large p and N settings. These methods include approximate "two-stage" and greedy forward-stage regression methods, and linear interaction models. We detail these approaches in more depth in Appendix C.2. The list below summarizes the candidate methods (and software implementations) selected from each category for empirical evaluation[3].

- **SPAM-2Stage**: performs variable selection by fitting a sparse additive model (SpAM) [Liu et al., 2008] to the data. We use the `sam` package in `R`. Since `sam` does not provide a default way to select the L_1 regularization strength, we use 5-fold cross-validation. For estimation, we generate all main and interaction effects among the subset of covariates selected by SpAM. We calculate these effects by taking pairwise products of univariate basis functions generated from a cubic spline basis with 5 total knots; see Appendix C.3 for details. The basis coefficients (and hence effects) are estimated using ridge-regression, where again we use 5-fold cross-validation to pick the L_2 regularization strength.

- **Multivariate Additive Regression Splines (MARS)**: we use the `python` implementation of MARS [Friedman, 1991] in `py-earth`. We consider two functional ANOVA decompositions of the fitted regression function:

[3]Agrawal et al. [2019d] fit linear interaction models in $O(pN^2 + N^3)$ time per iteration, which has the same asymptotic complexity as Algorithm 2. However, they use Hamiltonian Monte Carlo (HMC) to perform inference. Each HMC step requires computing and inverting an $N \times N$ kernel matrix many times. Hence, their method takes hours to complete when p and N are larger than 500. Due to this computational intensity, we do not benchmark against their method.

  - **MARS-Vanilla**: the main effect of each covariate equals the sum of all selected univariate basis functions of that covariate (i.e., after the pruning step of MARS). Similarly, each pairwise effect equals the sum of all selected bivariate basis functions of those two covariates. This is the functional ANOVA decomposition originally proposed in Friedman [1991] and the one actually implemented in existing MARS software packages. It is unclear, however, what measure this functional ANOVA decomposition is taken with respect to.
  - **MARS-EMP**: to the best of our knowledge, there currently does not exist a procedure to perform a functional ANOVA decomposition of MARS with respect to $\mu_{\otimes}$. We describe how to perform such a decomposition in Appendix C.4, which could be of independent interest.

- **Hierarchical Lasso (HierLasso)**: we use the implementation of HierLasso [Lim and Hastie, 2015] using the authors' `R` package `glinternet`. Since Lim and Hastie [2015] use cross-validation to pick the L_1 regularization strength, we similarly use 5-fold cross-validation.

- **Pairs Lasso**: runs the standard Lasso on the expanded set of features, i.e., $\{x_i\}_{i=1}^p$ and $\{x_i x_j\}_{i,j=1}^p$. We fit the Lasso using the `python` package `sklearn`, and use 5 fold cross-validation to select the L_1 regularization strength.

4.7.2 Evaluation Metrics

Variable Selection Evaluation Metrics. We consider both the power to select correct covariates and avoid incorrect ones.

- # of Correct Selected - number of covariates correctly selected by the method. Higher is better.

- # Wrong Selected: number of covariates incorrectly selected by the method (i.e., Type I error). Lower is better.

- # Correct Not Selected - number of covariates that belong to the true model but were not selected by the method (i.e., Type II error). Lower is better.

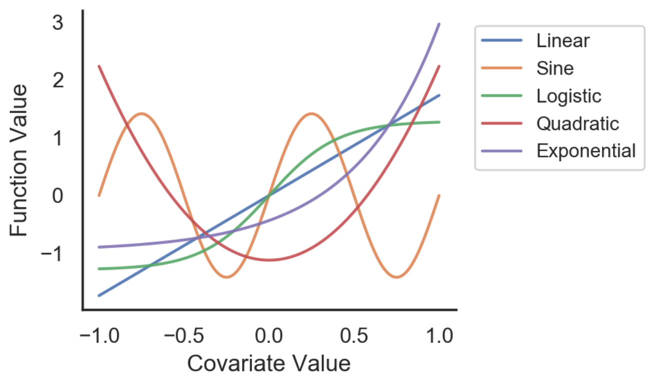
Estimation Evaluation Metrics. We evaluate how well a method estimates main effects and interaction effects. Instead of only looking at the total mean squared estimation error, we break this error into multiple buckets to understand what bucket drives the majority of the error. Lower is better for all of the quantities below.

- Correct Selected SSE (Main) - computes the sum of squared errors (SSE) between each estimated main effect component and true main effect component. This equals $\sum_{i \in S_1} \|f_i^* - \hat{f}_i\|_{\mu}^2$, where S_1 is the set of correctly identified main effects, $\hat{f}_i$ is the estimated main effect, and f_i^* is the true main effect.

- Correct Not Selected SSE (Main) - computes the total squared norm of each main effect component not selected. This equals, $\sum_{i \in S_2} \|f_i^*\|_\mu^2$, where S_2 is the set of correct main effects not selected.
- Wrong Selected SSE (Main) - computes the total squared norm of each main effect component incorrectly selected. This equals, $\sum_{i \in S_3} \|\hat{f}_i\|_\mu^2$, where S_3 is the set of incorrect main effects selected.
- Correct Selected SSE (Pair) - same as Correct Selected SSE (Main) except for correctly selected interactions
- Correct Not Selected SSE (Pair) - same as Correct Not Selected SSE (Main) except for correct interactions not selected
- Wrong Selected SSE (Pair) - same as Wrong Selected SSE (Main) except for interactions incorrectly selected
- Total SSE - the sum of the 6 buckets above
- Total SSE / Signal Variance - this equals the relative estimation error, i.e., Total SSE divided by the true signal variance.

4.7.3 Synthetic Data Evaluation

Figure 4-2: Synthetic Data Test Functions.



We randomly generate covariates and responses as follows. For the covariates, we draw each data point and covariate dimension $x_i^{(n)} \overset{\text{iid}}{\sim} \text{Uniform}([-1, 1])$. Since $[-1, 1]$ is compact, Theorem 4.2.1 ensures that the functional ANOVA decomposition is unique. We let y depend on the first 5 covariates; the remaining $p - 5$ covariates are taken as noise covariates that we do not want to select. To generate responses reflective of what we might expect in real data, we consider the 5 trends shown in Fig. 4-2: linear, sine, logistic, quadratic, and exponential. We let the main effects equal the sum of these 5 trends, where the ith trend is applied to covariate i. For the interactions between the

Table 4.1: Synthetic Data Variable Selection Performance Results for $p = 1000$

Method	Setting	# Correct Selected	# Wrong Selected	# Correct Not Selected
SKIM-FA	Main-Only	3	0	2
HierLasso	Main-Only	4	5	1
Pairs Lasso	Main-Only	4	6	1
SPAM-2Stage	Main-Only	5	15	0
MARS	Main-Only	5	70	0
SKIM-FA	Equal	5	0	0
SPAM-2Stage	Equal	5	15	0
HierLasso	Equal	4	40	0
MARS	Equal	4	71	0
Pairs Lasso	Equal	5	213	0
SKIM-FA	Weak Main	5	9	0
SPAM-2Stage	Weak Main	1	41	4
MARS	Weak Main	5	75	0
HierLasso	Weak Main	5	120	0
Pairs Lasso	Weak Main	5	144	0

first 5 covariates, we consider all pairwise products of the 5 trends above, resulting in 10 total interactions. We select a noise variance such that the $R^2 = \frac{\sigma^2_{\text{signal}}}{\sigma^2_{\text{signal}}+\sigma^2_{\text{noise}}} = .8$, where $\sigma^2_{\text{signal}} = \langle f^*, f^* \rangle_\mu$. We further decompose the signal variance in terms of the total variance explained by main effects and interactions. Similar to the empirical evaluations in Lim and Hastie [2015] we consider the following three settings:

- **Main Effects Only:** each of the 5 main effects has 1/5th of the total signal variance, and each pairwise effect has 0 signal variance (i.e., no pairwise interactions).
- **Equal Main and Interaction Effects:** each main effect and pairwise effect has .5 * 1/5 and .5(1 / 10) of the total signal variance, respectively. Hence, the total main effect variance equals the total pairwise signal variance.
- **Weak Main Effects:** each main effect and pairwise effect has .01 * 1/5 and .99(1 / 10) of the total signal variance, respectively. Hence, the total main effect and pairwise effect variances equal 1% and 99% of the total signal variance, respectively.

To test the impact of increasing dimensionality on inference quality, we consider $p \in \{250, 500, 1000\}$ and keep $N = 1000$ fixed for each setting.

Main Effects Only Setting. We summarize the variable selection and estimation performances of each method (for $p = 1000$) in Table 4.1 and Table 4.2, respectively. Appendix D.4 contains the results for the remaining choices of p (see Table C.1 and Table C.2). Each method selects the majority of correct covariates. However, some methods, namely Pairs Lasso and HierLasso have a systematic bias; for all choices of p, they never select covariate 3 (the quadratic trend) since a quadratic trend has a weak linear correlation. Since the other methods can model non-linear data, they

Table 4.2: Synthetic Data Estimation Performance Results for $p = 1000$

Method	**Setting**	**Correct Selected SSE (Main)**	**Correct Not Selected SSE (Main)**	**Wrong Selected SSE (Main)**	**Correct Selected SSE (Pair)**	**Correct Not Selected SSE (Pair)**	**Wrong Selected SSE (Pair)**	**Total SSE**	**Total SSE ÷ Signal Variance**
SPAM-2Stage	Main Only	2.67	0	0.78	0	0	0.02	3.46	0.17
MARS-EMP	Main Only	0.45	0	2.68	0	0	2.39	5.51	0.28
SKIM-FA	Main Only	2.7	8.1	0	0	0	0.24	11.03	0.55
MARS-VANILLA	Main Only	16.14	0	1.56	0	0	10.33	28.02	1.4
SKIM-FA	Equal	1.54	0	0	0.29	0	0	1.82	0.09
SPAM-2Stage	Equal	1.67	0	1.07	0.41	0	2.16	5.31	0.27
MARS-EMP	Equal	0.61	0	3.84	1.7	0	2.52	8.67	0.43
MARS-VANILLA	Equal	454.88	0	3.16	21.46	0	13.22	492.72	24.64
SKIM-FA	Weak Main	0.72	0	1.37	0.61	0	0.63	3.33	0.17
MARS-EMP	Weak Main	0.67	0	5.86	3.37	0	5.63	15.52	0.78
SPAM-2Stage	Weak Main	0.16	0.2	6.69	0	18.33	0.31	25.69	1.28
MARS-VANILLA	Weak Main	23.62	0	3.18	23.16	0	15.43	65.39	3.27

can pick up this trend. Hence, they have better statistical power to detect correct covariates, improving variable selection performance.

In terms of Type I error, some methods select incorrect covariates much more frequently than others. For example, MARS consistently selects over 50 incorrect covariates for all choices of p. Since MARS induces sparsity through a greedy pruning step instead of an actual sparsity inducing penalty as in the other methods, this might explain its poorer performance.

For estimation, we only compare the non-linear methods; linear methods will artificially perform poorly since some of the effects are highly non-linear by construction. Evaluating estimation performance is more tricky since the functional ANOVA decomposition depends on the choice of measure. Unless otherwise stated, we use the joint distribution of the covariates as the measure.

Since MARS-VANILLA performs a functional ANOVA decomposition with respect to an unspecified measure, it is unclear how to interpret its main and interaction effects. One might think (and truthfully what we initially thought) that MARS-VANILLA would still return a functional decomposition close to one with respect to the actual covariate distribution. Table C.2 shows that this intuition is incorrect - the relative estimation error of MARS-VANILLA always exceeds 1! This poor estimation performance stems from not specifying the measure (and hence the target of inference), not MARS's ability in finding a model with good predictive performance. In particular, MARS-EMP, which produces the *exact same predictions* as MARS-VANILLA, yields much better estimation performance because it re-orthogonalizes the fit with respect to the actual covariate distribution.

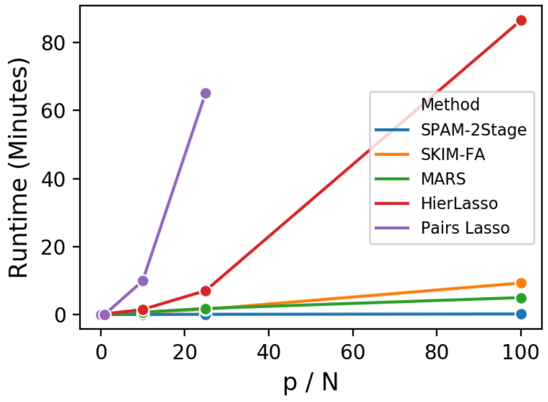
Equal Main and Interaction Effects Setting. We summarize the variable selection and estimation performances of each method in Table C.3 and Table C.4, respectively. In this setting, all methods are able to recover all 5 true covariates. Unlike in the Main Effects Only Setting, the linear methods can still select covariate 3 (the one with a quadratic main effect trend) because covariate 3 has interactions with other variables that result in stronger linear correlation with linear interaction effects.

For both estimation and variable selection, SKIM-FA performs better than the other methods.

Weak Main Effects Setting. We summarize the variable selection and estimation performances of each method in Table C.5 and Table C.6, respectively. In the setting of weak main effects, both Spam-2Stage and MARS perform very poorly relative to their performances in the other two settings. For Spam-2Stage, it only selects one correct covariate for $p = 500$ and $p = 1000$. This poor variable selection is expected; the effective R^2 for SpAM to perform variable selection is small (.01) due to the additivity assumption even though the true signal remains strong ($R^2 = .8$). The signal is locked away in the interactions!

MARS performs poorly for similar reasons due to the greedy sequential fitting procedure. In the extreme case of no main effects, for example, MARS randomly selects covariates as main effects. By random chance, MARS will eventually select a correct main effect. Once it selects a correct main effect covariate, it will then pick the correct covariate forming the (strong) interaction. Hence, MARS will need to select many incorrect covariates as main effects before identifying the true interactions.

Figure 4-3: Runtime Comparison on Simulated Data



Runtime Comparisons. We conclude this section by comparing each method in term of runtime. Apart from the two Lasso methods which take $O(p^2N)$ time, the remaining ones just depend linearly on p. When $p > N$, our method takes $O(\kappa N^3)$ while the two Lasso based methods take $O(\kappa^2 N^3)$ time, where $\kappa = p/N$. Hence, for higher-dimensional problems, our method will become much faster relative to the Lasso methods. For example, in genome-wide association studies, datasets can have N on the order of 10^3 and p on the order of 10^7 [1000 Genomes Project, 2015]. Hence, $\kappa = 10^4$, which corresponds to a potential 10^4 computational speedup factor. In Fig. 4-3, we compare the runtimes of each method as we vary p/N on simulated data. We keep N fixed at 100 and vary p from 10 to 10^4. As expected, as p/N increases, our method yields significant computational savings relative to Pairs Lasso and HierLasso.

Table 4.3: Variable Selection Performance for the Bike Sharing Dataset.

Method	**# Covariates**	**# Original Selected**	**# Wrong Selected**
SKIM-FA	1000	3	0
HierLasso	1000	3	5
SPAM-2Stage	1000	3	8
Pairs Lasso	1000	3	76
MARS	1000	3	119

Table 4.4: Estimation Performance for the Bike Sharing Dataset.

Method	**# Noise**	**Correct Selected SSE (Main)**	**Correct Not Selected SSE (Main)**	**Wrong Selected SSE (Main)**	**Correct Selected SSE (Pair)**	**Correct Not Selected SSE (Pair)**	**Wrong Selected SSE (Pair)**	**Total SSE**
SKIM-FA	1000	0.145	0.002	0	0.107	0.009	0	0.263
SPAM-2Stage	1000	0.149	0.002	0.027	0.081	0.009	0.000	0.269
MARS-EMP	1000	0.214	0.002	0.485	0.054	0.026	0.245	1.026
MARS-Vanilla	1000	6.556	0.002	0.796	0.947	0.026	1.882	10.209

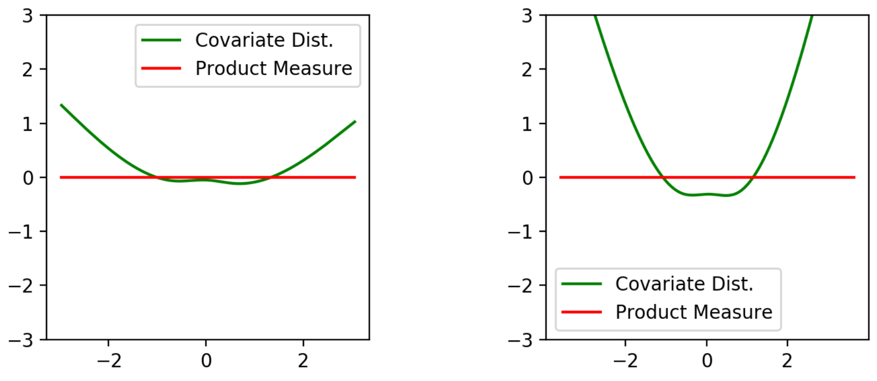
4.7.4 Evaluation on Real Data

Evaluating the methods in terms of variable selection and estimation quality is challenging because we typically do not have ground truth main and interaction effects for high-dimensional data. Similar to the evaluation procedure in Agrawal et al. [2019d], we instead take a low-dimensional dataset where N is large and p is small. We make it high-dimensional by adding synthetic random noise covariates. These two choices have several purposes. First, by fitting a regression function on the original low-dimensional dataset, standard $N^{-1/2}$ statistical convergence rates apply. Hence, for large N, a maximum-likelihood estimate of the regression function will be close to the true regression function, creating a (near) ground truth for estimation evaluation. For variable selection, the random noise covariates create a "synthetic control"; if a method selects any of the random noise covariates as a main or interaction effect, we know the method selected an incorrect covariate.

Based on these ideas, we consider the popular (low-dimensional) Bike Sharing dataset which we downloaded from the UCI Machine Learning Repository. This dataset contains 17,389 datapoints, and 13 covariates. We consider 4 continuous variables (hour, air temperature, humidity, windspeed) and use the total number of bikes rented as the response. We standardize the response by subtracting the mean and dividing by the standard deviation, and min-max standardize the covariates so that each covariate belongs to $[0, 1]$. For the proxy ground truth, we fit a pairwise interaction model consisting of all 4 main effects and 6 possible pairwise interactions.

Similar to our synthetic evaluation, we randomly subsample a total of $N = 10^3$ datapoints for training each model. To make the inference task high-dimensional we inject $p_{\text{noise}} \in \{250, 500, 1000\}$ random noise covariates, where these noise covariates

Figure 4-4: Effect of Correlated Predictors on the Main Effect for x_1



(a) $\rho = .1$ (b) $\rho = .5$

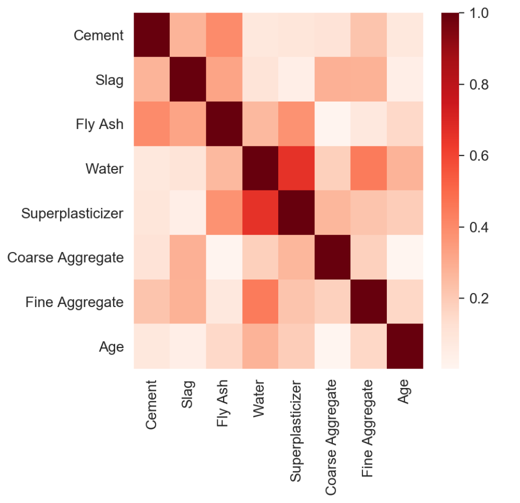
are drawn iid from a Uniform(0, 1) distribution. We report on the same variable selection and estimation metrics as in the synthetic experiments for $p_{\text{noise}} = 1000$ in Table 4.3 and Table 4.4, respectively; see Table C.8 and Table C.9 for all choices of p_{noise}. We see that again SKIM-FA has similar or much better estimation and variable selection performance relative to the other methods.

4.7.5 Impact of Correlated Predictors on the Functional ANOVA

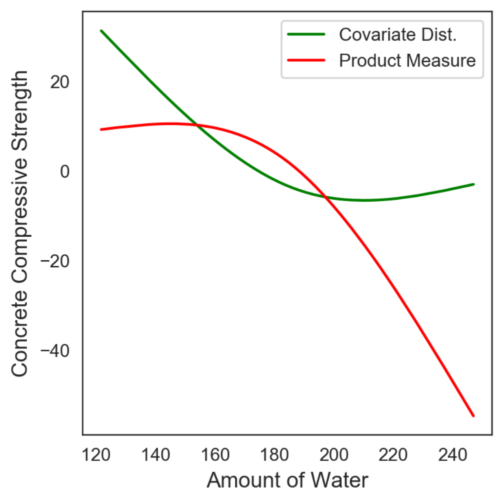
So far we have performed the functional ANOVA decomposition assuming that the covariates are jointly independent (for our synthetic data evaluation in Section 4.7.3 this was true by design). Here we show the effect correlated predictors have on the resulting decomposition. Since previous functional ANOVA methods assume product measure while Algorithm 5 provides the flexibility to select different measures, we demonstrate the practical utility of this flexibility here. To this end, we consider the simplest possible regression function with interactions; $f(x_1, x_2) = x_1 x_2$. If $x_1 \perp\!\!\!\perp x_2$, then the functional ANOVA decomposition of f with respect to $p(x_1, x_2)$ equals $x_1 x_2$. However, if x_1 and x_2 are correlated, then this is no longer the case. In particular, f can be explained well by additive effects (e.g., in the degenerate case when $x_1 = x_2$, then $f(x_1, x_2) = x_1^2$). To test this empirically, we randomly generate x_1, x_2 from a multivariate Gaussian distribution with marginal variances equal to 1 and pairwise correlation equal to ρ. We let $\rho \in \{.1, .5\}$. Fig. 4-4 shows that when ρ gets stronger, the discrepancy between a functional ANOVA decomposition with respect to $p(x_1, x_2)$ versus product measure $\mu_\otimes = N(0, 1) \otimes N(0, 1)$ increases. As expected, as the correlation increases, a quadratic-like function of x_1 and x_2 explains f increasingly well.

We perform a similar analysis above but for real data, namely the popular Concrete Compressive Strength dataset from the UCI machine learning repository. In Fig. 4-5, we plot the correlations between the 8 covariates that potentially predict the response (concrete strength). The two most correlated covariates are the amount of water and

Figure 4-5: Effect of Correlated Predictors on the Concrete Compressive Strength Dataset



(a) Absolute Value of Correlation Matrix



(b) Main Effect of Water on Strength

superplasticizer. Since the covariates have non-trivial correlations, we might expect that the functional ANOVA decomposition with respect to μ and $\mu_{\otimes}$ might be different based on Proposition 4.4.1. In Fig. 4-5 we see that there indeed is a difference; the (estimated) additive effect for water on concrete strength varies significantly depending on which measure is selected to perform the functional ANOVA decomposition.

4.8 Concluding Remarks

In this paper, we developed a new, computationally efficient method to perform sparse functional ANOVA decompositions. The heart of our procedure relied on a new kernel trick to implicitly represent non-linear interactions (Theorem 4.3.9), and a change-of-basis formula (Theorem 4.4.5) to re-express the fit in terms of an arbitrary measure. We compared our method against other methods often used to model high-dimensional with interactions. We found improved performance on both simulated and real datasets by relaxing assumptions such as linearity and the presence of strong-additive effects while still remaining competitive (or being orders of magnitude faster) in terms of runtime.

There are many interesting future research directions. One involves scaling our method to both the large N and p setting; our current method takes $O(pQN^2 + N^3)$ time which becomes problematic for large N. This cubic dependence, however, is not unique to our method but rather a fundamental obstacle faced by kernel ridge regression and Gaussian processes. Fortunately, many methods already exist to help alleviate these computational challenges with respect to N; see, for example, Gardner et al. [2018], Titsias [2009], Quiñonero Candela and Rasmussen [2005]. Another interesting direction involves applying our method to real biological datasets. In particular, an open challenge in genomics has been detecting *epistasis*, or interaction effects between genetic variants, from genome sequencing data [Maher, 2008, Aschard, 2016, Slim et al., 2018, Greene et al., 2010]. These challenges arise because p is in the millions, so the number of pairwise interactions is on the order of *trillions*. Since our method does not require explicitly generating all interactions, it has the potential to tractably detect interactions in such ultra high-dimensional data regimes.

Chapter 5

The DeCAMFounder: Non-Linear Causal Discovery in the Presence of Hidden Variables

Abstract

Many real-world decision-making tasks require learning causal relationships between a set of variables. Typical causal discovery methods, however, require that all variables are observed, which might not be realistic in practice. Unfortunately, in the presence of latent confounding, recovering causal relationships from observational data without making additional assumptions is an ill-posed problem. Fortunately, in practice, additional structure among the confounders can be expected, one such example being *pervasive confounding*, which has been exploited for consistent causal estimation in the special case of linear causal models. In this paper, we provide a proof and method to estimate causal relationships in the non-linear, pervasive confounding setting. The heart of our procedure relies on the ability to estimate the pervasive confounding variation through a simple spectral decomposition of the observed data matrix. We derive a DAG score function based on this insight, and empirically compare our method to existing procedures. We show improved performance on both simulated and real datasets by explicitly accounting for both confounders and non-linear effects.

5.1 Introduction

Many decision-making and scientific tasks require learning causal relationships between observed variables. For example, biologists seek to identify causal pathways in gene-regulatory networks, epidemiologists search for the causes of diseases in complex social networks, and data scientists in large companies or government agencies vet the biases of their machine-learning models by investigating their causal structure [Spirtes et al., 2000, Friedman et al., 2000b, Pearl, 2009, Robins et al., 2000a, Kusner et al., 2017b]. While linear causal methods are often used to model such relationships, they can fail to learn more nuanced information from the data. For instance, in time series data, some variables may exhibit seasonal variation, so that their dependence on time varies sinusoidally. In other instances, variables may exhibit non-monotonic (e.g., quadratic) dependence on their causes; an individual's health, for example, may generally be an increasing function of the amount of exercise they do, but may begin to decline with over-exercise. In either case, such trends have weak linear correlation. Hence, a linear causal model may lead to poor estimates of the DAG.

To remedy the limitations of linear methods, recent advancements in causal structure discovery have focused on recovery in the *nonlinear causal additive setting*, which assumes that each variable equals a nonlinear function of its parents plus independent noise [Hoyer et al., 2009, Mooij et al., 2009, Peters et al., 2014, Bühlmann et al., 2014]. While these non-linear methods consistently recover the true DAG, they assume that all variables are measured, i.e., no latent confounding. However, in many real-world applications, we might expect missing variables. For example, in the social sciences (e.g., economics or psychology), it can be tricky or impossible to measure potential confounders (e.g., abstract variables such as "the market", "happiness", etc.). Unfortunately, in the presence of such confounding, a DAG can no longer accurately capture the dependencies among the random variables [Richardson and Spirtes, 2002].

In the seminal work by Richardson and Spirtes [2002], the authors introduced a larger family of graphical models called *ancestral graphs* to account for confounding. While this work considers arbitrary patterns of confounding, we sometimes expect more structure about the confounders in real-world systems. For example, in gene-expression data, batch effects can lead to incorrect associations between genes [Leek and Storey, 2007]. In genome-wide association studies, ancestry differences between case and controls can create spurious correlations in disease studies [Price et al., 2006]. In finance, latent "market" and "sector" variables can explain much of the variation in stocks [Chandrasekaran et al., 2012, Fan et al., 2013]. Such confounders that have an effect on many observed variables are known as *pervasive confounders*; see Frot et al. [2019], Wang and Blei [2019], Shah et al. [2020], Chandrasekaran et al. [2012] for other real-world examples with pervasive confounders. Under the assumption that the confounders are pervasive, Frot et al. [2019], Shah et al. [2020] show how to recover the true causal structure over the observed variables, i.e., the DAG corresponding to the conditional distribution of the observed variables, given the confounders. However, they both assume that all causal relationships are *linear*.

Contributions. In this paper, we consider causal discovery in the non-linear additive noise and pervasive confounding setting. We show that the true graph is still recoverable in Theorem 5.4.1 and we provide a practical method for estimation in Section 5.4.3. The heart of our procedure relies on the ability to estimate the pervasive confounding variation through a simple spectral decomposition of the observed data matrix using a principal components analysis. This approach is similar in spirit to the *Deconfounder* algorithm proposed by Wang and Blei [2019], where the goal is not structure discovery but rather estimating average treatment effects.

Outline. The remainder of the paper is structured as follows: we start in Section 5.2 by formalizing the target of inference and the assumptions about the data generating

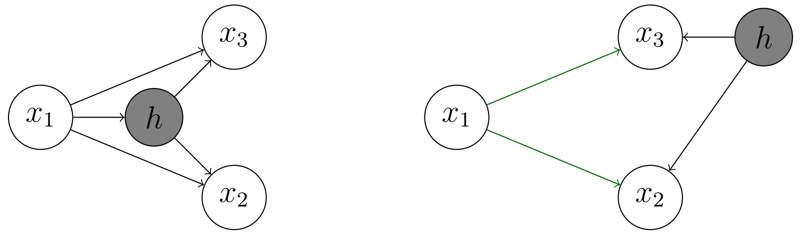


Figure 5-1: The right hand figure reparameterizes the model on the left such that the unobserved variable (i.e., shaded node) is a source in the graph. The green arrows represent the DAG corresponding to the conditional distribution $\mathbb{P}(x \mid h)$.

process. In Section 5.3, we discuss related methods for causal discovery in the presence of confounders. We conclude by comparing our method, which we outline in Section 5.4, to existing methods on both real and synthetic datasets in Section 6.5.

5.2 Problem Statement: Causal Discovery in the Presence of Confounding

Preliminaries. We would like to estimate the causal relationships between the components of $x \in \mathbb{R}^p$ given N observational datapoints $\{x^{(n)}\}_{n=1}^N$, where $x^{(n)} \overset{\text{iid}}{\sim} \mathbb{P}(x)$. Since we assume the existence of unmeasured confounders, in general there does not exist a DAG G such that $\mathbb{P}(x)$ factorizes as $\prod_{i=1}^p \mathbb{P}(x_i \mid \text{Pa}_G(x_i))$, where $\text{Pa}_G(x_i)$ denotes the parents of x_i in G [Richardson and Spirtes, 2002]. Problematically, this factorization or *Markov* property is required by many existing causal learning algorithms to correctly recover causal relationships; see, for example, Chickering [2002], Kalisch and Bühlmann [2007], Solus et al. [2020], Peters et al. [2014]. Instead, we might imagine a true underlying data-generating process $(x, h) \in \mathbb{R}^{p+K}$ such that $\mathbb{P}(x, h)$ indeed factorizes according to a DAG G^*, where h plays the role of the latent confounders. Then, the resulting *structural causal model* (SCM) associated with G^* implies that each variable x_j can be written as a function f_j of its parents in G^* and independent noise ϵ_j. Hence, a directed edge $x_i \to x_j$ in the graph represents that x_i is a direct cause of x_j.

In general, the dependencies between h and x can be complex. For example, if x_1 = age, x_2 = education, x_3 = health, h = income, we might expect that h is a cause of both x_2 and x_3, and a child of x_1; see also Fig. 5-1a. However, we prove in Proposition 5.2.1 that we can always reparameterize the model to remove some of these complex dependencies. In particular, we can construct new latent confounders that are exogenous (i.e., sources in the graph) without modifying the causal ordering of the observed variables.

Proposition 5.2.1. *Suppose that the joint distribution $\mathbb{P}$ of a random vector $(x, h) \in \mathbb{R}^{p+K}$ is Markov with respect to a causal DAG G. Then, there exists an exogenous $h' \in \mathbb{R}^K$ and joint distribution $\mathbb{P}'$ and DAG G', such that (1) $\mathbb{P}'(x, h')$ is Markov with* *respect to G', (2), $\mathbb{P}(x) = \mathbb{P}'(x)$, and (3) the partial order induced by G' equals the partial order induced by G on the subset of observed nodes $\{x_1, \cdots, x_p\}$.*

We prove Proposition 5.2.1 in Appendix D.1. The main proof idea is taking the structural causal model for (x, h), and letting h' equal the independent noise terms in h. In light of Proposition 5.2.1, we assume throughout that the latent confounders $h \in \mathbb{R}^K$ are sources, i.e., have no observed variables as parents. Hence, conditional on the confounders, $\mathbb{P}(x \mid h)$ also factorizes according to a DAG, namely the one formed by removing h and all arrows pointing out of h from the full DAG over (x, h) (i.e., the green arrows in Fig. 5-1b).

Target of Inference. In this paper, we would like to recover G^*, the DAG associated with the conditional distribution $\mathbb{P}(x \mid h)$ using only the N observed datapoints $\{x^{(n)}\}_{n=1}^N$. Since h is exogenous, the SCM simplifies to

$$x_j = f_j(x_{\mathrm{Pa}_{G^*}(j)}, h, \epsilon_j),$$

where $h \sim \prod_{k=1}^K \mathbb{P}(h_k)$, $\epsilon \sim \prod_{i=1}^p \mathbb{P}(\epsilon_i)$, $h \perp\!\!\!\perp \epsilon$. Unfortunately, due to the curse of dimensionality, the number of datapoints required to accurately estimate f_j depends exponentially on the number of parents [Gyorfi et al., 2003]. Given this statistical hardness result, we assume that each f_j has low-dimensional structure.

Causal additive models (CAMs) are a popular way of introducing low-dimensional structure to strike the balance between flexibility and statistical efficiency [Bühlmann et al., 2014]. A CAM assumes that each node can be written as an additive function of its parents and independent noise. We make this assumption and consider the following model in this paper:

$$\begin{aligned} x_j &= \sum_{x_i \in \mathrm{Pa}_{G^*}(x_j)} f_{ij}(x_i) + \sum_{k=1}^K g_{kj}(h_k) + \epsilon_j \\ &= \sum_{x_i \in \mathrm{Pa}_{G^*}(x_j)} f_{ij}(x_i) + d_j + \epsilon_j, \quad \text{where} \quad d_j = \sum_{k=1}^K g_{kj}(h_k), \end{aligned} \tag{5.1}$$

where the f_{ij} and g_{kj} are unknown functions. If we define a new noise term $\tilde{\epsilon_j} = d_j + \epsilon_j$, then this model is equivalent to an SCM with correlated errors.

Problem Statement. If we could remove the direct confounding effect d_j on each node x_j, then $x_j - d_j$ factorizes according to G^*. Hence, applying the procedures developed in Peters et al. [2014], Bühlmann et al. [2014] on the data $\{x_j^{(n)} - d_j^{(n)}\}_{n=1}^N$ recovers G^* as $N \to \infty$. Of course, the whole discussion so far and the motivation of our paper is that h is not observed. Hence, we cannot directly use existing non-linear causal discovery methods to recover G^*. The main two results of this paper (Theorem 5.4.1 and Theorem 5.4.6) show that when h has an effect on many variables, then we can construct "sufficient statistics" $s_1, \cdots, s_p$ of h using a principal component analysis (PCA) on the observed data. In particular, we show how to construct $s = (s_1, \ldots, s_p)$

and show that $\mathbb{P}(x \mid s)$ factorizes according to G^*.

Before presenting our method in Section 5.4, we first discuss related methods for learning causal structure in the presence of confounding.

5.3 Existing Causal Discovery Methods

Two approaches for learning causal structure in the presence of confounders have been proposed. The first approach models the conditional independence structure of $\mathbb{P}\left(()\, x\right)$ using an ancestral graph and then tries to recover edges in this graph. Algorithms for this include *FCI* [Spirtes et al., 2000], *RFCI* [Colombo et al., 2012], and *GSPo* [Bernstein et al., 2020]. Apart from the computational intensiveness of some of these algorithms, the price of not assuming any structure about the confounders translates into weaker identifiablity results relative to methods that exploit additional structural assumptions; see, for example, Figure 1 of [Frot et al., 2019].

The second approach assumes special structure about the confounders, namely that the confounders have an effect on a non-vanishing proportion of the observed variables as $p \to \infty$. Such variables are called *pervasive confounders.*

Definition 5.3.1. h_k is a *pervasive confounder* if $\mathrm{Ch}_{G^*}(h_k) \to \infty$ as $p \to \infty$, where $\mathrm{Ch}_{G^*}(h_k)$ denotes the children of h_k in G^*.

This type of confounding structure has been exploited for matrix completion and high-dimensional covariance estimation tasks [Cai et al., 2010, Chandrasekaran et al., 2009, Fan et al., 2013]. More recently, similar ideas have been used for learning both directed and undirected Gaussian graphical models. Directed Gaussian graphical models are a special case of the SCM in Eq. (5.1) considered in this paper, namely where f_{ij} and g_{ij} are linear and ϵ_j is Gaussian; in this case, Eq. (5.1) simplifies to the more familiar form:

$$x = Bx + \Theta h + \epsilon \quad \text{s.t.} \quad \epsilon \sim N(0, D),\ h \sim N(0, I),$$

where $B \in \mathbb{R}^{p \times p}$ and $\Theta \in \mathbb{R}^{p \times K}$ are upper-triangular matrices consisting of edge weights. Hence,

$$x = (I - B)^{-1}\epsilon + (I - B)^{-1}\Theta h. \tag{5.2}$$

In the Gaussian setting with pervasive confounders, a number of methods have been proposed to consistently estimate the covariance matrix of the conditional distribution $\mathbb{P}\left(()\, x \mid h\right)$ using only datapoints drawn from $\mathbb{P}\left(()\, x\right)$ [Chandrasekaran et al., 2012, Fan et al., 2013, Frot et al., 2019, Shah et al., 2020]. Since in this setting the covariance matrix suffices for conditional independence testing, these methods can consistently recover the true graph up to its Markov equivalence class for directed Gaussian graphical models. The methods for estimating the underlying covariance matrix in the Gaussian setting fall into two approaches, which we call the *spectral approach* and the *optimization approach* and which we describe below. Our proposed method generalizes the spectral approach.

Optimization Approach: Deconfounding the Precision Matrix. This approach starts by decomposing the precision matrix of (x, h) into the blocks

$$[\mathrm{Cov}((x,h))]^{-1} = \left(\begin{array}{c|c} J_O & J_{OH} \\ \hline J_{HO} & J_H \end{array}\right).$$

By standard theory for multivariate Gaussian distributions, $J_O \in \mathbb{R}^{p\times p}$ is the precision matrix of the conditional distribution $\mathbb{P}\left(()\, x \mid h\right)$, and hence the target of inference. The key idea for estimating J_O is to relate the observed precision matrix $[\mathrm{Cov}(x)]^{-1}$ and the target quantity J_O via Schur complements:

$$[\mathrm{Cov}(x)]^{-1} = J_O - J_{OH} J_{HH}^{-1} J_{HO}.$$

For sparse DAGs, J_O is a sparse matrix, and for $K \ll p$, $J_{OH} J_{HH}^{-1} J_{HO}$ is low-rank. Hence, when the confounders are pervasive, it is possible to estimate each component through a low-rank plus sparse matrix decomposition of $[\mathrm{Cov}(x)]^{-1}$ [Chandrasekaran et al., 2009, Chandrasekaran et al., 2012].

Spectral Approach: Deconfounding the Covariance Matrix. In contrast to the optimization approach, this approach works towards an estimate of the covariance matrix. Using the SCM in Eq. (5.2), we obtain

$$\begin{aligned} \mathrm{Cov}(x) &= (I-B)^{-1} D (I-B)^{-T} + (I-B)^{-1}\Theta\Theta^T (I-B)^{-T} \\ &= (I-B)^{-1} D (I-B)^{-T} + \tilde{\Theta}\tilde{\Theta}^T \quad \text{s.t.} \quad \tilde{\Theta} = (I-B)^{-1}\Theta \\ &= \underbrace{J_O^{-1}}_{\text{sparse inverse}} + \underbrace{\tilde{\Theta}\tilde{\Theta}^T}_{\text{low-rank}}. \end{aligned} \tag{5.3}$$

If the confounders are pervasive, then a non-neglible fraction of Θ is non-vanishing and the eigenvalues of $\tilde{\Theta}\tilde{\Theta}^T$ grow linearly with the dimension p. However, the eigenvalues of J_O are bounded. As a result, the spectrum of $\mathrm{Cov}(x)$ is dominated by the spectrum of $\tilde{\Theta}\tilde{\Theta}^T$ if its eigenvalues diverge significantly from those of J_O, i.e. they are well-separated. Hence, the first K principal components of $\mathrm{Cov}(x)$ approximate $\tilde{\Theta}\tilde{\Theta}^T$ well, and consequently provide a way to separate out J_O^{-1} [Fan et al., 2013, Wang and Fan, 2017]. In settings when the eigenvalues of J_O^{-1} and $\tilde{\Theta}\tilde{\Theta}^T$ in Eq. (5.3) are not well-separated, Shah et al. [2020] propose a modified covariance estimator and use it for learning the causal model in the linear setting. Outside of the graphical models literature, the spectral approach is also similar in flavor to the approach proposed by Wang and Blei [2019] to estimate average treatment effects in the potential outcomes framework.

The current methods described above for learning in the presence of pervasive confounders are all grounded in the linear setting. In the next section, we show how to extend these techniques to the nonlinear causal additive setting. As we will see, current methods are *not* consistent in this setting, and need careful modification. In particular, the methods above only estimate the low-rank component for the purpose of removing it, creating a new "processed" version of the data. However, in the nonlinear causal additive setting, the low-rank component plays a more complex role, as we now show.

5.4 Our Method

Our goal is to learn G^*, the DAG induced by the conditional distribution $\mathbb{P}(x \mid h)$ under the model in Eq. (5.1). In Section 5.4.1, we introduce a sufficient statistic s of the confounders h, and prove that $\mathbb{P}(x \mid s)$ also factorizes according to G^*. It therefore suffices to estimate s in order to recover G^*. In Section 5.4.2 we show that in the pervasive confounding setting, we can consistently estimate s using a principal components analysis on the observed data. We conclude in Section 5.4.3 by proposing a new score function that scores DAGs using these estimates of s.

In what follows, we assume without any loss of generality that $\pi^* := (1 \cdots p)$ is a consistent topological ordering of G^* (i.e., that x_i can only be a cause not an effect of a variable x_j when $i < j$) to simplify notation.

5.4.1 Sufficient Statistics for Recovering the DAG

Our main theorem below, which we prove in Appendix D.1, says that $\mathbb{P}(x \mid s)$ factorizes according to G^* when s equals the conditional expectation $\mathbb{E}[x \mid h]$.

Theorem 5.4.1. *There exist functions $\{r_j\}_{j=1}^p$ such that the SCM in Eq.* (5.1) *can be re-written as*

$$x_j = \sum_{x_i \in Pa_{G^*}(x_j)} f_{ij}(x_i) + [s_j - r_j(s_1, \cdots, s_{j-1})] + \epsilon_j,$$

where

$$s_j = \mathbb{E}[x_j \mid h].$$

Hence, the conditional distribution $x \mid s$ factorizes according to G^*.

Corollary 5.4.2. *G^* is identifiable from the conditional distribution $x \mid s$ when all the f_{ij} are non-linear and three-times differentiable.*

Proof. By Theorem 5.4.1, $x \mid s$ factorizes according to G^*. Since $\epsilon_j \perp\!\!\!\perp \{(x_i, r_i)\}_{i=1}^{j-1}$, the SCM in Theorem 5.4.1 is an additive noise model. By Corollary 31 of Peters et al. [2014], G^* is identifiable from observational data alone. $\square$

By Corollary 5.4.2, the task of identifying G^* reduces to one of computing or estimating s. Since we do not know h, we cannot estimate s by regressing x on h. Before showing how we can implicitly estimate s using principal components analysis in the next section, we describe the special linear case to build intuition for Theorem 5.4.1.

Example 5.4.3. In the linear setting of Eq. (5.2),

$$\begin{aligned} s &= \mathbb{E}[x \mid h] \\ &= \mathbb{E}[(I-B)^{-1}\epsilon + (I-B)^{-1}\Theta h \mid h] \\ &= (I-B)^{-1}\Theta h, \quad \text{since } \epsilon \perp\!\!\!\perp h \text{ and } \mathbb{E}[\epsilon] = 0. \end{aligned}$$

Lemma 5.4.4 below, which we prove in Appendix D.1 using a simple recursive argument, allows us to re-write each entry s_j of s in a more revealing form.

Lemma 5.4.4. *In the linear setting,*

$$\begin{aligned} s_j &= [(I-B)^{-1}\Theta h]_j \\ &= \sum_{i \in Pa_{G^*}(x_j)} B_{ij}[(I-B)^{-1}\Theta h]_i + \Theta_{j,\cdot}^T h, \end{aligned}$$

where $\Theta_{j,\cdot}$ denotes the jth row of Θ.

Recalling that $d_j = \Theta_{j,\cdot}^T h = s_j - r_j$, then

$$\begin{aligned} r_j &= \sum_{i \in \mathrm{Pa}_{G^*}(x_j)} B_{ij}[(I-B)^{-1}\Theta h]_i. \\ &= \sum_{i \in \mathrm{Pa}_{G^*}(x_j)} B_{ij} s_i \quad \text{(by Lemma 5.4.4).} \end{aligned} \tag{5.4}$$

Therefore, the SCM in Eq. (5.2) can be re-written as

$$(x_j - s_j) = \sum_{x_i \in \mathrm{Pa}_{G^*}(x_j)} B_{ij}(x_i - s_i). \tag{5.5}$$

If we can estimate s_j, then Eq. (5.5) leads to a natural data pre-processing algorithm: given an input dataset $\{x^{(n)}\}_{n=1}^N$, provide the analyst with the processed dataset $\{(x^{(n)} - s^{(n)})\}_{n=1}^N$. Given this processed dataset, the analyst can then feed this data into a linear causal learning algorithm and learn G^* up to its Markov equivalence class.

In the next section, we provide settings under which s can be estimated consistently in the non-linear setting and also provide statistical rates of convergence for our estimator.

5.4.2 Asymptotically Exact Estimates of the Sufficient Statistics

To estimate s, we reduce our problem into the linear latent factor setting studied by Fan et al. [2013]; see also Section 5.3.

Reduction to Fan et al. [2013]. Let μ^* denote the joint distribution of (ϵ, h). Assume that each random variable $x_j \in \mathcal{H}^* \subset {}^2(\mu^*)$, where $\mathcal{H}^*$ is a complete Hilbert space and ${}^2(\mu^*)$ consists of all square-integrable functions of (ϵ, h) with respect to the measure μ^*. Then, there exists M (possibly equal to infinity) set of basis functions $\{\phi_m\}_{m=1}^M$ such that

$$\begin{aligned} s_j &= \mathbb{E}[x_j \mid h] \in \mathcal{H}_M := \operatorname{span}\{\phi_1(h), \cdots, \phi_M(h)\} \quad \forall j \in [p] \\ &= \psi_j^T \Phi(h), \quad \Phi(h) := [\phi_1(h), \cdots, \phi_M(h)]^T, \end{aligned}$$

since $\mathcal{H}^*$ is separable [Rudin, 1974].

Assumption 5.4.5. Suppose there exists $M < \infty$ such that $s_j \in \mathcal{H}_M$ for all $j \in [p]$.

Under Assumption 5.4.5, we can express the nth datapoint $(x^{(n)}, h^{(n)}) \overset{\text{iid}}{\sim} \mathbb{P}(x, h)$ as

$$\begin{aligned} x^{(n)} &= \mathbb{E}[x^{(n)} \mid h^{(n)}] + [x^{(n)} - \mathbb{E}[x \mid h^{(n)}]] \\ &= \Psi\Phi(h^{(n)}) + u^{(n)}, \quad u^{(n)} := x^{(n)} - \Psi\Phi(h^{(n)}), \end{aligned} \tag{5.6}$$

where the jth row is $\Psi_{j,\cdot} = \psi_j^T$. Since $u^{(n)}$ is simply the residual after removing all of the variation explained by h, it holds that $\operatorname{Cov}(\Phi(h), u) = 0_{M\times p}$. Hence, $\Phi(h^{(n)})$, which plays the role of the *latent factors* in Fan et al. [2013], and $u^{(n)}$ are uncorrelated. Then, the model in Eq. (5.6) is equivalent to the one studied by Fan et al. [2013].

Our estimator for $s^{(n)} = \Psi\Phi(h^{(n)})$ below is mathematically equivalent to the least-squares estimator provided in Section 2.3 of Fan et al. [2013]. Although our estimator is equivalent, we derive the estimator from the perspective of matrix perturbation theory instead of least-squares. This alternative formulation provides additional geometric insights.

Derivation of the Estimator. By Eq. (5.6), we can decompose the data matrix $X \in \mathbb{R}^{N\times p}$ as,

$$\begin{aligned} X = S + U \quad \text{s.t.} \quad & X = [x^{(1)} \cdots x^{(N)}]^T, \quad S = [s^{(1)} \cdots s^{(N)}]^T, \\ & U = [u^{(1)} \cdots u^{(N)}]^T. \end{aligned}$$

The goal is to deconvolve S from the observable X. U can be viewed as playing the role of a perturbation to the matrix of interest S. To apply matrix perturbation results such as Weyl's eigenvalue theorem and the $\sin(\theta)$ theorem [Davis and Kahan, 1970], we need to understand the spectral properties of S and U. To this end, if $M < \max(N, p)$, then S has rank M. Hence,

$$\begin{aligned} \|S\|_F^2 &= \sum_{m=1}^{p} \lambda_m(S^T S) \\ &= \sum_{m=1}^{M} \lambda_m(S^T S), \quad (\text{since } \lambda_m(S^T S) = 0 \text{ for } m > M), \end{aligned} \tag{5.7}$$

where $\lambda_m(\cdot)$ denotes the mth largest eigenvalue of a matrix. If h has an effect on many variables, or equivalently if Ψ has many non-zero entries, then $\|S\|_F^2 \to \infty$. By Eq. (5.7), this implies that the top M non-zero eigenvalues of S must also go to infinity. Therefore, the spectrum of S is concentrated or *spiked* on M eigenvalues instead of being spread out on all p possible eigenvalues. Conversely, for sparse G^* with bounded node degrees, the eigenvalues of U are diffuse and bounded. Intuitively, the eigenvalues are not large because there are no "directions" that capture much of the variability or variance in U due to the sparsity condition on G^*. This discrepancy between the spectra of S and U give rise to a natural estimator to recover S: project X onto an M-dimensional subspace using principal component analysis (PCA). Since most of the variability in X occurs in the subspace spanned by S, PCA should output a subspace close to S. Recalling the many equivalent characterizations of PCA, we can find this projection by solving for the one that minimizes the empirical squared reconstruction loss:

$$\begin{aligned}
\Pi^*_{\text{PCA}} &= \operatorname*{argmin}_{\text{rank}(\Pi)\leq M} \frac{1}{N}\sum_{n=1}^{N} \|x^{(n)} - \Pi(x^{(n)})\|_2^2. \\
&= \operatorname*{argmin}_{\text{rank}(\Pi)\leq M} \|X - \Pi(X)\|_2^2 \quad \text{s.t.} \quad \Pi(X) = [\Pi(x^{(1)}) \cdots \Pi(x^{(N)})]^T \\
&= \operatorname*{argmin}_{\text{rank}(\Pi)\leq M} \|X - \Pi(X)\|_F^2.
\end{aligned}$$

Fortunately, this optimization problem has a closed-form solution, namely

$$\Pi^*_{\text{PCA}}(x) = V_M V_M^T x \quad \text{s.t.} \quad \hat{\Sigma} = V\Lambda V^T, \tag{5.8}$$

where $\hat{\Sigma}$ is the sample covariance matrix, $V\Lambda V^T$ is the eigendecomposition of $\hat{\Sigma}$, and V_M is the matrix consisting of the top M eigenvectors of V. Then, our estimate $\hat{s}_j^{(n)}$ of $s_j^{(n)} = [S]_{nj}$ equals

$$\hat{s}^{(n)} = \Pi^*_{\text{PCA}}(x^{(n)}) = V_M V_M^T x^{(n)}.$$

Theorem 5.4.6. *Under Assumption (1)-(4) of Fan et al. [2013], and Assumption 5.4.5,*

$$\max_{1\leq j\leq p, 1\leq n\leq N} \|\hat{s_j}^{(n)} - s_j^{(n)}\|_2 = O_p\left(\log(N)^{1/c}\sqrt{\frac{\log p}{N}} + \frac{N^{\frac{1}{4}}}{\sqrt{p}}\right),$$

for some constant $c > 0$.

Proof. See Corollary 1 of Fan et al. [2013]. □

Hence, ignoring log factors, we can estimate $s_j^{(n)}$ at the rate $\tilde{O}\left(\frac{1}{\sqrt{N}} + \frac{N^{\frac{1}{4}}}{\sqrt{p}}\right)$. Therefore, we need both p and N to grow for consistent estimation of $s_j^{(n)}$. Intuitively, as p grows, the eigenvalues or "spikiness" of S increases, making the projection via PCA tend towards S; for a precise statement, see Proposition 1 and 2 of Fan et al. [2013]. On the other hand, as N grows, the estimate of the covariance matrix improves and thereby reduces the estimation error of projecting via the sample covariance matrix. In the original paper by Fan et al. [2013], the authors required that $\Phi(h)$ have an effect on $O(p)$ variables. In later work by Wang and Fan [2017], the authors showed that this dependence can be weakened to $O(\sqrt{p})$; see Theorems 3.1 and 3.2 of Wang and Fan [2017]. Finally, we note that our matrix formulation in Section 5.4.2 resembles the decomposition used in robust synthetic control for counterfactual estimation [Amjad et al., 2018, Agarwal et al., 2019]. However, unlike in synthetic control, our goal is to estimate the causal graph.

Algorithm 6 summarizes our approach for estimating the matrix of sufficient statistics S. In practice, we might not know what M to use in Algorithm 6. To this end, we can use the estimator provided in Section 2.4 of Fan et al. [2013] to estimate M. As we show in Section 6.5, it is often easier to instead visually inspect the spectrum of the sample covariance matrix (i.e., the scree plot) to pick M.

5.4.3 The DeCAMFounder Score Function

In this section, we exploit the special conditional independence structure implied by Theorem 5.4.1 as well as Algorithm 6 to estimate G^* via a *score-based approach.* In general, score-based approaches consist of two parts: (1) a score-function to measure the quality of a DAG and (2) a combinatorial optimization procedure to optimize this score-function over the space of DAGs. There are many existing general-purpose procedures for (2); see, for example, Chickering [2002], Solus et al. [2020], Bühlmann et al. [2014]. Hence, we focus our attention on (1), namely developing a score function to estimate the graph in the pervasive confounding setting.

A natural starting point (based on its popularity in the linear setting) might be to use the *Bayesian information criterion* (BIC) as the score function, which penalizes DAGs based on the number of parameters [Chickering, 2002]. For linear models, computing this penalty is straightforward because the number of parameters simply equals the number of edges in the graph (corresponding to number of edge weights to be estimated) and p (the number of node noise variances to be estimated). For non-linear models, however, the number of parameters can be infinite. Hence, BIC cannot directly be applied. Instead, we use a Bayesian score, namely the marginal log-likelihood of the DAG (which the BIC approximates with a second-order Taylor series) conditional on the matrix of confounder sufficient statistics S. Specifically, we

Algorithm 6 Principal Confounding Sufficient Statistics

```
1: procedure PCSS($X, M$)
2:     $\hat{\Sigma} = \frac{1}{N} X^T X$
3:     $V, \Lambda, V^T = \text{eigen}(\hat{\Sigma})$          ▷ Eigendecomposition of $\hat{\Sigma}$
4:     $S = (V_M V_M^T X^T)^T \in \mathbb{R}^{N \times p}$     ▷ Sufficient statistics derived in Eq. (5.8)
5:     return $S$
```

score a DAG G by

$$\mathbb{P}(X \mid G, S) = \int_{\Omega_G} \mathbb{P}(X \mid G, S, \Omega_G) d\mathbb{P}(\Omega_G), \tag{5.9}$$

where $\mathbb{P}(\Omega_G)$ is a prior over the unknown set of functions $\Omega_G \coloneqq \{\{f_{ij}\}_{i\in \mathrm{Pa}_G(x_j)}, r_j\}_{j=1}^p$ defining the model (see Theorem 5.4.1). In the following, we consider the special setting where the prior $\mathbb{P}(\Omega_G)$ is given by a Gaussian process.

We define our score function, the *DeCAMFounder Score*, by placing Gaussian process priors on Ω_G as in Friedman and Nachman [2000]. This choice of prior provides flexibility to model non-linear relationships, while having desirable theoretical properties such as statistical consistency; see, for example, Chapter 7 of Rasmussen and Williams [2006]. In the following, we show in Proposition 5.4.7 that under suitable assumptions we can analytically compute Eq. (5.9) by using a GP prior. Based on the analytical formula provided in Proposition 5.4.7, we show in Corollary 5.4.8 that we can compute Eq. (5.9) in $O(p\kappa N^2 + pN^3)$ time, where κ depends on the maximum number of parents in G and the functions r_j.

Proposition 5.4.7. *Suppose that*

$$\begin{aligned} \epsilon_j &\sim \mathcal{N}(0, \sigma_j^2) \\ f_{ij} &\sim GP(0, k_{\theta_{ij}}(\cdot,\cdot)) \\ r_j &\sim GP(0, k_{\eta_j}(\cdot,\cdot)), \end{aligned}$$

where $k_{\theta_{ij}}(\cdot,\cdot)$ and $k_{\eta_j}(\cdot,\cdot)$ are positive definite kernel functions with kernel hyperparameters θ_{ij} and η_j, and ϵ_j denotes the noise terms in Eq. (5.1). For all $1 \le j \le p$, select $s_{C_j} \subset \{s_1, \cdots, s_{j-1}\}$ and a function q_j such that $r_j = q_j(s_{C_j})$ in Theorem 5.4.1. Then, Eq. (5.9) equals

$$\sum_{j=1}^p \log \mathbb{P}(X_j - S_j \mid X_{Pa_G(x_j)}, S_{C_j}) = -.5 \sum_{j=1}^p \left[\tilde{X}_j^T L_j^{-1} \tilde{X}_j - \log\det L_j - N\log(2\pi)\right], \tag{5.10}$$

where $X_j - S_j$, $L_j = K_j + \sigma_j^2 I_{N\times N}$, and $[K_j]_{nm} = k_{\theta_{ij},\eta_j}((x^{(n)}, s^{(n)}), (x^{(m)}, s^{(m)}))$ for

$$k_{\theta_{ij},\eta_j}((x,s),(\tilde{x},\tilde{s})) = \sum_{i\in Pa_G(x_j)} k_{\theta_{ij}}(x_i, \tilde{x}_i) + k_{\eta_j}(s_{C_j}, \tilde{s_{C_j}}).$$

Corollary 5.4.8. *Under the assumptions of Propositions 5.4.7, $\mathbb{P}(X \mid G, S)$ takes $O(p\kappa N^2 + pN^3)$ time to compute, where $\kappa = \max_{1\le j\le p} |Pa_G(x_j)| + |C_j|$.*

Proof. It suffices to show that $\mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{C_j})$ takes $O(\kappa N^2 + N^3)$ time to compute by Proposition 5.4.7. k_{θ_{ij},η_j} takes at most $O(\kappa)$ time to evaluate on a pair of points. Hence, the $N \times N$ kernel matrix K_j takes at most $O(\kappa N^2)$ time to compute. Computing the determinant and inverse of L_j takes $O(N^3)$ time. Finally, the matrix vector multiplication $\tilde{X}_j^T L_j^{-1} \tilde{X}_j$ takes $O(N^2)$ time. □

We prove Proposition 5.4.7 in Appendix D.1. Since s_{C_j} is not a subset of $s_{\mathrm{Pa}_{x_j}(G)}$ in general, Eq. (5.10) does not reduce into summands that only depend on the parents in G (i.e., lacks *decomposablity*). Decomposablity, however, is critical from a computational perspective, since given the marginal likelihood of a DAG G, it allows computing the marginal of a new DAG G' by only recomputing the marginal likelihood of the parent sets that changed relative to G. To this end, we assume, for computational efficiency, that each r_j in Theorem 5.4.1 only depends on $s_{\mathrm{Pa}_{G^*}(x_j)}$ in Algorithm 7.

Assumption 5.4.9. There exists some q_j such that $r_j = q_j(s_{\mathrm{Pa}_{G^*}(x_j)})$ for all $1 \leq j \leq p$.

Under Assumption 5.4.9, we can compute Eq. (5.9) in $O(p\kappa N^2 + pN^3)$ time, where κ equals two times the maximum number of parents in G by Corollary 5.4.8. We might expect that Assumption 5.4.9 holds (at least approximately) based on the following considerations: as we show in the proof of Theorem 5.4.1 in Appendix D.1, $r_j = \sum_{i \in \mathrm{Pa}_{G^*}(x_j)} \mathbb{E}[f_{ij}(x_i) \mid h]$. Since $s_i = E[x_i \mid h]$, we might expect that there exists a function z_i such that $\mathbb{E}[f_{ij}(x_i) \mid h] \approx z_i(s_i)$. When f_{ij} is a linear function, for example, then there always exists such a z_i; see Eq. (5.4) for the exact formula of z_i. Hence, if $\mathbb{E}[f_{ij}(x_i) \mid h] \approx z_i(s_i)$, then $r_j \approx \sum_{i \in \mathrm{Pa}_{G^*}(x_j)} z_i(s_i)$ is well approximated by a function of $s_{\mathrm{Pa}_{G^*}(x_j)}$. Under Assumption 5.4.9, the log marginal likelihood simplifies into a decomposable score

$$\log \mathbb{P}(X \mid S, G) = \sum_{j=1}^{p} \log \mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{\mathrm{Pa}_G(x_j)})$$

by Proposition 5.4.7. In general, we might not know a priori what kernel hyperparameters θ_{ij}, η_j, or noise variance σ_j to select in Eq. (5.10).

To this end, we follow the common approach of maximizing the marginal likelihood (also know as Type II maximum likelihood estimation) with respect to these parameters using gradient ascent instead of placing hyperpriors over these parameters [Friedman and Nachman, 2000, Section 4]. We summarize our final derived score function in Algorithm 7. In our implementation of the DeCAMFounder score, we use the probabilistic programming language *GPyTorch* [Gardner et al., 2018] to fit kernel hyperparameters via gradient ascent. In Appendix D.2, we detail the particular kernel functions used in our experiments.

Algorithm 7 The DeCAMFounder Score

```
1: procedure DECAM($X, G, M$)                          ▷ $G$ a DAG with vertex set $[p]$
2:     $S = \mathrm{PCSS}(X, M)$                       ▷ Computed from Algorithm 6
3:     $\pi$ is a topological order of $G$
4:     $mll = 0$
5:     for $i$ in $[p]$ do
6:         $mll = mll + \mathrm{argmax}_{\theta_{ij},\eta_j,\sigma_j}$[ Eq. (5.10) ]   ▷ Maximize via gradient ascent
7:     return $mll$                                    ▷ The log marginal likelihood of $G$.
```

5.5 Experiments

In this section, we start by empirically assessing how well the *Principal Confounding Sufficient Statistics* or PCSS procedure outlined in Algorithm 6 estimates the confounder sufficient statistics $s = \mathbb{E}[x \mid h]$. Although PCSS provides asymptotically exact estimates for s by Theorem 5.4.6, these estimates are noisy in the finite observational data regime. We test the impact of the estimation error on the performance of our DeCAMFounder score function (which assumes s is provided exactly) for the task of causal discovery in Section 5.5.1.2 and Section 5.5.2.2. We benchmark our DeCAMFounder score function against other popularly used score functions in the fully-observed and/or pervasive confounding setting. We discuss these benchmark methods in greater depth in Section 5.5.1.2.

We start our evaluation with simulated data so as to have access to a known ground truth DAG. In Section 5.5.2, we then also evaluate our method on an ovarian cancer dataset, which can partially be validated based on prior biological knowledge about the underlying system. We thank the authors in Frot et al. [2019] for providing a pre-processed and easily reproducible version of this dataset (as well as personal assistance). All results can be regenerated using the data and code provided in https://github.com/uhlerlab/decamfound.

5.5.1 Simulated Data

In this set of simulations we analyze the sensitivity and robustness of our method with respect to different data characteristics. To this end, we vary the following parameters: strength of confounding, linear or non-linear SCM, and dimensionality.

Controlling the Strength of Confounding. Generating non-linear (and even linear) data can be challenging. For example, without proper normalization, the variance of downstream nodes will explode (e.g., consider a line graph with edge weights equal to 2). From a signal processing perspective, if the noise variance of each node is drawn from the same distribution (e.g., as in Peters et al. [2014], Hauser and Bühlmann [2012]), then downstream nodes will typically have a higher signal-to-noise ratio than upstream nodes. To prevent such artifacts about the underlying data simulation process to skew results, we use the following normalization (some edge cases occur for source nodes or those independent of h, see Appendix D.3 for details):

- **Unit variance nodes**: $\mathrm{Var}(x_j) = 1$ for all $j \in [p]$.
- **Zero mean nodes**: $\mathbb{E}[x_j] = 0$ for all $j \in [p]$.
- **Fixed signal, confounding, and noise variances**: for all $j \in [p]$:
  - $\mathrm{Cov}(x_j, \epsilon_j) = \sigma^2_{\text{noise}}$,
  - $\mathrm{Cov}(\sum_{k=1}^{K} g_{kj}(h_k)) = \sigma^2_h$, and
  - $\mathrm{Cov}(\sum_{x_i \in \mathrm{Pa}_{G^*}(x_j)} f_{ij}(x_i)) = \sigma^2_{\text{signal}}$.

Since all nodes have unit variance, σ_h^2 (σ_{signal}^2) equals the fraction of the variation in x_j explained by h (observed variables), or equivalently, the R^2 of a model where the explanatory variables consist of all confounders (all observed variables). In our experiments, we fix $\sigma_{\text{noise}}^2 = 0.2$. In other words, if we could actually observe both x and h, then the noise in the problem is relatively small. To assess how the strength of confounding affects DAG recovery, we vary σ_h^2. Note that σ_{signal}^2 is a deterministic function of σ_h^2: $\sigma_{\text{signal}}^2 = \text{Var}(x_j) - \sigma_{\text{noise}}^2 - \sigma_h^2 = 0.8 - \sigma_h^2$, so that varying σ_h^2 implicitly varies the observed signal variance σ_{signal}^2.

Linear / Non-Linear SCM. In real data, we often expect certain characteristic patterns such as upward, seasonal, or quadratic trends. Based on this intuition, we draw each f_{ij} and g_{kj} from the set of linear trends $\{\theta x : \theta \in \mathbb{R}^p\}$, seasonal trends $\{\theta \sin(\pi x) : \theta \in \mathbb{R}^p\}$, or quadratic trends $\{\theta x^2 : \theta \in \mathbb{R}^p\}$ when generating non-linear data. For data generated from a linear SCM, we assume that all f_{ij} and g_{kj} belong to the set of linear trends $\{\theta x : \theta \in \mathbb{R}^p\}$.

Data Generation. We randomly draw G^* from an Erdös-Rényi random graph model with expected neighborhood size of 5 and consider graphs with number of observed nodes $p \in \{250, 500, 1000\}$. As in Frot et al. [2019], we assume that each confounder h_k is a direct cause of node x_j with a 70% chance. Given the graph, we randomly select a trend type for each edge (i.e., the f_{ij} and g_{kj}) with equal probability. We then randomly draw a weight $\theta \sim \text{Uniform}([-1, -.25] \cup [.25, 1])$ and appropriately scale weights of all parents simultaneously to satisfy the normalization constraints above. We finally add $N(0, \sigma_{\text{noise}}^2)$ noise to each node; see Appendix D.3 for more details and references to our `python` code. Unless otherwise stated, we draw 25 random datasets for each specific configuration of confounding strength, dimensionality, etc. We consider $K = 1$ pervasive confounders which allows us to plot how each node varies as a function of this confounder. This confounder could, for example, represent batch effects in biological experiments, the market in stock-market data, or ancestry in genome-wide association studies.

5.5.1.1 Confounder Sufficient Statistics Estimation Performance

In the following, we quantify the mean-squared estimation error of s. Based on Theorem 5.4.6, this error should decrease as N and p increase. To test this empirically, we keep the ratio of $\frac{p}{N}$ fixed at 2 (i.e., the high-dimensional case), and vary the confounding strength, dimensionality, etc. Fig. 5-2 shows the maximum mean-squared error which equals

$$\max_{j \in [p]} \frac{1}{N} \sum_{n=1}^{N} \left(s_j^{(n)} - \hat{s}_j^{(n)} \right)^2 .$$

In the linear case, we can compute s analytically (see Lemma 5.4.4). In the non-linear case, there is no analytical expression for s. However, for observed nodes x_j that only have confounders as parents, $s_j = \sum_{k=1}^{K} g_{kj}(h_k) = d_j$. Since we are simulating data and have access to h, we can compute s_j for these nodes. Thus, in Fig. 5-2, the max

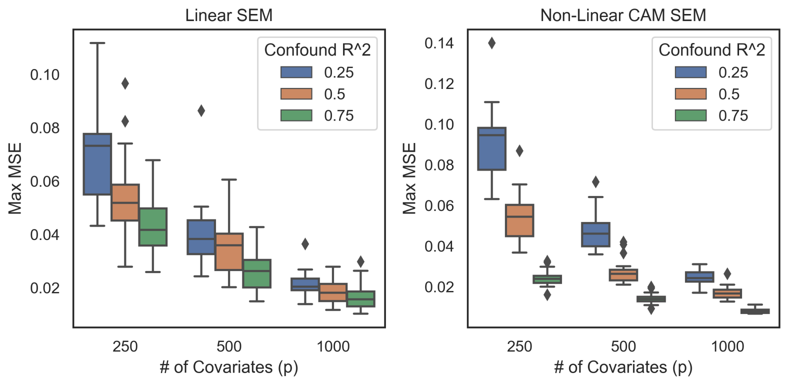


Figure 5-2: Maximum Mean-Squared Error (MSE) across all dimensions for estimating s via PCSS. Twenty-five total simulations were performed for each dataset configuration.

MSEs for the non-linear SCM case are only with respect to nodes that have only confounders as parents. From this figure, we see that the MSE decreases as p increases or when the strength of the confounders increases.

To evaluate the quality of PCSS for nodes x_j that occur further downstream (i.e., have some observed nodes as parents), we can make qualitative assessments. In particular, since we only have $K = 1$ confounders, we can plot x_j against h_1. The visual trend we see from the resulting scatterplot corresponds to the desired conditional expectation $s_j = \mathbb{E}[x_j \mid h]$. By plotting the confounder sufficient statistics estimated from PCSS on the same plot for each dimension, we can qualitatively check for a matching trend. Fig. 5-3 shows these scatterplots for a particular random data simulation and select nodes that have at least one parent. We see that PCSS indeed matches the data trend well.

5.5.1.2 Evaluation of the DeCAMFounder Score

We now transition to evaluating the quality of our DeCAMFounder score function, which depends on noisy estimates of s via PCSS to score DAGs. For this, we want to separate the merits of the score function from the underlying combinatorial opti-

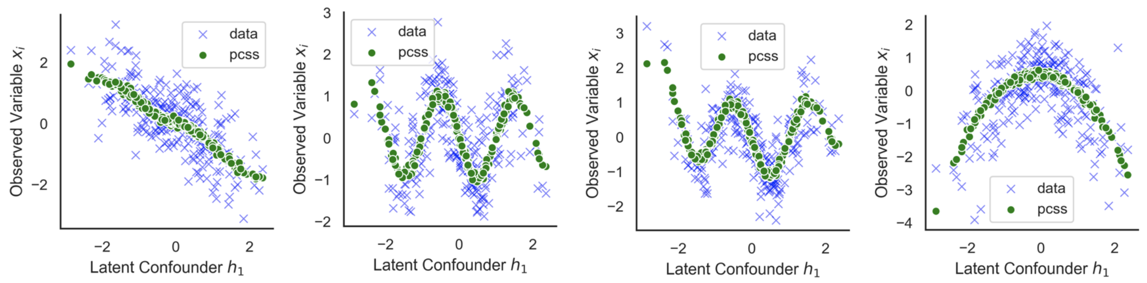


Figure 5-3: Non-linear estimation of $E[x_i \mid h]$ via PCSS for nodes with at least one parent.

mization procedure. Since most score functions used in practice are decomposable (including ours), we can reduce the evaluation to the task of effectively scoring parent sets; typical DAG optimization procedures build up a DAG by greedily optimizing over parent sets [Chickering, 2002]. As we show below, this reduction not only allows us to abstract out the challenging combinatorial optimization problem but also enables us to evaluate methods on larger scale datasets since we do not need to score the whole DAG. Our method, similar to existing kernel constraint-based methods such as RESIT [Peters et al., 2014], takes $O(N^3)$ time to score a single parent set.

Description of the Parent Set Evaluation Tasks. A local change to a parent set in greedy procedures typically involves a single node addition (i.e., adding a node to the current parent set) or deletion (i.e., removing a node from the current parent set). In the presence of latent confounding, methods that do not account for confounders are susceptible to adding spurious edges. For example, if nodes x_i and x_j have a common cause, a method that assumes no latent confounding might add an edge from x_i to x_j or vice versa with high probability, even when all the true parents of the node are *included* in the current parent set. This observation leads to a natural evaluation procedure: consider the set of all nodes $C := \{x_j : d_j \neq 0\}$ that have a direct confounding effect. Randomly select a node $x_{j'}$ from C and randomly sample M nodes $x_{r_1}, \cdots, x_{r_M}$ from C that are not parents of $x_{j'}$. Then, we can evaluate the methods in terms of scoring the following $M+1$ parent sets:

$$\begin{aligned} P_{\text{correct}} &= \text{Pa}_{G^*}(x_{j'}) \\ P_i &= \text{Pa}_{G^*}(x_{j'}) \cup \{x_{r_i}\} \quad \text{for} \quad i \in [M]. \end{aligned}$$

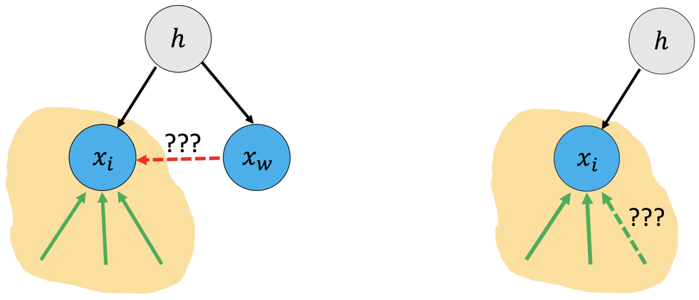
Again, we might expect that methods that do not correct for confounders score the P_i higher than P_{correct} due to spurious correlations created by the confounders. In our evaluation, we pick $M = 100$. We call this evaluation procedure the *Wrong Parent Addition Task*. We note that this task is similar to the popular "CauseEffectPairs" causality challenge introduced by Mooij et al. [2016].

The second parent set evaluation task concerns the node deletion phase. Here, we expect that linear methods may suffer by potentially removing true parent nodes. For example, if a parent node has weak *linear* correlation with the target node (e.g., as is the case with our sine and quadratic trends), then that parent would likely be pruned off using a sparsity-inducing score function such as BIC or penalized log-likelihood with an L_0 or L_1 penalty.

Similar to the Wrong Parent Addition task, we first randomly select a node $x_{j'}$ with at least one observed parent node. For each node in the parent set of $x_{j'}$, let

$$\begin{aligned} P_{\text{correct}} &= \text{Pa}_{G^*}(x_{j'}) \\ P_i &= \text{Pa}_{G^*}(x_{j'}) \setminus \{x_i\} \quad \text{for} \quad x_i \in \text{Pa}_{G^*}(x_{j'}). \end{aligned}$$

We would again like to understand if certain score functions favor incorrect P_i parent sets when the parents have non-linear effects on the target. We call this evaluation procedure the *Correct Parent Deletion* task. These two tasks are illustrated in Fig. 5-4.



(a) Wrong Parent Addition

(b) Correct Parent Deletion

Figure 5-4: Our parent set evaluation tasks. Green arrows represent the set of true edges, and red arrows indicate incorrect edges. Dotted arrows indicate a potential incorrect modifcation to the true parent set of a node.

We consider the following benchmark procedures. The first three methods are linear, and use the BIC score (with different estimates of the covariance matrix). In particular, the BIC score for Gaussian errors is a function of the target node x_j, parent set P_j, and estimated covariance matrix $\hat{\Sigma}$:

$$\mathrm{BIC}(x_j, P_j, \hat{\Sigma}) = -\left(\frac{N}{2}\log(2\pi\hat{\sigma}^2_{j|P_j}) + \frac{N}{2}\right) - .5\log(N)(|P_j| + 2), \tag{5.11}$$

where the conditional noise variance (based on Schur complements) equals

$$\hat{\sigma}^2_{j|P_j} = \hat{\Sigma}_{jj} - \hat{\Sigma}_{jP_j}\hat{\Sigma}^{-1}_{P_jP_j}\hat{\Sigma}_{P_jj}.$$

Benchmark Methods.

1. **Vanilla BIC:** scores each parent set by inputting the sample covariance matrix $\frac{1}{N}X^TX$ into Eq. (5.11).

2. **LRPS + BIC:** scores each parent using the covariance matrix estimated from a low-rank plus sparse (LRPS) decomposition of the sample precision matrix; see Frot et al. [2019]. We use the author's code in `R` to fit this decomposition, and pick the hyperparameters using cross-validation.

3. **PCSS + BIC:** scores each parent using the covariance matrix $\frac{1}{N}(X-S)^T(X-S)$, where S is the output from Algorithm 6. We pick one principal component and 3 principal components for the linear and non-linear case, respectively. This choice was based on visual inspection of the spectrum of the covariance matrix. See the Appendix.

4. **CAM:** scores each parent via Eq. (5.10) but sets S_{C_j} equal to the zero matrix (i.e., the marginal likelihood from a vanilla Gaussian process CAM model).

5. **CAM-OBS:** scores each parent via Eq. (5.10) but sets S_{C_j} equal to h (the true confounders) and $\tilde{X}_i = X_i$ instead.

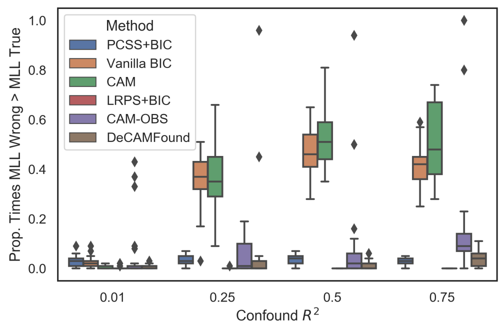
For both parent set tasks, we fix $N = 250$ and $p = 500$ but vary the strength of confounding. For the Correct Parent Deletion task, we exclude the linear trend from the set of trends when generating non-linear data to focus on the particular problem of modeling non-linearities as described above. Apart from that, the data in both settings are generated as specified at the beginning of this section. Fig. 5-5 shows the non-linear SEM results. The remaining results are shown in Fig. D-1 and Fig. D-2. In these figures, the different methods are compared using the following metric:

Prop. Times MLL Wrong > MLL True: out of the M incorrect parent sets, what proportion have scores larger than the true parent set. Lower is better here.

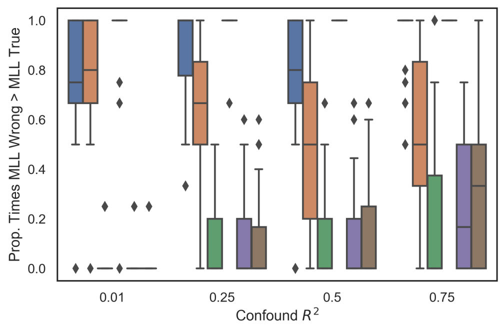
Our findings based on this analysis are: Fig. 5-5 and Fig. D-1 show that methods that account for confounders (i.e., LRPS+BIC, PCSS+BIC, CAM-OBS, DeCAM-Found), place higher probability on the correct parent set for the Wrong Parent Addition task across different settings relative to those that do not model the confounders. In addition, Fig. 5-5 and Fig. D-2 show that for the Correct Parent Deletion task, as expected, the linear methods suffer in the non-linear setting. Note that unlike PCSS+BIC, LRPS+BIC suffers even in the linear setting because it induces a very sparse graph, causing it to delete true parents. Also note that it is not possible to run CAM-OBS in practice, since it requires knowing the unobserved confouders h; we include it merely as a benchmark to understand how parent set recovery is affected by estimation error (i.e., only having finite observational data) rather than latent confounding. Interestingly, our method, which does not require knowing h, sometimes outperforms CAM-OBS. This might be because our method leverages all p observed nodes to estimate the confounding variation via PCA. CAM-OBS, on the other hand, estimates the confounding variation one node at a time (i.e., by regressing each observed node on h). Finally, all methods suffer when the confounding strength increases since the observed signal necessarily becomes weaker. We provide additional empirical results including a DAG evaluation scoring task in Appendix D.4.1.

5.5.2 Real Data: Ovarian Cancer Dataset

We use the (pre-processed) ovarian cancer RNA-seq dataset from Frot et al. [2019] which consists of $N = 247$ human samples and $p = 501$ genes. There are 15 total transcription factors (TFs), and the remaining 486 genes interact with at least one of these TFs. This dataset has a partial ground truth, namely that the 15 TFs should precede the 486 genes in the true causal ordering. The reference network for the 501 variables is obtained using Netbox, a software tool for performing network analysis using biological knowledge and community network-based approaches [Cerami et al., 2010]. A caveat, however, is that edges in this network might not have a precise probabilistic meaning (i.e., in either a causal or undirected probabilistic graphical model sense).

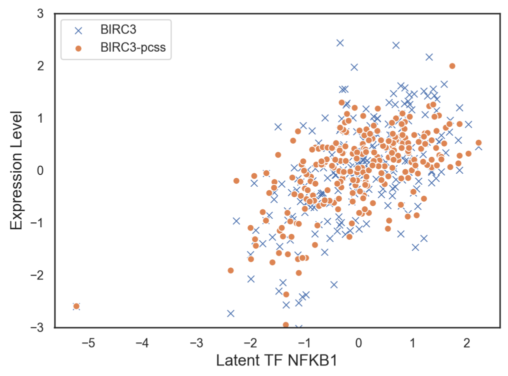


(a) Wrong Parent Addition Task



(b) Correct Parent Deletion Task

Figure 5-5: Results for the Wrong Parent Addition and Correct Parent Deletion tasks (lower values on the y-axis are better for both tasks). The data are generated according to a non-linear SEM. 25 total simulations per dataset configuration were performed.



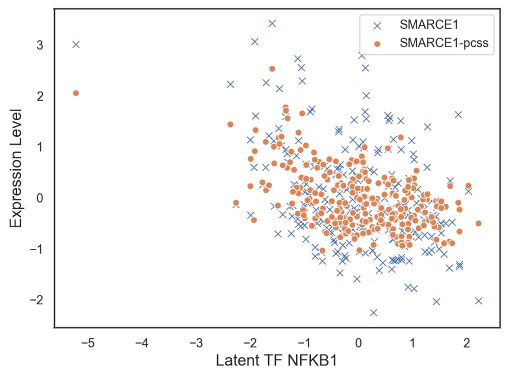
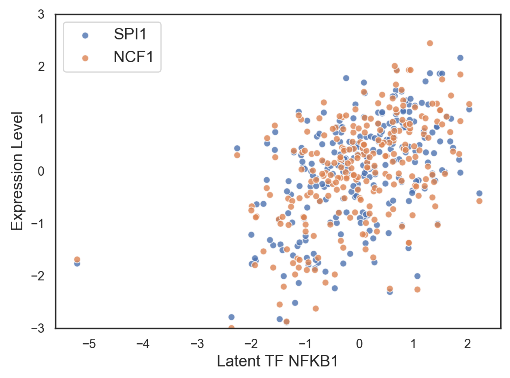


Figure 5-6: The two genes most positively and negatively correlated with the transcription factor NFKB1, respectively. GENE-pcss refers to the total latent confounding variation estimated for that gene via PCSS. GENE refers to the observed values of the gene.

One way the authors in Frot et al. [2019] benchmarked their methods was by counting the number of directed edges from TFs to genes that agree with the edges in NetBox. The authors stated that confounding can be expected in this dataset due to unobserved transcription factors or batch effects. We here take a different approach for the evaluation and instead *explicitly* create confounders by design. In particular, we remove all 15 TFs from the dataset and assume that we only observed the remaining 486 genes.

5.5.2.1 Estimating the Confounding Variation

By removing the TFs, we can evaluate the methods in a similar fashion as the simulated experiments since we know the true values of the TFs. In the following, we first assess our ability to estimate s. Suppose that an observed gene x_j is strongly correlated with one of the 15 latent TFs t_k. Then, we would expect that $\mathbb{E}[x_j^{(n)} \mid t_k = t_k^{(n)}] \approx$



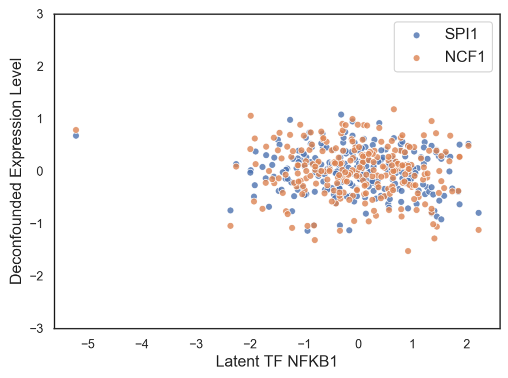


Figure 5-7: Both genes have a correlation greater than 0.4 with NFKB1 (left hand plot). After subtracting out the confounding variation estimated using PCSS for each gene (denoted as "deconfounded" expression level), the genes are no longer correlated with the unobserved transcription factor NFKB1.

$\mathbb{E}[x_j^{(n)} \mid h = h^{(n)}] = s_j^{(n)}$. Since we know t_k, we can produce a similar plot as in Fig. 5-3. To this end, we look at NFKB1 which is a TF known to be associated with ovarian cancer [Harrington and Annunziata, 2019]. In Fig. 5-6, we look at the highest positive and negatively correlated genes with NFKB1, which are BIRC3 and SMARCE1 respectively. We see that the estimated confounding variation for each gene estimated from PCSS correlates well with the unobserved TF NFKB1. This suggests that the confounding variation estimated from PCSS corresponds to true confounding (i.e., in this instance NFKB1).

5.5.2.2 Parent Recovery Performance

Unlike in the simulated dataset experiments, we do not know the true causal graph. In Frot et al. [2019], the authors used the graph structure predicted by NetBox as one way to evaluate their methods. We follow a similar strategy in order to replicate the parent evaluation tasks from the simulated data experiments. We do this by first treating the graph outputted by NetBox as the ground truth undirected graph. Since we do not know the true parent sets for each node, we use the neighborhood set in the undirected graph as a proxy. Hence, to find genes likely to have spurious edges due to the removed TFs, we consider the set of gene pairs that are conditionally independent given the TFs and neighborhood sets of each node but marginally dependent given just the neighborhood sets. For each ordered pair of genes, we select the first element to be the target node and the second element as the spurious parent to add. We take the neighborhood set of the target nodes as the proxy for the true parent set, and score this neighborhood set relative to the neighborhood set appended with the wrong parent candidate. For each method, we compute the proportion of times that the neighborhood set appended with the wrong parent candidate has a higher score than just the neighborhood set (i.e., analogous to the Wrong Parent Addition task). Since the 15 removed TFs have high node degrees, many gene pairs satisfy the criteria above (which would require us to score about 70K possible parent sets). Instead, we focus

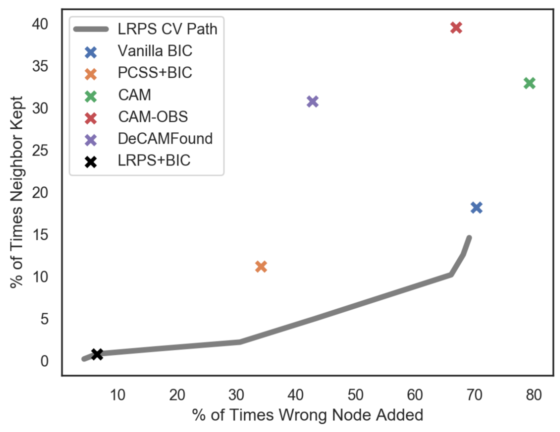


Figure 5-8: The x-axis denotes the proportion of times a method scored the incorrect parent appended to the neighborhood set higher than just the neighborhood set of a node (i.e., our proxy for the true parent set). The y-axis denotes the proportion of times a method scored the full neighborhood set higher than the neighborhood set after removing out one of the neighbors. For LRPS, we provide the intermediate results for each covariance matrix outputted along its cross-validation path. The black 'x' for LRPS corresponds to the performance of the covariance matrix selected based on cross-validation. We compute the *positive likelihood ratio* for each method which equals the ratio between our proxy for the true positive rate (i.e., the y-axis) and false positive rate (i.e., the x-axis). These ratios are as follows: Vanilla BIC=.26, PCSS+BIC=.33, CAM=.42, CAM-OBS=.59, DeCAMFound= .72, LRPS+BIC=.12.

on a subset of edges with at least one strong TF confounder. In particular, we require that each node in the formed edge have correlation greater than 0.4 with one of the 15 removed TFs. This choice makes it more likely that one of the 15 left out TFs is the actual confounder rather than e.g., batch effects, and results in roughly 1000 parent sets to score. One such gene pair with a strong TF confounder is illustrated in Fig. 5-7. We see how removing the confounder sufficient statistics from each gene results in removing the shared confounding effect of the strongest transcription factor NFKB1.

To also evaluate a method's statistical power / ability to detect edges, we randomly sample 500 edges from the NetBox undirected graph. For each edge, we randomly select one of the edges as the target node and consider removing the second node from the target node's neighborhood set. In particular, we check if a method scores the parent set with the second node removed from the neighborhood set lower than the full neighborhood set (i.e., similar to the Correct Parent Deletion task).

The results for both tasks are summarized in Fig. 5-8. We see that LRPS produces a very sparse graph, and hence almost never adds a wrong parent. However, it almost always removes true edges, which comes at the expense of statistical power. Our method has the highest power per unit of false positives. Rather surprisingly, CAM-OBS, which explicitly uses the 15 held out transcription factors, selects wrong parents more often than our method. This might be the result of having additional (pervasive) confounders beyond the 15 TFs we introduced by design, as raised by Frot et al. [2019].

5.6 Conclusion

In this paper, we showed that we can identify the causal graph G^* among the observed nodes in the setting of non-linear effects and pervasive confounders. We proposed a practical algorithm, the DeCAMFounder, to consistently estimate G^* using Gaussian processes. Since the DeCAMFounder explicitly accounts for confounders and non-linear effects, we found improved performance on both simulated and real datasets relative to existing methods.

There are a number of interesting future directions. One involves improving scalability; our current method takes $O(N^3)$ time to score a single parent set since we must invert an $N \times N$ kernel matrix. An interesting direction is to improve computation time by making use of existing techniques for large-scale kernel matrix approximation; see, for example, Titsias [2009], Drineas and Mahoney [2005], Agrawal et al. [2019a]. Another interesting direction is to explore how to handle selection bias. In Frot et al. [2019], for example, the authors showed how to account for selection bias and pervasive confounders in the linear setting. It would be interesting to explore whether a similar idea could be used to extend the DeCAMFounder to handle selection bias.

Chapter 6

ABCD-Strategy: Budgeted Experimental Design for Targeted Causal Structure Discovery

Abstract

Determining the causal structure of a set of variables is critical for both scientific inquiry and decision-making. However, this is often challenging in practice due to limited *interventional* data. Given that randomized experiments are usually expensive to perform, we propose a general framework and theory based on optimal Bayesian experimental design to select experiments for *targeted causal discovery*. That is, we assume the experimenter is interested in learning some function of the unknown graph (e.g., all descendants of a target node) subject to design constraints such as limits on the number of samples and rounds of experimentation. While it is in general computationally intractable to select an optimal experimental design strategy, we provide a tractable implementation with provable guarantees on both approximation and optimization quality based on submodularity. We evaluate the efficacy of our proposed method on both synthetic and real datasets, thereby demonstrating that our method realizes considerable performance gains over baseline strategies such as random sampling.

6.1 Introduction

Determining the causal structure of a set of variables is a fundamental task in causal inference, with widespread applications not only in artificial intelligence but also in scientific domains such as biology and economics [Friedman et al., 2000a, Pearl, 2003, Robins et al., 2000b, Spirtes et al., 2000]. One of the most common ways of representing causal structure is through a *directed acyclic graph* (DAG), where a directed edge between two variables in the DAG represents a direct causal effect and a directed path indicates an indirect causal effect [Spirtes et al., 2000].

Causal structure learning is intrinsically hard, since a DAG is generally only identifiable up to its *Markov equivalence class* (MEC) [Verma and Pearl, 1991, Andersson et al., 1997]. Identifiability can be improved by performing *interventions* [Hauser and Bühlmann, 2012, Yang et al., 2018], and several algorithms have been proposed for structure learning from a combination of observational and interventional data [Wang et al., 2017, Hauser and Bühlmann, 2012, Yang et al., 2018]. Since experiments tend to be costly in practice, a natural question is how principled *experimental design* (i.e., selection of intervention targets) can be leveraged to maximize the performance of these algorithms under budget constraints.

Seminal works by Tong and Koller [2001] and Murphy [2001] showed that experimental design can improve structure recovery in causal DAG models. However, these methods assume a basic framework in which experiments are performed one sample at a time. In practice, experimenters often perform a batch of interventions and collect samples over multiple rounds of experiments; and they must also factor in budget and feasibility constraints, such as on the number of unique interventions that can be performed in a single experiment, the number of experimental rounds, and the total number of samples to be collected. In genomics, for instance, genome editing technologies have enabled the collection of batches of large-scale interventional gene expression data [Dixit et al., 2016]. An imminent problem is understanding how to optimally select a batch of interventions and allocate samples across these interventions, over multiple experimental rounds in a computationally tractable manner.

Since the initial works by Tong and Koller [2001] and Murphy [2001], there have been a number of new experimental design methods under budget constraints [Hauser and Bühlmann, 2014, Ghassami et al., 2018, Ness et al., 2018]. These methods suffer from two drawbacks: (1) poor computational scaling [cf. Ness et al., 2018] or (2) strong assumptions including the availability of infinite observational and/or interventional data from each experiment [cf. Hauser and Bühlmann, 2014, Ghassami et al., 2018]. Since it is difficult to learn the correct MEC in a limited sample setting, it is desirable to use interventional samples not only to improve identifiability but also to help distinguish between observational MECs.

Generalizing the frameworks in [Tong and Koller, 2001, Murphy, 2001, Cho et al., 2016, Hauser and Bühlmann, 2014, Ness et al., 2018], we assume the experimenter is interested in learning some function $f(G)$ of the unknown graph G. Returning to gene regulation, one might set $f(G)$ to indicate whether some gene X is downstream of some gene Y, i.e. if X is a *descendant* of Y in G. Using targeted experimental design, all statistical power is placed in learning the target function rather than being agnostic to recovering all features in the graph. In addition, we also explicitly take into account that only finitely many samples are allowed in each round, and work under various budget constraints such as a limit on the number of rounds of experimentation.

We start by reviewing causal DAGs in Section 6.2 and then propose an entropy-based score function that generalizes the one by Tong and Koller [2001] and Murphy [2001] in Section 6.3. Since optimizing this score function is in general computationally intractable, we propose our *ABCD-Strategy* consisting of approximations via *weighted importance sampling* and greedy optimization in Section 6.4. We also provide guarantees for this algorithm based on *submodularity*. Further, in contrast to earlier score functions, we show that our proposed score function is provably *consistent*. Finally, in

Section 6.5 we demonstrate the empirical gains of the proposed method over random sampling on both synthetic and real datasets.

6.2 Preliminaries

Causal DAGs. Let $G = ([p], A)$ be a *directed acyclic graph* (DAG) with vertices $[p] := \{1, \dots, p\}$ and directed edges A, where $(i, j) \in A$ represents the arrow $i \to j$. A *linear causal model* is specified by a DAG G and a corresponding set of edge weights $\theta \in \mathbb{R}^{|A|}$. Each node i in G is associated with a random variable X_i. Under the *Markov Assumption*, each variable X_i is conditionally independent of its nondescendants given its parents, which implies that the joint distribution factors as $\prod_{i=1}^{p} \mathbb{P}\big(X_i \mid \mathsf{Pa}_G(X_i)\big)$, where $\mathsf{Pa}_G(X_i)$ denotes the parents of node X_i [Spirtes et al., 2000, Chapter 4]. This factorization implies a set of *conditional independence* (CI) relations; the *Markov equivalence class* (MEC) of a DAG G consists of all DAGs that share the same CI relations [Lauritzen, 1996, Chapter 3]. The *essential graph* $\mathrm{Ess}(G)$ is a partially oriented graph that uniquely represents the MEC of a DAG by placing directed arrows on edges consistent across the equivalence class and leaves the other edges undirected [Andersson et al., 1997].

Learning with Interventions. Let *intervention* $I \subseteq [p]$ be a set of intervention targets. Intervening on I removes the incoming edges to the random variables $X_I := (X_i)_{i \in I}$ in G and sets the joint distribution of X_I to a new interventional distribution $\mathbb{P}^I$. The resulting *mutilated graph* is denoted by G^I. A typical choice of $\mathbb{P}^I$ is the product distribution $\prod_{i \in I} f_i(X_i)$, where each $f_i(X_i)$ is the probability density function for the intervention at X_i. We denote by $\mathcal{I}^* := \{I_1, \cdots, I_K\}$ the set of all $K \in \mathbb{N}$ allowed interventions and by $\mathcal{I} \subseteq \mathcal{I}^*$ the subset of selected interventions. An intervention $I = \emptyset$ indicates observational data. We assume that $\mathcal{I}^*$ is a *conservative family* of interventions, i.e., for any $i \in [p]$, there exists some $I_j \in \mathcal{I}^*$ such that $i \notin I_j$ [Hauser and Bühlmann, 2012]. Given a conservative family of targets $\mathcal{I}$, two DAGs G_1 and G_2 are *$\mathcal{I}$-Markov equivalent* if they are observationally Markov equivalent and for all $I \in \mathcal{I}$, G_1^I and G_2^I have the same skeleta [Hauser and Bühlmann, 2012, 2015]. The set of *$\mathcal{I}$-Markov equivalent* DAGs can be represented by the *$\mathcal{I}$-essential graph* $\mathrm{Ess}^{\mathcal{I}}(G)$, a partially directed graph with at least as many directed arrows as $\mathrm{Ess}(G)$ [Hauser and Bühlmann, 2012, Theorem 10].

Bayesian Inference over DAGs. In various applications, the goal is to recover a function $f(G)$ of the underlying causal DAG G given a mix of n independent observational and interventional samples $D = \{(X_{mi}, I^{(m)}) : I^{(m)} \in \mathcal{I}^*, m \in [n], i \in [p]\}$. For example, we might ask whether an undirected edge (i, j) is in A, or we might wish to discover which nodes are the parents of a node i. We can encode our prior structural knowledge about the underlying DAG through a *prior* $\mathbb{P}(G)$. The *likelihood*

$\mathbb{P}(D \mid G)$ is obtained by marginalizing out θ:

$$\begin{aligned}\mathbb{P}(D \mid G) &= \int_\theta \mathbb{P}(D, \theta \mid G)\, d\theta \\ &= \int_\theta \mathbb{P}(D \mid \theta, G)\mathbb{P}(\theta \mid G)\, d\theta\end{aligned}$$

and can be computed in closed-form for certain distributions [Geiger and Heckerman, 1999, Kuipers et al., 2014]. Applying Bayes' Theorem yields the *posterior distribution* $\mathbb{P}(G \mid D) \propto \mathbb{P}(D \mid G)\mathbb{P}(G)$, which describes the state of knowledge about G after observing the data D. Given the posterior, we can then compute $\mathbb{E}_{\mathbb{P}(G|D)} f(G)$, the posterior mean of some target function $f(G)$. Note that when f is an indicator function, this quantity is a posterior probability.

6.3 Optimal Bayesian Experimental Design

Our goal is to learn some feature $f(G)$ of the unknown graph through experimental design under budget constraints such as limited number of experimental rounds. In principle, this question can be answered using *optimal Bayesian experimental design*, namely by selecting the experiment that maximizes the expected value of some *utility function* U, where the expectation is with respect to hypothetical data generated according to our current beliefs [Chaloner and Verdinelli, 1995]. Here, the expected utility function U is a function defined on multisets of $\mathcal{I}^*$:

Definition 6.3.1. The *expected utility* $U^f(\xi; D)$ of a multiset of interventions $\xi \in \mathbb{Z}^{\mathcal{I}^*}$ for learning a function $f(G)$ given currently collected data D is given by

$$\begin{aligned}U^f(\xi; D) &= \mathbb{E}_{y \sim \mathbb{P}(y|D,\xi)}\ U^f(y, \xi; D) \\ &= \mathbb{E}_{G,\theta|D}\mathbb{E}_{y|G,\theta,\xi}\ U^f(y, \xi; D), \quad y \in \mathbb{R}^{|\xi|},\end{aligned} \tag{6.1}$$

where $U^f(y, \xi; D) \in \mathbb{R}$ is a function measuring the utility of observing additional samples y from a proposed design ξ and $|\xi| := \sum_{I \in \mathcal{I}^*} |\#\text{ times } I \text{ in } \xi|$. The *optimal Bayesian design* ξ^* under a set of design constraints C is given by

$$\xi^* \in \operatorname{argmax} \xi \in \mathbb{Z}^{\mathcal{I}^*} \cap C\ U^f(\xi; D). \tag{6.2}$$

We denote samples collected from such an optimal strategy ξ^* by D_{ξ^*}

In Definition 6.3.1, y is distributed according to our current beliefs $\mathbb{P}(y \mid D, \xi) = \mathbb{E}_{G,\theta|D}\left[\mathbb{P}(y \mid G, \theta, \xi)\right]$, a *mixture distribution* over (G, θ), and the utility function $U^f(y, \xi; D)$ is averaged over this distribution. There are many potential choices for $U^f(y, \xi; D)$, a popular one being *mutual information.* Tong and Koller [2001], Cho et al. [2016] and Murphy [2001] propose optimizing mutual information for the problem of recovering the full graph. More precisely, they consider the problem where $f(G) = G$ in the active learning setting, where the experimenter can adaptively collect one sample at a time. We here extend their framework to general functions $f(G)$ and the *batched setting*, where multiple samples are collected at once and the total number of batches is fixed by the experimenter. Hence, U^f must be defined on multisets instead of elements of $\mathcal{I}^*$ since multiple samples (i.e., interventions of the same type) may be collected in each batch. Note that the difficulty in solving Eq. (6.2) stems from the constraint set C, which renders this optimization problem combinatorial.

Recently, Ness et al. [2018] proposed a Bayesian experimental design method to work in the batched setting. The authors proposed a utility function based on the expected number of additional edges that could be oriented by performing a particular intervention given the observational MEC. This function is similar to the one proposed by Hauser and Bühlmann [2014] and Ghassami et al. [2018], in which interventions are chosen that fully identify the causal network given the MEC. Unfortunately, the algorithm in Ness et al. [2018] has *factorial* dependence on the size of the batch; in addition, we prove in Appendix E.1.3 that their proposed utility function is, in general, not *consistent*; see Definition 6.3.3 for a definition of consistency.

We therefore follow the approach taken by Tong and Koller [2001] and Murphy [2001] and consider the utility function $U^f(y, \xi; D)$ to be given by mutual information. Maximizing the mutual information is equivalent to picking the set of interventions that leads to the greatest expected decrease in entropy of $f(G)$. The mutual information utility function is given by

$$U^f_{\text{M.I.}}(y, \xi; D) \coloneqq H(f \mid D) - H(f \mid D, y = y, \xi), \tag{6.3}$$

where the *entropy* $H(f \mid D)$ equals

$$\begin{aligned} &\sum_{e: f(G)=e} -\mathbb{P}(f(G) = e \mid D) \log \mathbb{P}(f(G) = e \mid D), \\ \text{and } \ &\mathbb{P}(f(G) = e \mid D) = \mathbb{E}_{\mathbb{P}(G|D)} \mathbb{1}(f(G) = e), \\ &\mathbb{P}(G \mid D) \propto \int_\theta \mathbb{P}(D \mid G, \theta) \mathbb{P}(\theta \mid G) \mathbb{P}(G). \end{aligned}$$

To better understand the behavior of $U^f_{\text{M.I.}}$, we prove the following proposition, which highlights the behavior of $U^f_{\text{M.I.}}$ in the limit of infinite samples per intervention; this is the setting studied by Hauser and Bühlmann [2014] and Ghassami et al. [2018].

Proposition 6.3.2. *Suppose that the Markov equivalence class $\mathcal{G}$ of G^* is known and the goal is to identify the underlying true DAG G^*. Furthermore, assume a uniform prior over $\mathcal{G}$, infinite samples per intervention $I \in \mathcal{I}$, and at most K unique interventions per batch as in Ghassami et al. [2018]. Then, $U_{M.I.}$ selects the interventions*

$$\mathcal{I}_{M.I.} \in \operatorname{argmin} |\mathcal{I}| \leq K \ \frac{1}{|\mathcal{G}|} \sum_{G \in \mathcal{G}} \log_2 |Ess^{\mathcal{I}}(G)|$$

where $|Ess^{\mathcal{I}}(G)| \coloneqq |\{G' \in \mathcal{G} : G' \in Ess^{\mathcal{I}}(G)\}|$.

This result (proof in Appendix) shows that in the limiting case, mutual information selects interventions that lead to the finest expected log $\mathcal{I}$-MEC sizes. This limiting behavior of mutual information parallels what graph-based score functions do, such as the ones considered by Hauser and Bühlmann [2014], Ghassami et al. [2018] and Ness et al. [2018], that invoke the *Meek Rules* [Verma and Pearl, 1992] to select interventions that orient the most number of edges in the $\mathcal{I}$-essential graphs (in expectation).

A score function based on mutual information is particularly appealing since it not only has desirable properties in the infinite sample setting, but also does not require the MEC to be known, naturally handling the case of finite sample sizes. In particular, a score function based solely on Meek rules will not pick the same intervention twice by definition, since repeating the same intervention does not improve identifiability. As a result, adapting graph-based score functions in the finite sample regime requires first constructing an intervention set and then allocating samples instead of jointly picking and allocating samples. Mutual information, on the other hand, can pick the same intervention twice; for example if a particular intervention is very informative, selecting it twice and allocating more samples to it might lead to a greater expected decrease in entropy than a new intervention.

6.3.1 Budget Constraints

So far we have not specified the constraint set C in Eq. (6.2). To this end, we assume that the experimenter has a total of N samples to allocate across B batches. While one could try to optimize the partition of N samples across batches, in this work we study the simpler case where each batch b, $1 \leq b \leq B$, receives a pre-specified amount of samples N_b with $\sum_b N_b = N$. For simplifying notation we assume throughout that $N_b = \frac{N}{B}$. We leave the study of adaptive batch sizes N_b for future work. The constraint set then equals,

$$C_{N,b} \coloneqq \{\xi \in \mathbb{Z}^{\mathcal{I}^*} : |\xi| = N_b\}, \tag{6.4}$$

where the subscripts on C emphasize the dependence on N and b. Then, the optimal design in batch b is obtained by solving the following combinatorial optimization problem:

$$\xi_b^* \in \operatorname{argmax} \xi \in \mathbb{Z}^{\mathcal{I}^*} \cap C_{N,b}\, U(\xi; D_{b-1}), \tag{6.5}$$

where $D_{b-1} \coloneqq [D_{\xi_1^*}, \cdots, D_{\xi_{b-1}^*}]$ is all the data collected at the start of batch b and $U(\xi; D_{b-1})$ could, for example, be the mutual information defined in Eq. (6.3). Notice that while a particular form of $U(\xi; D_{b-1})$ is provided in Definition 6.3.1, $U(\xi; D_{b-1})$ need not necessarily be a Bayesian utility function to fit within the framework of Eq. (6.5).

We now define a natural notion of consistency for any experimental design method that can be cast as an optimization routine in the form of Eq. (6.5). Since the consistency of a utility function should not depend on a specific constraint set such as $C_{N,b}$, Definition 6.3.3 assumes the constraint set is arbitrary and set by the practitioner.

Definition 6.3.3. Suppose $f(G)$ is identifiable in $\mathrm{Ess}^{\mathcal{I}^*}(G^*)$, where G^* is the true unknown DAG. Let $C_{N,b}$, $1 \leq b \leq B$ denote the constraints in batch b. A utility function $U(\xi)$ is *budgeted batch consistent* for learning a target feature $f(G)$ if

$$\mathbb{P}\left(f(G) \mid D_B\right) \xrightarrow{\mu^* \text{ a.s.}} \mathbb{1}(f(G) = f(G^*)),$$

as $N, B \to \infty$, where μ^* is the law determined by the true unknown causal DAG (G^*, θ^*)

Theorem 6.3.4. *$U^f_{M.I.}$ is budgeted batch consistent for* single-node interventions*, i.e., when $\mathcal{I}^* = \{\{1\}, \cdots, \{p\}\}$.*

Remark 1. While Theorem 6.3.4 may not be surprising (proof in the Appendix), we found that various utility functions that seem natural and have been proposed in earlier work are not consistent in the budgeted setting. In particular, in Appendix E.1.3 we show that the utility function proposed by Ness et al. [2018] is not consistent for single-node interventions. The main issue is that there are DAGs and constraint sets for which the same interventions keep getting selected, instead of selecting new interventions to fully identify $f(G)$.

6.4 Tractable Algorithm

While Section 6.3 provides a general framework for targeted experimental design, there are several computational challenges that we have not yet addressed. The first challenge is computing $U^f_{\text{M.I.}}(\xi; D)$. This objective function requires summing over an exponential number of DAGs and marginalizing out the edge weights θ. In this section, we discuss how to approximate $U^f_{\text{M.I.}}(\xi; D)$ by sampling graphs (either through MCMC or the *DAG-bootstrap* Friedman et al. [1999]) and using the maximum likelihood estimator of θ for each graph. Taken together, these approximations not only allow the mutual information score to be computed tractably but also lead to desirable optimization properties. In particular, we prove in Theorem 6.4.1 that our approximate utility function is submodular. This property enables optimizing the approximate objective in a sequential greedy fashion with provable guarantees on optimization quality.

6.4.1 Expectation over (G, θ)

A serious problem from a computational perspective is the expectation over (G, θ) in Definition 6.3.1. Since the number of DAGs grows *superexponentially* with p, enumerating all possible DAGs is intractable. Instead, in each batch b, we propose to sample T graphs according to the posterior $\mathbb{P}(G \mid D_{b-1})$. This can be done using a variety of different *Markov chain Monte-Carlo* (MCMC) samplers; see for example Heckerman et al. [1997], Ellis and Wong [2008], Friedman and Koller [2003], Niinimaki et al. [2016], Kuipers and Moffa [2017], Madigan and York [1995], Grzegorczyk and Husmeier [2008], Agrawal et al. [2018]. An alternative that is often faster but still achieves good performance, is approximating the posterior via a high-probability candidate set of T DAGs $\hat{\mathcal{G}}_T$ [Heckerman et al., 1997, Friedman et al., 1999]. While there are many ways to build up this set, a popular approach is through the *nonparametric DAG bootstrap* [Friedman et al., 1999]. The main idea is to subsample the data (with replacement) T times and fit a DAG learning algorithm to each of the generated datasets to construct $\hat{\mathcal{G}}_T$. Each $G \in \hat{\mathcal{G}}_T$ can then be weighted according to the ratio of unnormalized posterior probabilities,

$$w_{G,D} \coloneqq \frac{\mathbb{P}(G)\mathbb{P}(D \mid G)}{\sum_{G \in \hat{\mathcal{G}}_T} \mathbb{P}(G)\mathbb{P}(D \mid G)} \tag{6.6}$$

to form an approximate posterior $\hat{\mathbb{P}}(G) \coloneqq w_{G,D}\mathbb{1}(G \in \hat{\mathcal{G}}_T)$. The DAG learning algorithm used for this purpose must be able to handle a mix of observational and interventional data. Two recent methods that have been developed for this purpose are given in Hauser and Bühlmann [2012] and Wang et al. [2017]. We summarize constructing an approximate posterior via the DAG bootstrap in Algorithm 8.

Algorithm 8 `DAGBootSample`

```
    Input: N datapoints $D_N$, number of samples T
    Output: $T$ bootstrap DAG samples $\hat{\mathcal{G}}_T$
1: $\hat{\mathcal{G}}_T \leftarrow \emptyset$
2: for $s = 1 : T$ do
3:     $\tilde{D}_N \leftarrow N$ datapoints sampled (with replacement) from $D_N$
4:     $G_s \leftarrow$ DAGLearner($D_N$) e.g. [Wang et al., 2017, Hauser and Bühlmann, 2012]
5:     $\hat{\mathcal{G}}_T \leftarrow \hat{\mathcal{G}}_T \cup G_s$
    return $\hat{\mathcal{G}}_T$
```

Given $\hat{\mathcal{G}}_T$, which can be constructed from Algorithm 8 or sampled from a Markov chain, we next discuss how to compute the expectation over θ. Recall that $U^f_{\text{M.I.}}(\xi; D)$ is given by

$$\begin{aligned} &\mathbb{E}_{G|D}\left[\mathbb{E}_{y|G,\xi} U^f_{\text{M.I.}}(y, \xi; D)\right] \\ &\quad = \mathbb{E}_{G|D}\left[\mathbb{E}_{\theta|G,D}\mathbb{E}_{y|G,\theta,\xi} U^f_{\text{M.I.}}(y, \xi; D)\right]. \end{aligned} \tag{6.7}$$

Instead of carrying out the expensive expectation over $\theta \mid G, D$ in Eq. (6.7), we use the MLE of θ for each sampled G. This approximation is justified by the *Bernstein-von Mises Theorem*, which implies that

$$\begin{aligned} \mathbb{P}(\theta \mid G, D) &\to N(\hat{\theta}^G_{\text{MLE}}, \frac{1}{n} I(\theta_G)^{-1}), \\ \hat{\theta}^G_{\text{MLE}} &\coloneqq \operatorname{argmax} \theta\ \mathbb{P}(D \mid G, \theta). \end{aligned} \tag{6.8}$$

Here, n is the number of datapoints in D, and $I(\theta_G)$ is the Fisher information matrix of the parameter θ_G, which is the asymptotic limit of the maximum likelihood estimator $\hat{\theta}^G_{\text{MLE}}$ [Van der Vaart, 2000, Chapter 10]. Therefore, the posterior distribution $\theta \mid D, G$ concentrates around $\hat{\theta}^G_{\text{MLE}}$ at the standard $O(1/\sqrt{n})$ statistical rate. Hence, for moderate n (e.g., when a moderate amount of observational data is provided at the start of the experimental design), Eq. (6.8) implies

$$\begin{aligned}&\mathbb{E}_{G|D}\mathbb{E}_{\theta|G,D}\left[\mathbb{E}_{y|G,\theta,\xi}U^f_{\text{M.I.}}(y,\xi;D)\right]\\ &\approx \mathbb{E}_{G|D}\left[\mathbb{E}_{y|G,\hat{\theta}^G_{\text{MLE}},\xi}U^f_{\text{M.I.}}(y,\xi;D)\right].\end{aligned} \tag{6.9}$$

In Hauser and Bühlmann [2015, Section 6.1], the authors provide a closed-form expression for $\hat{\theta}^G_{\text{MLE}}$ when $y \mid G, \theta$ is multivariate Gaussian. In this case, $\hat{\theta}^G_{\text{MLE}}$ is a simple function of the sample covariance matrix.

6.4.2 Approximating Mutual Information

While in the previous subsection we showed how to approximate the expectations in Eq. (6.9), computing $U^f_{\text{M.I.}}(y,\xi)$ even for a fixed y is intractable since we must sum over all possible DAGs. Recall from Eq. (6.3) that the mutual information utility function is

$$U^f_{\text{M.I.}}(y,\xi;D) = H(f \mid D) - H(f \mid D, y = y, \xi). \tag{6.10}$$

Note that $H(f \mid D)$ is a constant and does not matter in the optimization over ξ. More care is required for computing the second term in Eq. (6.10), since the posterior of G changes as a result of observing y, the realizations of the interventions specified by ξ. We therefore cannot immediately use the samples in $\hat{\mathcal{G}}_T$ to approximate this term. To overcome this problem, we propose to use *weighted importance sampling* and approximate $H(f \mid D, y, \xi)$ by a weighted average of DAGs in $\hat{\mathcal{G}}_T$. We define the importance sample weights for DAG G_i, $1 \leq i \leq T$, by

$$w_i := \frac{\mathbb{P}(D, y \mid G_i, \xi)}{\mathbb{P}(D \mid G_i)}. \tag{6.11}$$

In general, w_i is not equal to $\mathbb{P}(y \mid G, \xi)$ since D and y are dependent without conditioning on θ. While $\mathbb{P}(D, y \mid G, \xi)$ can be computed in closed-form if the prior on $\theta \mid G$ belongs to one of the families described in Geiger and Heckerman [1999], the dependence on previous samples in the importance weights makes greedily building up the intervention set ξ expensive. In particular, since Eq. (6.11) does not factorize, the importance weights must be recomputed with every new additional intervention, which again requires an integration over all parameters. Motivated by the approximation in Section 6.4, where the parameters of each sampled $G \in \hat{\mathcal{G}}_T$ are not marginalized out, we instead propose using the importance sample weights

$$\begin{aligned}\hat{w}_i &:= \frac{\mathbb{P}(D, y \mid G_i, \hat{\theta}^{G_i}_{\text{MLE}}, \xi)}{\mathbb{P}(D \mid G_i, \hat{\theta}^{G_i}_{\text{MLE}})}\\ &= \mathbb{P}(y \mid G_i, \xi, \hat{\theta}^{G_i}_{\text{MLE}});\end{aligned} \tag{6.12}$$

$\hat{w}_i$ has the natural interpretation of re-weighting each DAG by the likelihood of the newly observed data y.

Recall from Eq. (6.3), that $U^f_{\text{M.I.}}(y, \xi; D)$ is based on weighting each DAG according to its posterior probability $\mathbb{P}(G \mid D) \propto \int_\theta \mathbb{P}(D \mid G, \theta)\mathbb{P}(\theta \mid G)\mathbb{P}(G)$. Using the importance sample weights $\hat{w}_i$ translates into approximating the mutual information against a different posterior distribution in Eq. (6.3), namely

$$\tilde{\mathbb{P}}(G \mid D) \propto \mathbb{P}(D \mid G, \hat{\theta}^G_{\text{MLE}})\mathbb{P}(G), \tag{6.13}$$

which is a specific instance of an empirical Bayes approximation. In what follows, we denote the mutual information score based on the posterior in Eq. (6.13) by $\tilde{U}^f_{\text{M.I.}}(y, \xi; D)$.

6.4.3 Greedy Optimization

The cardinality constraint $|\xi| = N_b$ makes our optimization problem a difficult integer program. In the following, we show how to overcome this final computational hurdle using a generalized notion of *submodularity* for multisets [Soma and Yoshida, 2016]. In particular, we prove that greedily selecting interventions provides a $(1 - \frac{1}{e})$ guarantee on optimization quality.

Algorithm 9 `GreedyDesign`

```
    **Input:** Utility function $U$, number of samples $N_b$, intervention family $\mathcal{I}^*$
    **Output:** Multiset of interventions $\xi$
 1: $\xi \leftarrow \emptyset$
 2: for $s = 1 : N_b$ do
 3:     $I^* \in \text{argmax}\, I \in I^*\ U(\xi \cup I)$
 4:     $\xi \leftarrow \xi \cup I^*$
    return $\xi$
```

Theorem 6.4.1. *Suppose $f(G) = G$ i.e. the goal is to recover the full graph as in Tong and Koller [2001], Cho et al. [2016], Murphy [2001], Ness et al. [2018]. Then the difference between the global optimum*

$$v^*_b = \max_{\xi \in \mathbb{Z}^{\mathcal{I}^*} \cap C_{N,b}} \mathbb{E}_{G|D_{b-1}} \mathbb{E}_{y|G,\hat{\theta}^G_{MLE},\xi}\ \tilde{U}^f_{M.I.}(y, \xi; D)$$

and $\tilde{v}_b =$ `GreedyDesign`$(\tilde{U}^f_{M.I.}, N_b, \mathcal{I}^*)$, the output of Algorithm 9 in batch b, satisfies $\tilde{v}_b \geq (1 - \frac{1}{e})v^*_b$, where $C_{N,b}$ is defined as in Eq. (6.4)*.*

Remark. We conjecture that Theorem 6.4.1 holds for arbitrary functions f, but we currently only have a proof (see Appendix) for the case when $f(G) = G$.

We conclude this section by summarizing the developed *Active Budgeted Causal Design Strategy* (*ABCD-Strategy*) in Algorithm 10 and then summarizing all the proposed approximations.

Algorithm 10 `ABCD`-Strategy

```
**Input:** Target functional $f$, interventional data collected $D_{b-1}$, observational data $D_{obs}$, number of batch samples $N_b$, intervention family $\mathcal{I}^*$, number of DAGs $T$, number of datasets $M$
**Output:** Multiset of interventions $\xi$
1: $\xi \leftarrow \emptyset$
2: $G_T \leftarrow$ DAGBootSample($[D_{obs}, D_{b-1}]$, $T$)
3: Compute $\hat{U}^f_{\text{M.I.}}$ via Eq. (6.14)
4: return GreedyDesign($\hat{U}^f_{\text{M.I.}}$, $N_b$, $\mathcal{I}^*$)
```

In terms of approximations, Eq. (6.9) implies

$$\begin{aligned}
&\mathbb{E}_{G,\theta|D}\left[\mathbb{E}_{y|G,\theta,\xi}\tilde{U}^f_{\text{M.I.}}(y,\xi;D)\right] \\
&\approx \mathbb{E}_{G,|D}\left[\mathbb{E}_{y|G,\hat{\theta}^G_{\text{MLE}},\xi}\ \tilde{U}_{\text{M.I.}}(y,\xi;D)\right] \\
&\approx \sum_{t=1}^{T}\sum_{m=1}^{M}\tilde{U}^f_{\text{M.I.}}(y_{tm},\xi;D), \\
&\qquad \text{s.t. } y_{tm} \overset{\text{i.i.d}}{\sim} y \mid G_t, \hat{\theta}^G_{\text{MLE}}, \xi \\
&\approx \sum_{t=1}^{T}\sum_{m=1}^{M}\hat{U}^f_{\text{M.I.}}(y_{tm},\xi;D), \text{ where}
\end{aligned} \tag{6.14}$$

$$\begin{aligned}
&\hat{U}^f_{\text{M.I.}}(y_{tm},\xi;D) := H_1(f \mid D) - H_2(f \mid D), \\
&\hat{\mathbb{P}}_1(G \mid D) := w_{G,D}\mathbb{1}(G \in \hat{\mathcal{G}}_T), \\
&\hat{\mathbb{P}}_2(G \mid D, y, \xi) := \frac{w_{G,D}\mathbb{P}(y \mid G, \xi, \hat{\theta}^G_{\text{MLE}})}{\sum_{t=1}^{T} w_{G_t,D}\mathbb{P}(y \mid G_t, \xi, \hat{\theta}^{G_t}_{\text{MLE}})},
\end{aligned}$$

where M is the number of synthetic datasets generated, H_1 and H_2 are the entropies induced by $\hat{\mathbb{P}}_1$ and $\hat{\mathbb{P}}_2$ respectively, and $w_{G,D}$ is defined in Eq. (6.6). Note that $\hat{U}^f_{\text{M.I.}}$ is based on the importance sample weights given in Eq. (6.12).

Proposition 6.4.2. *The total runtime of Algorithm 9 with input utility function $\hat{U}^f_{M.I.}$ is $O(pT\kappa^3 + |\mathcal{I}^*|MT^2N_b\kappa p)$, where κ is the maximum indegree of a graph in $\hat{\mathcal{G}}_T$.*

See Appendix E.1.5 for the proof of Proposition 6.4.2.

6.5 Experiments

We begin by considering a simple case to demonstrate the behavior of our ABCD-strategy under easily interpretable conditions. Consider the chain graph on $2m-1$ nodes,

$$1 \to 2 \to \ldots \to m \to \ldots \to p = 2m-1. \tag{6.15}$$

The corresponding essential graph is completely undirected, and the MEC has $2m-1$ members, one with each node as the source. Assume that sufficient observational data is available to identify the MEC, and we are interested in fully identifying the DAG. Then, our ABCD-strategy selects interventions in order to minimize the expected entropy of the posterior over this MEC. Given a limit of one intervention per batch but infinite samples per batch, Proposition 6.3.2 implies the expected entropy after intervening at node i or $2m-i$, $1 \leq i \leq m$, is

$$\frac{1}{2m-1}\Big(\sum_{j<i} \log(i-2) + \sum_{j>i} \log(m-(i+2))\Big),$$

which is minimized by choosing the midpoint $i = m$. Analogously, we see that the updated $\{\emptyset, \{m\}\}$-MEC is of the same form, so in the second batch, the optimal intervention will be halfway through the remaining nodes. This process of bisection is illustrated in Figure 6-1 and matches the behavior of our algorithm even in the finite-sample regime as described next.

Figure 6-2 illustrates the performance of our ABCD-strategy on fifty 11-node chain graphs with random edge weights sampled from $[-1, -.25] \cup [.25, 1]$. For comparison, we consider a random intervention strategy that uniformly distributes the samples in each batch to k interventions picked uniformly at random, where k is the maximum number of unique interventions allowed per batch. Whereas the median-performing random strategy barely reduces the entropy, the ABCD-strategy reduces the entropy significantly in all runs. When all k interventions are picked for the same batch, so that ABCD receives no feedback, the median-performing run of active learning still reduces the entropy as much as the best-performing runs of the random strategy.

Having demonstrated the behavior of ABCD for a simple case, we now analyze the performance of our method on more general DAGs. The skeleton of each graph is sampled from an Erdös-Rényi model with density $\rho = 0.25$. The edges of these graphs are directed by sampling a permutation of the nodes uniformly at random and orienting the edges accordingly. To avoid long runtimes when enumerating the MEC, we disposed of graphs with more than 100 members in their MEC.[1] When the MEC is known, we may define a variant of the random strategy, *Chordal-Random*, which only intervenes on nodes that are in chordal components of the essential graph, i.e., nodes adjacent to at least one undirected edge. Since the Meek rules can only propagate by intervening within chordal components, Chordal-Random is a more fair baseline strategy for comparison than simple random sampling.

Figure 6-3a demonstrates the improvement in selecting interventions using the ABCD-strategy as compared to Chordal-Random when the number of unique interventions per batch is bounded by one. The entropy reduction for an interventional data set D_ξ is defined as $\frac{H(G)-H(G|D_\xi)}{H(G)}$, and it is used as a metric so that MECs of different sizes are comparable. Since the number of total possible unique interventions is kB, an increase in the number of batches also increases the variability of the interventions,

[1]From a sample of 10,000 graphs, only 54 had MEC size greater than 100. Based on the results by Gillispie and Perlman [2001], we expect the MECs to be typically small.

reflected in the increase of entropy reduction with batch size. Already with only 192 samples and 3 total batches, our ABCD-strategy is able to learn most graphs with complete certainty. The comparable performance of the Budgeted Experiment Design (BED) strategy [Ghassami et al., 2018] suggests that for the given experimental setup, the interventions that orient the most edges correspond well to those that most reduce entropy as we discussed in Proposition 6.3.2. Figure 6-3b shows that the performance of the ABCD-strategy remains strong even when the MEC of the graph is not known. Specifically, up to 3 additional MECs were generated by randomly flipping non-covered edges that did not create cycles, and again only graphs for which the union of these MECs had cardinality less than 100 were kept. Note that we are not able to compare with BED since BED requires that the MEC is known.

DREAM4 Synthetic Dataset. Finally, we applied our experimental design strategy to gene expression data from the DREAM4 10-node in-silico network reconstruction challenge [Schaffter et al., 2011]. These data are generated from stochastic differential equations and simulate microarray data of gene regulatory networks. We constructed an observational dataset from the wild-type, multifactorial perturbation, and time 0 time-series samples (16 samples in total), and similarly, interventional datasets from the knockdown and knockout samples (2 samples each).

Previous work on experimental design applied to biological datasets [Cho et al., 2016] has focused on learning the entire network. In practice, practitioners may be specifically interested in performing experiments to elucidate a functional of the network, such as the pathway or local network surrounding a gene of interest. To emulate this setting, we applied our ABCD-strategy towards learning the *downstream genes* of select genes from the true network (Figure 6-4, top). Despite high variations in learning due to the small size of the dataset, we observed an improvement over the random strategy for several central genes (Fig. 6-4, bottom; Fig. E-2). These results illustrate the promise of applying targeted experimental design for applications to genomics.

6.6 Concluding Remarks

We proposed *Active Budgeted Causal Design Strategy* (*ABCD-Strategy*), an experimental method based on optimal Bayesian experimental design with provable guarantees on approximation quality. Empirically, we demonstrated that ABCD yields considerable boosts over random sampling for both targeted and full causal structure discovery. Such experimental design strategies are particularly relevant for applications to genomics, where the number of possible experiments is huge due to the possibility of intervening on combinations of genes.

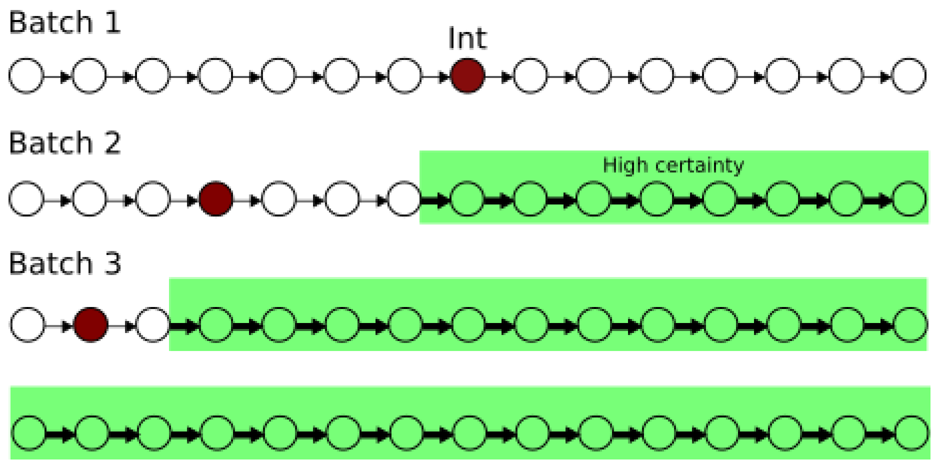


Figure 6-1: Illustration of active learning on a chain graph, beginning with a known MEC on a simulated dataset with $p = 15$ nodes. The brown circles indicate the interventions selected in each batch.

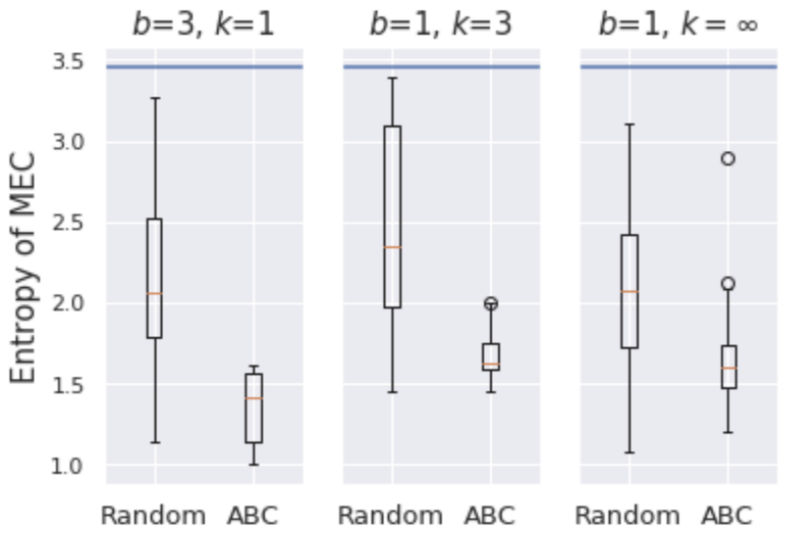
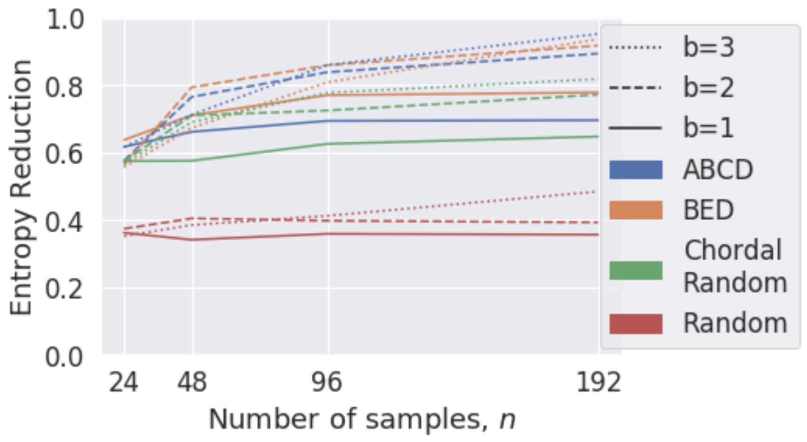
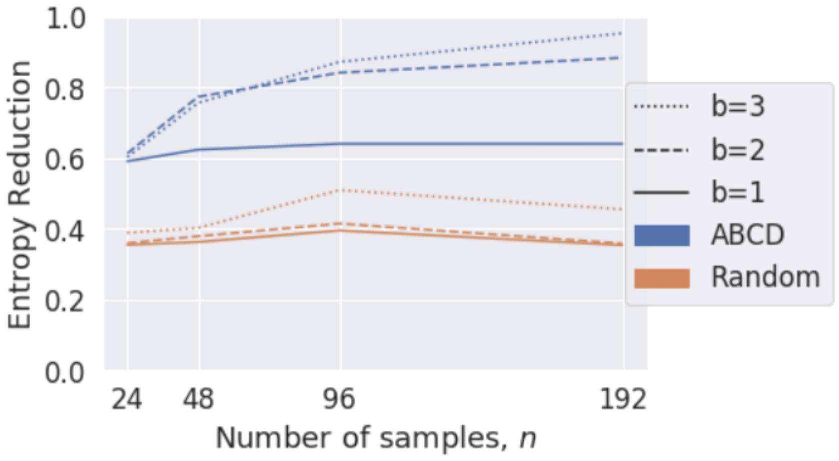


Figure 6-2: Box plots for 50 runs of the random strategy versus our ABCD-strategy on the graph in Figure 6.5 with $p = 11$ and $n = 30$ samples. The horizontal line indicates the entropy of the prior distribution, i.e. uniform over the MEC. Note that $k = \infty$ corresponds to the case with no constraints on the number of unique interventions.

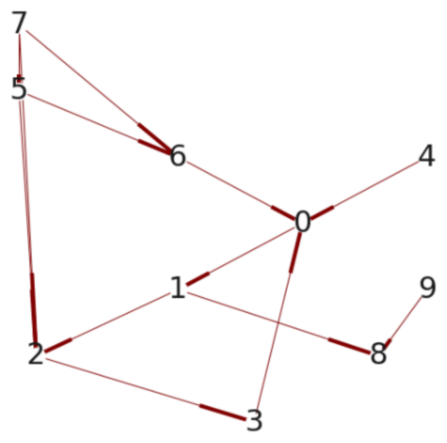


(a) Single MEC



(b) Multiple MECs.

Figure 6-3: Performance of intervention strategies for batch sizes b as a function of the total number of samples, computed from 50 Erdös-Rényi DAGs with density $\rho = 0.25$.



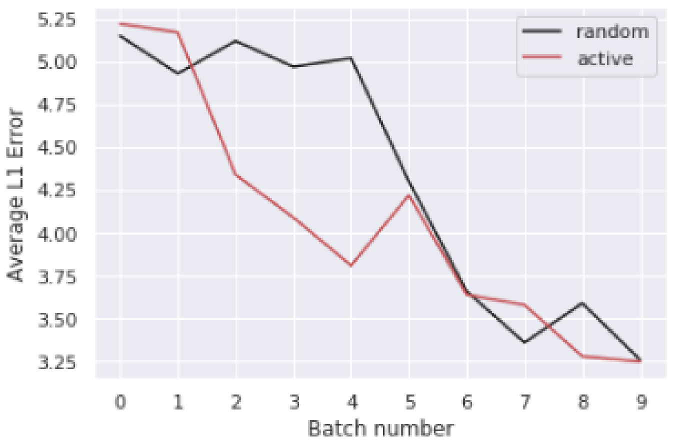


Figure 6-4: Top: DREAM4 ground truth 10-node network. Bottom: Performance of intervention strategies on predicting the descendants of gene 0.

Appendix A

Appendix for "Data-Dependent Compression of Random Features for Large-Scale Kernel Approximation"

A.1 Proof of Theorem 2.3.4

The proofs of Theorem 2.3.2 and Theorem 2.3.4 rely on the main error bound for the *Hilbert coreset construction problem* given in Eq. (A.1) [Campbell and Broderick, 2019]. We restate this error bound in Lemma A.1.2, which depends on several key quantities given below:

- $c_{ls} := \frac{1}{J_+} \cos(\omega_l^T x_{i_s} + b_l) \cos(\omega_l^T x_{j_s} + b_l)$, such that $1 \leq s \leq S$ and $1 \leq l \leq J_+$
- $\hat{\sigma}_j^2 := \frac{1}{S} \sum_{s=1}^{S} c_{js}^2 = \frac{1}{S} \|R_j\|_2^2$
- $\hat{\sigma}^2 := \left(\sum_{j=1}^{J_+} \hat{\sigma}_j \right)^2$

Definition A.1.1. [Campbell and Broderick, 2019] The *Hilbert construction problem* is based on solving the quadratic program,

$$\underset{w \in \mathbb{R}_+^{J_+}}{\operatorname{argmin}} \quad \frac{1}{S} \|r - r(w)\|_2^2 \quad \text{s.t.} \sum_{j=1}^{J_+} w_j \hat{\sigma}_j = \hat{\sigma}. \tag{A.1}$$

Remark. The minimizer of Eq. (A.1) is $w^* = (1, \cdots, 1)$ since $r(w^*) = r$. However, the goal is to find a sparse w. Instead of adding sparsity-inducing constraints (such as L_1 penalties), which would lead to computational difficulties for large-scale problems, Campbell and Broderick [2019] minimize Eq. (A.1) greedily through the Frank-Wolfe algorithm. Frank-Wolfe outputs a sparse w since the sparsity of w is bounded by the number of iterations Frank-Wolfe is run for.

Lemma A.1.2. *[Campbell and Broderick, 2019, Theorem 4.4] Solving Eq.* (A.1) *with J iterations of Frank-Wolfe satisfies*

$$\begin{aligned}\frac{1}{S}\|r - r(w)\|_2^2 &\le \frac{\hat{\sigma}^2\eta^2\bar{\eta}^2\nu_J^2}{\bar{\eta}^2\nu^{-2(J-2)} + \eta^2(J-1)} \\ &\le \nu_J^{2J-2},\end{aligned} \tag{A.2}$$

where $0 \le \nu_J < 1$. *Furthermore,* $\nu_J^2 = 1 - \frac{d^2}{\sigma^2\bar{\eta}^2}$ *where d is the distance from r to the nearest boundary of the convex hull of* $\left\{\frac{\hat{\sigma}}{\hat{\sigma}_j}R_j\right\}_{j=1}^{J_+}$ *and* $\bar{\eta}^2 := \frac{1}{S}\max_{i,j\in[J_+]}\left\|\frac{R_i}{\hat{\sigma}_i} - \frac{R_j}{\hat{\sigma}_j}\right\|^2, 0 \le \bar{\eta} \le 2$.

We prove Theorem 2.3.4 first since the main idea is captured in this proof. The proof of Theorem 2.3.2 is more involved since we must use a number of concentration bounds to justify subsampling only S datapoint pairs instead of all $\frac{N(N-1)}{2}$ possible datapoint pairs. Both proofs will also depend on the following constants.

- $\sigma_j^2 := \frac{1}{V^*}\sum_{s=1}^{V^*} c_{js}^2 = \frac{1}{V^*}\|R_j\|_2^2$
- $\sigma^2 := \left(\sum_{j=1}^{J_+}\sigma_j\right)^2$

Here, $V^* = \frac{N(N-1)}{2}$, that is when all datapoint pairs above the diagonal are included. $\hat{\sigma}_j^2$ and $\hat{\sigma}^2$ are simply unbiased estimates of σ_j^2 and σ^2 based on sampling only S instead of all V^* datapoint pairs.

While Lemma A.1.2 guarantees $0 < \nu_{J_+} < 1$, it does not guarantee that $\nu_{J_+} \to 1$ as the number of random features $J_+ \to \infty$. The following Lemma is critical in showing that ν_{J_+} does not approach 1, which would result in no compression.

Lemma A.1.3. *Let* $\{x_i\}_{i=1}^K$ *be a set of points in* $\mathbb{R}^p$ *that satisfies Assumption 2.3.1(a). Consider the vector* $v_{\omega,b} = (\cos(\omega^T x_i + b)\cos(\omega^T x_j + b))_{i<j,i\in[K-1]} \in \mathbb{R}^{\frac{K(K-1)}{2}}$. *Let the unit vector* $u_{\omega,b} := \frac{v_{\omega,b}}{\|v_{\omega,b}\|}$. *If* $\omega_j \overset{i.i.d.}{\sim} F$ *and* $b_j \overset{i.i.d.}{\sim} G$, *where F has positive density on all of* $\mathbb{R}^p$ *and G has positive density on* $[0, 2\pi]$, *then*

$$\begin{aligned}&\mathrm{d}\left(\mathrm{ConvexHull}\{u_{\omega_j,b_j}\}_{j=1}^J, \mathcal{S}^{\frac{K(K-1)}{2}-1}\right) \to 0 \quad \textit{for} \quad J \to \infty \\ &\textit{s.t.} \quad \mathrm{d}(A,B) := \max_{a\in A, b\in B}||a-b||_2.\end{aligned} \tag{A.3}$$

Here, $\mathcal{S}^{\frac{K(K-1)}{2}-1}$ *denotes the surface of the unit sphere in* $\mathbb{R}^{\frac{K(K-1)}{2}}$.

Proof. By construction, each unit vector $u_i := u_{\omega_i,b_i}$ lies on the boundary of the unit sphere in $\mathbb{R}^{\frac{K(K-1)}{2}}$. Hence, F, G induce a distribution on $\mathcal{S}^{\frac{K(K-1)}{2}-1}$. It suffices to show $\mathcal{S}^{\frac{K(K-1)}{2}-1}$ has strictly positive density everywhere since, as $J \to \infty$, any arbitrarily small neighborhood around a collection of points that cover $\mathcal{S}^{\frac{K(K-1)}{2}-1}$ will be hit by some u_i with probability 1. By standard convexity arguments, the convex hull of the u_i will arbitrarily approach $\mathcal{S}^{\frac{K(K-1)}{2}-1}$ by taking the radius of the neighborhoods to zero. We now show $\mathcal{S}^{\frac{K(K-1)}{2}-1}$ has strictly positive density everywhere. Since u_i is the normalized vector of $v_i \coloneqq v_{\omega_i,b_i}$ and each component of v_i is between -1 and 1, it suffices to show, by the continuity of the cosine function, that for any $a \in \{-1,1\}^{\frac{K(K-1)}{2}}$ there exist some ω_i, b_i such that $\text{sign}(v_i) \coloneqq (\text{sign}(v_{il}))_{l\in\frac{K(K-1)}{2}}$ equals a. Recall that

$$\cos(a)\cos(b) = \frac{1}{2}(\cos(a+b)+\cos(a-b)). \tag{A.4}$$

Take $b_i = 0$. Then, Equation (A.4) implies $v_{il} = \frac{1}{2}(\cos(\omega_i^T(x_{i_l}+x_{j_l})+\cos(\omega^T(x_{i_l}-x_{j_l}))$. Consider the vector $\tilde{v}_i = (\cos(\omega_i^T(x_{i_l}+x_{j_l}), \cos(\omega_i^T(x_{i_l}-x_{j_l}))_{l\in\frac{K(K-1)}{2}} \in \mathbb{R}^{K(K-1)}$. It suffices to show that for any $\tilde{a} \in \{-1,1\}^{K(K-1)}$, there exists an ω_i such that $\text{sign}(\tilde{v}_i) = \tilde{a}$. Recall that the cosine function has infinite *VC dimension*, namely that for any labeling $y_1, \cdots, y_M \in \{-1,1\}$ of distinct points $x_1, \cdots x_M \in \mathbb{R}^p$, there exists an ω^* such that $\text{sign}(\cos((\omega^*)^T x_m)) = y_m$. Take $M = K(K-1)$, $y_m = \tilde{a}_m$, $x_m = x_{i_m} + x_{j_m}$, and $x_{m+1} = x_{i_m} - x_{j_m}$. Since all the x_m are distinct by Assumption 2.3.1(a), we can find an ω_i such that $\text{sign}(\tilde{v}_i) = \tilde{a}$ as desired. □

Proof. We now prove Theorem 2.3.4. Each $R_j \in \mathbb{R}^{\frac{N(N-1)}{2}}$ and the R_j's are i.i.d. since each ω_j is drawn i.i.d. from Q. The induced Hilbert norm $\|\cdot\|_H$ of each R_j is given by $\|R_j\|_H^2 = \frac{2}{N(N-1)}\|R_j\|_2^2$ [Campbell and Broderick, 2019]. Hence, $\tilde{R}_j \coloneqq \frac{R_j}{\sigma_j}$ is a unit vector in the vector space with norm $\|\cdot\|_H$. By Lemma A.1.3,

$$\text{d}\left(\text{ConvexHull}\{\tilde{R}_j\}_{j=1}^{J_+}, \mathcal{S}^{\frac{N(N-1)}{2}-1}\right) \to 0 \tag{A.5}$$

Let $\tilde{r} \coloneqq \frac{1}{\sigma}\sum_{j=1}^{J_+}\sigma_j\tilde{R}_j \in \text{ConvexHull}\{\tilde{R}_j\}_{j=1}^{J_+}$ and observe that $\tilde{r} = \frac{r}{\sigma}$. The distance, which we denote as d_{J_+}, between $\tilde{r}$ and the $\text{ConvexHull}\{\tilde{R}_j\}_{j=1}^{J_+}$ approaches $1 - \|\tilde{r}\|_H$ since the $\text{ConvexHull}\{\tilde{R}_j\}_{j=1}^{J_+}$ approaches $\mathcal{S}^{\frac{N(N-1)}{2}-1}$. Hence,

$$\lim_{J_+\to\infty} d_{J_+} = 1 - \lim_{J_+\to\infty}\|\tilde{r}\|_H = 1 - \frac{\lim_{J_+\to\infty}\|r\|_H}{\lim_{J_+\to\infty}\sigma}. \tag{A.6}$$

Now,

$$r_s = \frac{1}{J_+}\sum_{j=1}^{J_+} c_{js} \overset{J_+\to\infty}{\longrightarrow} k(x_{i_s}, x_{j_s}). \tag{A.7}$$

Hence, as $J_+ \to \infty$,

$$\|r\|_H \to \sqrt{\frac{2}{N(N-1)}\sum_{i<j}(k(x_i,x_j))^2}. \tag{A.8}$$

Now,

$$\begin{aligned}\sigma &= \sum_{j=1}^{J_+} \sigma_j \\ &= \sum_{j=1}^{J_+} \sqrt{\frac{1}{V^*} \sum_{s=1}^{V^*} c_{js}^2} \\ &= \sum_{j=1}^{J_+} \sqrt{\frac{1}{V^*} \sum_{s=1}^{V^*} \frac{1}{J_+^2} \cos^2(\omega_j^T x_{i_s} + b_j) \cos^2(\omega_j^T x_{j_s} + b_j)} \\ &= \frac{1}{J_+} \sum_{j=1}^{J_+} \sqrt{\frac{1}{V^*} \sum_{s=1}^{V^*} \cos^2(\omega_j^T x_{i_s} + b_j) \cos^2(\omega_j^T x_{j_s} + b_j)} \\ &= \sqrt{\frac{2}{N(N-1)}} \frac{1}{J_+} \sum_{j=1}^{J_+} \|(\cos(\omega_j^T x_m + b_j) \cos(\omega_j^T x_n + b_j))_{m<n}\|_2 \\ &\to \sqrt{\frac{2}{N(N-1)}} \mathbb{E}_{\omega,b} \|(\cos(w^T x_m + b) \cos(w^T x_n + b))_{m<n}\|_2\end{aligned} \tag{A.9}$$

If $x \neq y$ and $w \neq 0$, then

$$\begin{aligned}k(x,y) &= \mathbb{E}_{\omega,b} \cos(w^T x + b) \cos(w^T y + b) \\ &< \mathbb{E}_{\omega_i,b} |\cos(w^T x + b) \cos(w^T y + b)|.\end{aligned} \tag{A.10}$$

by Jensen's inequality. Hence, Eq. (A.10) and Assumption 2.3.1(a-b) together imply

$$\frac{\lim_{J_+\to\infty} \|r\|_2}{\lim_{J_+\to\infty} \sigma} < 1. \tag{A.11}$$

By Eq. (A.8) and Eq. (A.9),

$$\frac{\lim_{J_+\to\infty} \|r\|_H}{\lim_{J_+\to\infty} \sigma} \leq \frac{\|K\|_F}{\mathbb{E}_{\omega,b} \|u(\omega, b)\|_2}, \tag{A.12}$$

where $u(\omega, b)$ is defined in Theorem 2.3.4. Lemma A.1.2 says that $\nu_{J_+}^2 = 1 - \frac{d^2}{\sigma^2 \bar{\eta}^2}$, where d is the distance from r to the nearest boundary of the convex hull of $\left\{\frac{\sigma}{\sigma_j} R_j\right\}_{j=1}^{J_+}$. Hence, $d = \sigma d_{J_+}$ and $\nu_{J_+}^2 = 1 - \frac{d_{J_+}^2}{\bar{\eta}^2}$. Eq. (A.6) and Eq. (A.12) together imply,

$$\liminf_{J_+\to\infty} d_{J_+} \leq 1 - \frac{\|K\|_F}{\mathbb{E}_{\omega,b} \|u(\omega, b)\|_2}. \tag{A.13}$$

Therefore, since $0 \leq \bar{\eta}^2 \leq 2$ by Lemma A.1.2,

$$\begin{aligned}\limsup_{J_+\to\infty} \nu_{J_+}^2 &\leq \limsup_{J_+\to\infty} 1 - \frac{d_{J_+}^2}{2} \\ &= 1 - \liminf_{J_+\to\infty} \frac{d_{J_+}^2}{2} \\ &\leq 1 - \frac{\left(1 - \frac{\|K\|_F}{\mathbb{E}_{\omega,b}\|u(\omega,b)\|_2}\right)^2}{2}.\end{aligned} \tag{A.14}$$

□

A.2 Proof of Theorem 2.3.2

The following technical lemma is needed to derive the probability bound in Theorem 2.3.2.

Lemma A.2.1. *Suppose* $\frac{\sigma^2}{J_+^2\sigma_i^2} \leq M$ *for some* $1 \leq M < \infty$ *for all* $i \in [J_+]$. *For* $S \geq 8\frac{M^2}{\sigma^4}\log\left(\frac{2J_+}{\delta^2}\right)$

$$\mathbb{P}\left(\frac{\hat{\sigma}^2}{J_+^2\hat{\sigma_i}^2} \geq 5M\right) \leq \delta \tag{A.15}$$

for all $i \in [J_+]$.

Proof. Notice that

$$\begin{aligned}\mathbb{E}_{i_s,j_s}\hat{\sigma_l}^2 &= \frac{1}{S}\sum_{s=1}^{S}\mathbb{E}_{i_s,j_s}c_{ls}^2 \\ &= \frac{1}{N^2}\sum_{s=1}^{N^2} c_{ls}^2 \\ &= \sigma_l^2.\end{aligned}$$

Hence, $\hat{\sigma_l}^2$ is an unbiased estimator of σ_l^2. Each $c_{ls}^2 \leq \frac{1}{{J_+}^2}$ is a bounded random variable, and the collection of random variables $\{c_{ls}^2\}_{s=1}^S$ are i.i.d. since $i_s, j_s \overset{\text{i.i.d.}}{\sim} \pi$. Hence, by Hoeffding's inequality,

$$\mathbb{P}\left(|\hat{\sigma}_l^2 - \sigma_l^2| \geq t\right) \leq 2\exp\left(-2SJ_+^4t^2\right). \tag{A.16}$$

Define the event $A_t := \cup_{i=1}^{J_+}\{|\hat{\sigma}_i^2 - \sigma_i^2| < t\}$ and pick t such that $t \leq \min_{i\in[J_+]}\sigma_i^2$. Since $\sigma_i^2 \geq \frac{\sigma^2}{M}$ by assumption, it suffices to pick $0 < t \leq \frac{\sigma^2}{M}$. Conditioned on A_t,

$\hat{\sigma}_i \leq \sqrt{\sigma_i^2 + t} \leq \sigma_i + \sqrt{t}$, which implies $\hat{\sigma}^2 \leq (\sigma + J_+\sqrt{t})^2$. Therefore,

$$\begin{aligned}
\mathbb{P}\left(\frac{\hat{\sigma}^2}{J_+^2 \hat{\sigma_i}^2} \geq cM\right) &= \mathbb{P}\left(A_t^c \cup \left\{\frac{\hat{\sigma}^2}{J_+^2 \hat{\sigma_i}^2} \geq cM\right\}\right) + \mathbb{P}\left(A_t \cup \left\{\frac{\hat{\sigma}^2}{J_+^2 \hat{\sigma_i}^2} \geq cM\right\}\right) \\
&\leq \mathbb{P}\left(A_t^c\right) + \mathbb{P}\left(A_t, \left\{\frac{\hat{\sigma}^2}{J_+^2 \hat{\sigma_i}^2} \geq cM\right\}\right) \\
&\leq \mathbb{P}\left(A_t^c\right) + \mathbb{P}\left(\frac{\hat{\sigma}^2}{J_+^2 \hat{\sigma_i}^2} \geq cM \mid A_t\right) \\
&\leq \mathbb{P}\left(A_t^c\right) + \mathbb{P}\left(\frac{(\sigma + \sqrt{t}J_+)^2}{J_+^2(\sigma_i^2 - t)} \geq cM \mid A_t\right).
\end{aligned} \tag{A.17}$$

Notice that $\mathbb{P}\left(\frac{(\sigma+\sqrt{t}J_+)^2}{\sigma_i^2 - t} \geq cM^2 \mid A_t\right)$ is either 0 or 1 since σ_i and σ are constants. We pick t so that this probability is 0. To pick t, notice that,

$$\begin{aligned}
\frac{(\sigma + \sqrt{t}J_+)^2}{J_+^2(\sigma_i^2 - t)} &= \frac{\left(\frac{\sigma}{\sigma_i} + \frac{\sqrt{t}J_+}{\sigma_i}\right)^2}{J_+^2(1 - \frac{t}{\sigma_i^2})} \\
&\leq \frac{\left(J_+\sqrt{M} + \frac{J_+\sqrt{tM}J_+}{\sigma}\right)^2}{J_+^2(1 - \frac{t}{\sigma_i^2})} \\
&\leq \frac{M\left(1 + \frac{\sqrt{t}J_+}{\sigma}\right)^2}{1 - \frac{MJ_+^2 t}{\sigma^2}},
\end{aligned} \tag{A.18}$$

where the last inequality holds as long as $0 < t < \frac{\sigma^2}{MJ_+^2}$ and follows by noting that $\frac{1}{\sigma_i^2} \leq \frac{MJ_+^2}{\sigma^2}$ by assumption. Pick $t = \frac{\sigma^2}{4J_+^2 M}$. Since $0 \leq \sigma \leq 1$, this choice of t implies $\frac{M\left(1+\frac{\sqrt{t}J_+}{\sigma}\right)^2}{1-\frac{MJ_+^2 t}{\sigma^2}} \leq 5M$. Hence, for $c = 5$ and this choice of t, $\mathbb{P}\left(\frac{(\sigma+\sqrt{t}J_+)^2}{J_+^2(\sigma_i^2 - t)} \geq 5M \mid A_t\right) = 0$. Combining Eq. (A.17) and Eq. (A.16), we have by a union bound that,

$$\mathbb{P}\left(\frac{\hat{\sigma}^2}{J_+^2 \hat{\sigma_i}^2} \geq 5M\right) \leq 2J_+ \exp\left(-\frac{1}{8} S \frac{\sigma^4}{M^2}\right), \tag{A.19}$$

for all $i \in [J_+]$. Solving for S by setting the right hand side above to δ yields the claim. □

We have all the pieces to prove Theorem 2.3.2. We follow the proof strategy in [Campbell and Broderick, 2019, Theorem 5.2].

Proof. Let $R^* = \left[z_{+1}^{\ T} \circ z_{+1}^{\ T}, \cdots z_{+N-1}^{\ T} \circ z_{+N}^{\ T}, z_{+N}^{\ T} \circ z_{+N}^{\ T}\right] \in \mathbb{R}^{J_+ \times N^2}$. Notice,

$$\frac{1}{N^2}\|Z_+{Z_+}^T - Z(w)Z(w)^T\|_F^2 = (1-w)^T \frac{R^*}{N}\frac{R^{*T}}{N}(1-w). \tag{A.20}$$

We approximate Eq. (A.20) with $(1-w)^T \frac{R}{\sqrt{S}}\frac{R^T}{\sqrt{S}}(1-w)$ and bound the error. Suppose

$$D^* := \max_{i,j\in[J_+]} \left| \left(\frac{R^*}{N}\frac{R^{*T}}{N}\right)_{ij} - \left(\frac{R}{\sqrt{S}}\frac{R^T}{\sqrt{S}}\right)_{ij} \right| \leq \frac{\epsilon}{2}.$$

Then,

$$\begin{aligned}(1-w)^T \frac{R^*}{N}\frac{R^{*T}}{N}(1-w) - (1-w)^T \frac{R}{\sqrt{S}}\frac{R^T}{\sqrt{S}}(1-w) &\leq \sum_{i,j\in[J_+]} |w_i - 1||w_j - 1| D^* \\ &\leq \|w-1\|_1^2 \frac{\epsilon}{2}.\end{aligned} \tag{A.21}$$

Notice,

$$\begin{aligned}\mathbb{E}_{i_s,j_s}\left[\left(\frac{R}{\sqrt{S}}\frac{R^T}{\sqrt{S}}\right)_{ij}\right] &= \mathbb{E}_{i_s,j_s}\left[\frac{1}{S}\sum_{s=1}^{S} c_{is}c_{js}\right] \\ &= \frac{1}{S}\sum_{s=1}^{S} E_{i_s,j_s}\left[c_{is}c_{js}\right] \\ &= E_{i_s,j_s}\left[c_{is}c_{js}\right] \\ &= \frac{1}{N^2}\sum_{s=1}^{N^2} c_{is}c_{js} \\ &= \left(\frac{R^*}{N}\frac{R^{*T}}{N}\right)_{ij}.\end{aligned} \tag{A.22}$$

Hence, the i.i.d. collection of random variables $\{c_{is}c_{js}\}_{s=1}^S$ yields an unbiased estimate of $\left(\frac{R^*}{N}\frac{R^{*T}}{N}\right)_{ij}$. Each $c_{is}c_{js}$ is bounded by $\frac{1}{J_+^2}$. Therefore, by Hoeffding's inequality and a simple union bound,

$$\mathbb{P}\left(D^* \geq \frac{\epsilon}{2}\right) \leq 2J_+^2 \exp\left(-2SJ_+^4\epsilon^2\right). \tag{A.23}$$

Setting the right-hand side to $\frac{\delta^*}{2}$ and solving for $\frac{\epsilon}{2}$ implies with probability at least

$1 - \frac{\delta^*}{2}$,

$$\frac{\epsilon}{2} \leq \frac{1}{\sqrt{S}{J_+}^2} \log \left[\frac{4J_+^2}{\delta^*} \right]^{\frac{1}{2}}. \tag{A.24}$$

Hence, with probability at least $1 - \frac{\delta^*}{2}$,

$$\begin{aligned}\frac{1}{N^2}\|Z_+{Z_+}^T - Z(w)Z(w)^T\|_F^2 &\leq (1-w)^T \frac{R}{\sqrt{S}} \frac{R^T}{\sqrt{S}}(1-w) + \|1-w\|_1^2 \frac{1}{\sqrt{S}J_+^2} \log \left[\frac{4J_+^2}{\delta^*} \right]^{\frac{1}{2}} \\ &= \frac{1}{S}\|r - r(w)\|_2^2 + \|1-w\|_1^2 \frac{1}{\sqrt{S}J_+^2} \log \left[\frac{4J_+^2}{\delta^*} \right]^{\frac{1}{2}}\end{aligned}$$

Lemma A.1.2 implies that there exists a $0 \leq \nu < 1$ such that $\frac{1}{S}\|r - r(w)\|_2^2 \leq \nu^{2J-2}$. Since ν depends on the pairs i_l, j_l picked, we can take ν^* to be the largest ν possible. Since the set of all possible S pairs is finite, that implies $0 \leq \nu^* < 1$. Hence, setting $J = \frac{1}{2}\log_{\nu^*}\left(\frac{\epsilon}{2}\right) + 2$ guarantees that $\frac{1}{S}\|r - r(w)\|_2^2 \leq \frac{\epsilon}{2}$ for any collection of drawn $i_l, j_l, 1 \leq l \leq S$. Assume for any $a \in (0, 1]$ and $\delta > 0$, we can find an M such that

$$\mathbb{P}\left(\max_j \sigma^2/(J_+^2\sigma_j^2) > M\right) < a\delta. \tag{A.25}$$

If Eq. (A.25) holds, we may assume $\max_j \sigma^2/(J_+^2\sigma_j^2) < M$ by setting M large enough since we just need a $1 - \delta$ probabilistic guarantee. By the polytope constraint in Eq. (A.1), $w_i^* \leq \frac{\hat{\sigma}}{\hat{\sigma}_i}$ for all $i \in [J_+]$. Without loss of generality, assume the first J components of w^* can be the only non-zero values since w^* is at least J sparse. For $S \geq 8\frac{M^4}{\sigma^4}\log\left(\frac{2J_+}{\delta^2}\right)$, Lemma A.2.1 implies with probability at least $1 - \frac{\delta^*}{2}$,

$$\begin{aligned}\|1 - w^*\|_1^2 &\leq \left(\frac{\hat{\sigma}}{\hat{\sigma}_i}J + (J_+ - J)\right)^2 \\ &\leq \big(JMJ_+ + J_+)\big)^2 \\ &\leq (2JM\sqrt{5}J_+)^2 \\ &\leq 10J_+^2M^2J^2 \\ &\leq 10J_+^2M^2\frac{(\log\frac{2}{\epsilon})^2}{(\log\nu)^2}\end{aligned} \tag{A.26}$$

Therefore, with probability at least $1 - \delta^*$,

$$\frac{1}{N^2}\|Z_+{Z_+}^T - Z(w)Z(w)^T\|_F^2 \leq \frac{\epsilon}{2} + \frac{10M^2(\log\frac{2}{\epsilon})^2}{\sqrt{S}(\log\nu)^2}\log\left[\frac{4J_+^2}{\delta^*}\right]^{\frac{1}{2}}. \tag{A.27}$$

Finally, setting $S \geq \max\left(\frac{100}{\epsilon^2}\left[M\frac{(\log\frac{2}{\epsilon})}{(\log\nu)}\right]^4\log\left[\frac{4{J_+}^2}{\delta^*}\right], 8\frac{M^4}{\sigma^4}\log\left(\frac{2J_+}{\delta^2}\right)\right)$ implies $\frac{1}{N^2}\|Z_+{Z_+}^T -$

$Z(w)Z(w)^T\|_F^2 \leq \epsilon$ with probability at least $1-\delta^*$ which matches the rate provided in Theorem 2.3.2. It remains to show Eq. (A.25). Notice that

$$\frac{\sigma}{J_+\sigma_j} = \frac{1}{J_+} + \frac{1}{J_+}\sum_{i\neq j}\tilde{\sigma_{ij}}, \tag{A.28}$$

where $\sigma_{ij} := \frac{\sigma_i}{\sigma_j}$. Notice that each σ_{ij} are i.i.d. for $i \neq j$. Let the $\mu_j = \mathbb{E}\sigma_{ij}$ and s_j be the standard deviation of σ_{ij}. Since each σ_j is i.i.d. that implies μ_j and s_j are both constant across j so we drop the subscript. By a union bound, it suffices to show for any $\tau > 0$ we can find an M such that

$$\mathbb{P}\left(\max_{1\leq j\leq J_+} \frac{1}{J_+}\sum_{i\neq j}\tilde{\sigma_{ij}} > M\right) < \tau. \tag{A.29}$$

By Chebyshev's inequality,

$$\mathbb{P}\left(\frac{1}{J_+}\sum_{i\neq j}\tilde{\sigma_{ij}} - \mu > \frac{cs}{J_+}\right) \leq \frac{1}{c^2}. \tag{A.30}$$

Take $c = J_+\tau$. Then,

$$\mathbb{P}\left(\frac{1}{J_+}\sum_{i\neq j}\tilde{\sigma_{ij}} - \mu > \frac{cs}{J_+}\right) \leq \frac{1}{J_+^2\tau} < \tau. \tag{A.31}$$

By a union bound, Eq. (A.30) implies

$$\mathbb{P}\left(\max_{1\leq j\leq J_+} \frac{1}{J_+}\sum_{i\neq j}\tilde{\sigma_{ij}} > M\right) < \frac{1}{\tau J_+} < \tau$$

for $M = \mu + s\tau$ as desired.

The proof showing that $\limsup_{J_+\to\infty} \nu_{J_+} < 1$ is the same as the proof Theorem 2.3.4.

□

A.3 Runtime analysis of methods

The ridge regression and PCA runtimes depend on the number of features used, as specified in Table 2.1, and therefore follow from the first column of the table.

First, we show that using RFM with $J_+ = O\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$ number of random features ensures that $\frac{1}{N^2}\|K - \hat{K}\|_F^2 = O(\epsilon)$ with high probability. By a union bound, $\mathbb{P}\left(\frac{1}{N^2}\|K-\hat{K}\|_F^2 \leq \epsilon\right) \geq \mathbb{P}\left(\max_{i,j\in[N]}|K_{ij} - \hat{K}_{ij}| \leq \sqrt{\epsilon}\right)$. Now, Claim 1 of Rahimi and Recht [2007] implies

$$\mathbb{P}\left(\max_{i,j\in[N]} |K_{ij} - \hat{K}_{ij}| \geq \sqrt{\epsilon}\right) = O\left(\frac{1}{\epsilon}e^{-J_+\epsilon}\right). \tag{A.32}$$

Setting the right-hand side of Eq. (A.32) to some fixed probability threshold δ^* implies $J_+ = O\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon\delta^*}\right)\right)$. Since δ^* is some fixed constant, $J_+ = O\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$ number of random features suffices for an $O(\epsilon)$ error guarantee. Hence, it suffices to use $J_+ = O\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$ as the up-projection dimension for both RFM-FW and RFM-JL.

To prove the bounds for RFM-FW, take $S = \Omega(J_+^2(\log J_+)^2)$. It is straightforward to check that this choice of S satisfies the requirements of Theorem 2.3.2. By Theorem 2.3.2, it suffices to set $J = O(\log J_+)$ for an $O(\epsilon)$ error guarantee. Hence, Algorithm 1 takes $O(SJ_+\log J_+)$ time to compute the random feature weights w since Frank-Wolfe has to be run for a total of $O(\log J_+)$ iterations. Finally, it takes $O(N\log J_+)$ to apply these $O(\log J_+)$ weighted random features to the N datapoints. We conclude by proving the time complexity of RFM-JL.

Denote $\tilde{x_i} := (Z_+)_i \in \mathbb{R}^{J_+}$ as the mapped datapoints from RFM. Let $A \in \mathbb{R}^{J\times J_+}$ for $J \leq J_+$ be a matrix filled with i.i.d. $N(0,\frac{1}{J})$ random variables for the JL compression step. Let $f(x) := Ax$. It suffices to pick a J such that,

$$\mathbb{P}\left(\max_{i,j\in[N]} \left|\tilde{x_i}^T\tilde{x_j} - f(\tilde{x_i})^T f(\tilde{x_j})\right| \geq \sqrt{\epsilon}\right) \leq \delta^* \tag{A.33}$$

for RFM-JL. We use the following corollary from Kakade and Shakhnarovich [2009, Corollary 2.1] to bound the above probability.

Lemma A.3.1. *Let $u, v \in \mathbb{R}^d$ and such that $\|u\| \leq 1$ and $\|v\| \leq 1$. Let $f(x) = Ax$, where A is a $k \times d, k \leq d$ matrix of i.i.d. $N(0,\frac{1}{k})$ random variables. Then,*

$$\mathbb{P}\left(\mid u^Tv - f(u)^T f(v) \mid\right) \leq 4e^{-\frac{1}{4}(\epsilon^2-\epsilon^3)k}. \tag{A.34}$$

$\|\tilde{x_i}\|_2 = 1$ since $\tilde{x_i} = \frac{1}{\sqrt{J_+}}\left(\cos(\omega_1^Tx_i + b), \cdots, \cos(\omega_{J_+}^Tx_i + b)\right)$. Hence, we may apply Lemma A.3.1 to $\tilde{x_i}$. By a union bound and an application of Lemma A.3.1, Eq. (A.33) is bounded by $O\left(N^2e^{-J\epsilon}\right)$. Setting $N^2e^{-J\epsilon}$ equal to δ^* and solving for J implies that $J = \Omega\left(\frac{1}{\epsilon}\log\left(\frac{N^2}{\delta^*}\right)\right)$. Hence, $J = O\left(\frac{1}{\epsilon}\log N\right)$. Now, $O\left(\frac{1}{\epsilon}\right) = O\left(\frac{J_+}{\log\frac{1}{\epsilon}}\right)$ which implies $J = O\left(\frac{J_+\log N}{\log\frac{1}{\epsilon}}\right)$. Since $N > J_+ > O\left(\frac{1}{\epsilon}\right)$, $J = \Omega(J_+)$ suffices for an for an $O(\epsilon)$ error guarantee. While the JL algorithm typically takes $O(NJ_+k)$ time to map a $N \times J_+$ matrix to a $N \times k$ matrix, the techniques in Hamid et al. [2014, Section 3.5] show that only $O(NJ_+\log J)$ time is required by using the Fast-JL algorithm.

A.4 Impact of kernel approximation

Here we provide the precise error bound and runtimes for kernel ridge regression, kernel SVM, and kernel PCA when using a low-rank factorization ZZ^T of K. We denote $X \subset \mathbb{R}^p$ as the input space and define $c > 0$ such that $K(x, x) \leq c$ and $\hat{K}(x, x) \leq c$ for all $x \in X$. This condition is verified with $c = 1$ for Gaussian kernels for example. All the bounds provided follow from Cortes et al. [2010], Talwalkar [2010], where we simply replace the spectral norm with the Frobenius norm since the Frobenius norm upper bounds the spectral norm.

A.4.1 Kernel ridge regression

Exact kernel ridge regression takes $O(N^3)$ since K must be inverted. Suppose $K \approx ZZ^T := \hat{K}$, where Z could be found using RFM for example. Running ridge regression with the feature matrix Z just requires computing and inverting the covariance matrix $Z^T Z \in R^{J\times J}$ which takes $\Theta(\max(J^3, NJ^2))$ time. Proposition A.4.1 quantifies the error between the regressor obtained from K and the one from $\hat{K}$.

Proposition A.4.1. *(Proposition 1 of Cortes et al. [2010]) Let $\hat{f}$ denote the regression function returned by kernel ridge regression when using the approximate kernel matrix $\hat{K} \in \mathbb{R}^{N\times M}$, and f^* the function returned when using the exact kernel matrix K. Assume that every response y is bounded in absolute value by M for some $0 < M < \infty$. Let $\lambda := N\lambda_0 > 0$ be the ridge parameter. Then, the following inequality holds for all $x \in X$:*

$$\begin{aligned}
|\hat{f}(x) - f^*(x)| &\leq \frac{cM}{\lambda_0^2 N}\|\hat{K} - K\|_2 \\
&\leq \frac{cM}{\lambda_0^2 N}\|\hat{K} - K\|_F \\
&= O\left(\frac{1}{N}\|\hat{K} - K\|_F\right)
\end{aligned}$$

A.4.2 Kernel SVM

Kernel SVM regression takes $O(N^3)$ using K since K must be inverted. Again suppose $K \approx ZZ^T := \hat{K}$. Then, training a linear SVM via dual-coordinate decent on Z has time complexity $O\left(NJ \log \rho\right)$, where ρ is the optimization tolerance Hsieh et al. [2008].

Proposition A.4.2. *(Proposition 2 of Cortes et al. [2010]) Let $\hat{f}$ denote the hypothesis returned by SVM when using the approximate kernel matrix $\hat{K}$, f^* the hypothesis returned when using the exact kernel matrix K, and C_0 be the penalty for SVM. Then,* *the following inequality holds for all $x \in X$:*

$$
\begin{aligned}
|\hat{f}(x) - f^*(x)| &\leq \sqrt{2}c^{\frac{3}{4}}C_0\|\hat{K} - K\|_2^{\frac{1}{4}}\left[1 + \frac{\|\hat{K} - K\|_2^{\frac{1}{4}}}{4c}\right] \\
&\leq \sqrt{2}c^{\frac{3}{4}}C_0\|\hat{K} - K\|_F^{\frac{1}{4}}\left[1 + \frac{\|\hat{K} - K\|_F^{\frac{1}{4}}}{4c}\right]. \\
&= O\left(\|\hat{K} - K\|_F^{\frac{1}{2}}\right).
\end{aligned}
$$

A.4.3 Kernel PCA

We follow Talwalkar [2010] to understand the effect matrix approximation has on kernel PCA. For a more in-depth analysis, see pg. 92-98 of Talwalkar [2010]. Without loss of generality, we assume the data are mean zero.

Let $\Phi(\cdot)$ be the unique feature map such that $k(x, y) = \langle\Phi(x), \Phi(y)\rangle$. Let the feature covariance matrix be denoted as $\Sigma_\Phi := \Phi(X_N)\Phi(X_N)^T$, where $\Phi(X_N) := \left[\Phi(x_1)\cdots\Phi(x_n)\right]$. Since the rank of Σ_Φ is at most N, let v_i $1 \leq i \leq N$ be the N singular vectors of Σ_Φ. For certain kernels, e.g., the RBF kernel, the v_i are infinite dimensional. However, the projection of $\Phi(x)$ onto each v_i is tractable to compute via the kernel trick:

$$\Phi(x)^T v_i = \Phi(x)\frac{\Phi(X_N)u_i}{\sqrt{\sigma_i}} = \frac{k_x^T u_i}{\sqrt{\sigma_i}}, \tag{A.35}$$

where $k_x := (K(x_1, x), \cdots, K(x_N, x))$ and u_i is the ith singular vector of K with associated eigenvalue σ_i. Often, the goal is to project $\Phi(x)$ onto the first l eigenvectors of Σ_Φ for dimensionality reduction. To analyze the error of the projection, let P_{V_l} be defined as the subspace V_l spanned by the top l eigenvectors of Σ_Φ. Then, the *average empirical residual* $R_l(K)$ of a kernel matrix K is defined as,

$$
\begin{aligned}
R_l(K) &:= \frac{1}{N}\sum_{n=1}^{N}\|\Phi(x_n)\|^2 - \frac{1}{N}\sum_{n=1}^{N}\|P_{V_l}(\Phi(x_n))\|^2 \\
&= \sum_{i>l}\sigma_i
\end{aligned}
\tag{A.36}
$$

$R_l(K)$ is simply the spectral error of a low-rank decomposition of Σ_Φ using the SVD. If we instead use $\hat{K}$ for the eigendecomposition, the following proposition bounds the difference between $R_l(K)$ and $R_l(\hat{K})$.

Proposition A.4.3. *(Proposition 5.4 of Talwalkar [2010]) For $R_l(K)$ and $R_l(\hat{K})$*

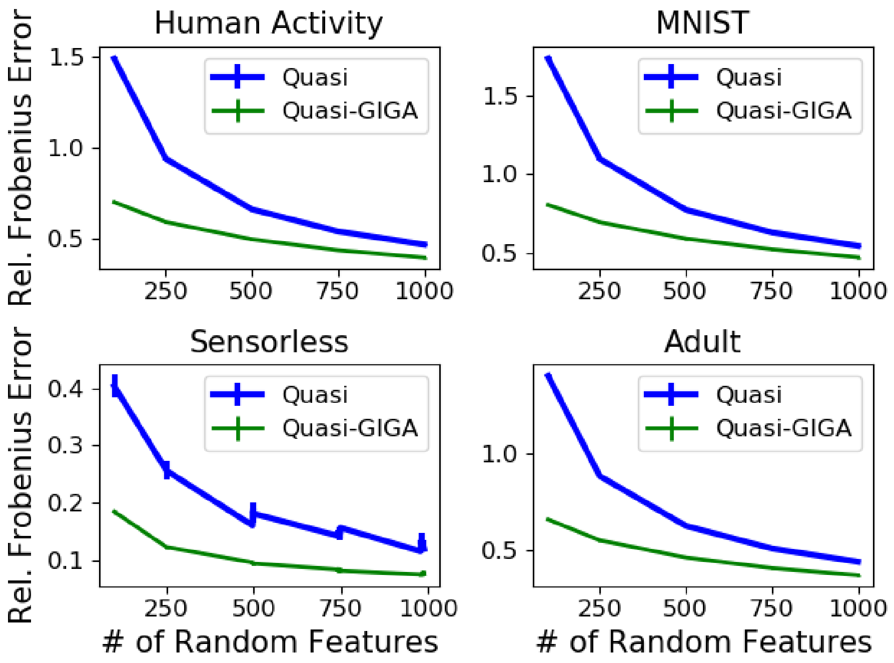


Figure A-1: Kernel matrix approximation errors. Lower is better. Each point denotes the average over 20 simulations and the error bars represent one standard deviation. The HALTON sequence was used to generate the quasi random features.

defined as above,

$$\begin{aligned} |R_l(K) - R_l(\hat{K})| &\leq \left(1 - \frac{l}{N}\right) \|K - \hat{K}\|_2 \\ &\leq \left(1 - \frac{l}{N}\right) \|K - \hat{K}\|_F. \end{aligned}$$

A.5 Additional Experiments

As stated in Section 6.5, our method may be applied on top of other random feature methods. In particular, many previous works have reduced the number of random features needed for a given level of approximation by sampling them from a different distribution (e.g., through importance sampling or Quasi-Monte-Carlo techniques). Regardless of the way the random features are sampled, our method can still be used for compression.

To demonstrate this point further, we consider generating random features using Quasi-Monte-Carlo [Avron et al., 2016]. Quasi random features work by generating a sequence of points from a (low-discrepancy) grid of points in $[0, 1]^p$. Points are sampled from the target random-features distribution Q by applying the inverse CDF of Q on each of these points in the sequence. In Avron et al. [2016], the authors showed that generating random features in this way improved performance over the classical random features method provided in Rahimi and Recht [2007]. In Fig. A-1

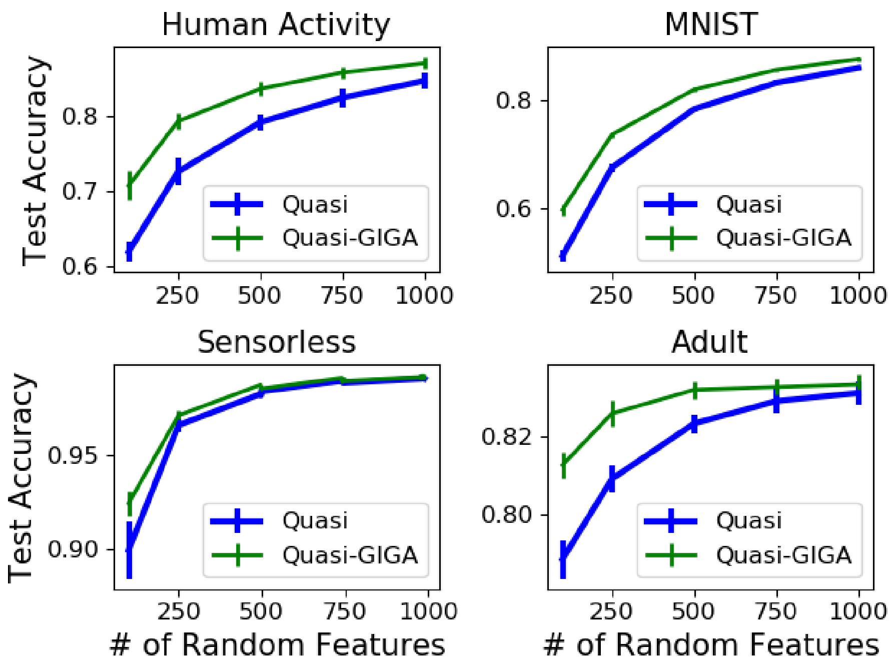


Figure A-2: Classification accuracy. Higher is better. Each point denotes the average over 20 simulations and the error bars represent one standard deviation. The HALTON sequence was used to generate the Quasi random features.

and Fig. A-2, we see that our method is able to compress the number of quasi random features, which is similar to the behavior in Fig. 2-1 and Fig. 2-2. Note that the experimental setup is exactly the same as in Section 6.5 except that the random features are now generated using Quasi-Monte-Carlo.

Appendix B

Appendix for "The Kernel Interaction Trick: Fast Bayesian Discovery of Pairwise Interactions in High Dimensions"

B.1 Modeling Multi-Way Interactions

In certain applications, we might expect that there are interactions of order greater than two. For example, suppose we are trying to predict college admissions. Then, we might expect a three-way interaction between a candidate's SAT score, GPA, and extracurricular involvement. Individually, these variables might only exhibit moderate association but together they could have a multiplicative effect. For example, we might expect that candidates who have high SAT scores, high GPAs, and excellent extracurricular activities will be accepted with near certainty, while candidates who only possess one/two of these qualities are borderline applicants.

We now show how to extend our results to handle such three-way, or more generally, *r-way interactions*.

Definition B.1.1. (r-way interactions) The *r-way interactions* of a covariate vector $x \in \mathbb{R}^p$ are generated from the feature map

$$\Phi_r(x) \coloneqq \bigoplus_{d=1}^{r} \bigoplus_{k:k_1+\cdots+k_p=d} \prod_{j=1}^{p} x_j^{k_j}, \quad k \in \mathbb{N}^p,$$

where $\bigoplus_{j=1}^m a_j \coloneqq (a_{11}, \cdots, a_{1k_1}, \cdots, a_{m1}, \cdots, a_{mk_m})$ denotes the concatenation of vectors $a_j \in \mathbb{R}^{k_j}$.

To model r-way interactions, we must use degree r polynomial kernels to generate all the necessary interactions. Hence, we recommend using the following generalized two-way interaction kernel, which we call the r-way interaction kernel.

Definition B.1.2. (r-way interaction kernel) A kernel k is called an *r-way interaction kernel* if for some choice of $M_1, M_2, M_3 \in \mathbb{N}$, $\alpha, \psi, \lambda^{(m)} \in \mathbb{R}^p_+$ ($m = 1, \ldots, M_1$), $\nu^{(m)} \in \mathbb{R}_+$ ($m = 1, \ldots, M_2$), and $\nabla^{(m)} \in \mathbb{R}^p_+$ ($k = 1, \ldots, M_3$) it can be re-expressed as

$$\sum_{m=1}^{M_1} k^1_{poly,r}(\lambda^{(m)} \odot x, \lambda^{(m)} \odot y) + \sum_{m=1}^{M_2} \nu^{(m)} \left[\prod_{s=1}^{r} x_{i_{sm}} \prod_{s=1}^{r} y_{i_{sm}}\right] + \sum_{m=1}^{M_3} k_{r-1}(\nabla^{(m)} \odot x, \nabla^{(m)} \odot y),$$

where $\odot$ is the Hadamard product and k_{r-1} is an $r-1$ degree interaction kernel. The base case kernel (i.e., when $r = 2$) is provided in Definition 3.4.2.

To select the weights for an r-way interaction kernel, we must solve a system of equations similar to Eq. (3.9), except for a target prior covariance matrix $\Sigma_\tau \in \mathbb{R}^{\dim(\Phi_r)\times\dim(\Phi_r)}$.

B.2 Proofs

B.2.1 Proof of Proposition 3.4.1

Let $g(\cdot) = \theta^T \Phi_2(\cdot)$ and $\theta \mid \tau \sim \mathcal{N}(0, \Sigma_\tau)$. Then, $y^{(n)} = g(x^{(n)}) + \epsilon^{(n)}$. The first claim follows by taking $\phi = \Phi_2$ and $f = g$ in Rasmussen and Williams [2006, Equation 2.12].

The second claim follows directly from the duality between the weight-space and function-space view of a GP [Rasmussen and Williams, 2006, Chapter 2].

B.2.2 Proof of Theorem 3.4.3

The proof of Theorem 3.4.3 depends critically on Lemma B.2.1 below, which characterizes the relation between adding two kernels and the resulting induced prior covariance matrix.

Lemma B.2.1. *Let k_1 and k_2 be two kernels such that there exists vectors $a^{(1)}, a^{(2)} \in \mathbb{R}^{dim(\Phi_2)}$ for which $k_i(x, y) = \langle a^{(i)} \odot \Phi_2(x), a^{(i)} \odot \Phi_2(y)\rangle$. Let $k_3(x, y) = k_1(x, y) + k_2(x, y)$. Then,*

$$k_3(x, y) = \langle \Sigma_3^{\frac{1}{2}} \Phi_2(x), \Sigma_3^{\frac{1}{2}} \Phi_2(y)\rangle \quad s.t. \quad \Sigma_3 = diag(a^{(1)} \odot a^{(1)} + a^{(2)} \odot a^{(2)}). \tag{B.1}$$

Proof. By the sum property of kernels,

$$\begin{aligned}
k_1(x, y) + k_2(x, y) &= \langle [a_1\ a_2] \odot [\Phi_2(x)\ \Phi_2(x)], [a_1\ a_2] \odot [\Phi_2(y)\ \Phi_2(y)]\rangle \\
&= \langle a^{(1)} \odot \Phi_2(x), a^{(1)} \odot \Phi_2(y)\rangle + \langle a^{(2)} \odot \Phi_2(x), a^{(2)} \odot \Phi_2(y)\rangle \\
&= \langle a^{(1)} \odot a^{(1)} \odot \Phi_2(x), \Phi_2(y)\rangle + \langle a^{(2)} \odot a^{(2)} \odot \Phi_2(x), \Phi_2(y)\rangle \\
&= \langle a^{(1)} \odot a^{(1)} \odot \Phi_2(x) + a^{(2)} \odot a^{(2)} \odot \Phi_2(x), \Phi_2(y)\rangle \qquad \text{(B.2)} \\
&= \langle (a^{(1)} \odot a^{(1)} + a^{(2)} \odot a^{(2)}) \odot \Phi_2(x), \Phi_2(y)\rangle \\
&= \Phi_2^T(x)\ \text{diag}((a^{(1)} \odot a^{(1)} + a^{(2)} \odot a^{(2)})\ \Phi_2(y) \\
&= k_3(x, y).
\end{aligned}$$

□

By Lemma B.2.1, it suffices to write out the feature map of each kernel in Definition 3.4.2. The induced feature maps of each respective kernel term in Definition 3.4.2 are given by $a_i \odot \Phi_2(x), 1 \leq i \leq 4$ for

$$\begin{aligned}
a_1 &:= ((\lambda_1^{(m)})^2, \cdots, (\lambda_p^{(m)})^2, \sqrt{2}\lambda_1^{(m)}\lambda_2^{(m)}, \cdots, \sqrt{2}\lambda_{p-1}^{(m)}\lambda_p^{(m)}, \sqrt{2}\lambda_1^{(m)}, \cdots, \sqrt{2}\lambda_p^{(m)}, 1) \\
a_2 &:= (0, \cdots, 0, 0, \cdots, 0, \alpha_1, \cdots, \alpha_p, \sqrt{A}) \\
a_3 &:= (\psi_1, \cdots, \psi_p, 0, \cdots, 0, 0, \cdots, 0, 0) \\
a_4 &:= (0, \cdots, 0, 0, \cdots, 0, \sqrt{\nu^{(m)}}, 0, \cdots, 0, 0, \cdots, 0, 0)
\end{aligned} \tag{B.3}$$

The first claim follows from Eq. (B.3) and Lemma B.2.1.

To prove the second claim, take an arbitrary diagonal prior covariance matrix $S \in \mathbb{R}^{\dim(\Phi_2)\times\dim(\Phi_2)}$. It suffices to show that there exists a solution of,

$$\begin{aligned}
\text{diag}(S)_{(i)} &= \alpha_i^2 + 2\sum_{m=1}^{M_1}\left[\lambda_i^{(m)}\right]^2 \\
\text{diag}(S)_{(ij)} &= 2\sum_{m=1}^{M_1}\left[\lambda_i^{(m)}\lambda_j^{(m)}\right]^2 + \sum_{m:i_m=i,j_m=j}^{K_2}\nu^{(m)} \\
\text{diag}(S)_{(ii)} &= \psi_i^2 + \sum_{m=1}^{M_1}\left[\lambda_i^{(m)}\right]^4 \\
\text{diag}(S)_{(0)} &= M_2 + A.
\end{aligned}$$

for some choice of $M_1, M_2 \in \mathbb{N}$, $\alpha, \psi, \lambda^{(m)} \in \mathbb{R}_+^p$ $(m = 1, \ldots, M_1)$, $\nu^{(m)} \in \mathbb{R}_+$ $(m = 1, \ldots, M_2)$, and $A \in \mathbb{R}$. Take $\alpha_i^2 = \text{diag}(S)_{(i)}$ and $\psi_i^2 = \text{diag}(S)_{(ii)}$, for $i = 1, \cdots, p$. Take $\lambda^{(m)} = 0$. Let $M_2 = \frac{p(p-1)}{2}$ and $\nu^{(1)} = \text{diag}(S)_{(12)}, \cdots, \nu^{(M_2)} = \text{diag}(S)_{((p-1)p)}$. Finally, letting $A = \text{diag}(S)_{(0)} - M_2$ solves the system.

Remark. While we have shown one of the *many* ways to solve the above system for an arbitrary S, the strategy taken above is not practically useful; computing the kernel in this fashion will take $\Theta(p^2)$ time because $M_2 = \Theta(p^2)$. In practice, we must leverage the polynomial kernels (i.e., those in the M_1 sum) to avoid making M_2 large. We show how such a strategy works in Appendix B.3.

B.2.3 Proof of Theorem 3.5.1

Define $g(A^{ij}) := (g(e_i), g(-e_i), g(e_j), g(e_{ij}))$. Then,

$$\begin{aligned}
g(A^{ij}) \mid D, \tau \sim \mathcal{N}(\mu_{g_{ij}}, \Sigma_{ij}) \quad \text{s.t.} \quad \mu_{g_{ij}} &:= K_\tau(A^{ij}, X)H_\tau Y, \\
\Sigma_{ij} &:= \left[K_\tau(A^{ij}, A^{ij}) - K_\tau(A^{ij}, X)H_\tau K_\tau(X, A^{ij})\right],
\end{aligned} \tag{B.4}$$

which follows directly from Rasmussen and Williams [2006, Equation 2.21]. Notice that,

$$\theta_{x_i} = \frac{g(e_1)}{2} - \frac{g(-e_1)}{2} = a_i^T g(A^{ij}) \quad \text{and} \quad \theta_{x_i x_j} = \frac{g(e_1)}{2} - \frac{g(-e_1)}{2} - g(e_j) + g(e_{ij}) = a_{ij}^T g(A^{ij}), \tag{B.5}$$

where $a_i = (1/2, -1/2, 0, 0)$ and $a_{ij} = (-1/2, 1/2, -1, 1)$. Furthermore, $\theta_{x_i^2} = a_{ii}^T g(A^{ij})$ for $a_{ii} = (1/2, 1/2, 0, 0)$. The proof follows from Eq. (B.4), Eq. (B.5), and recalling that an affine transformation $h : x \mapsto Ax$ of a multivariate Gaussian distribution $Z \sim \mathcal{N}(\mu, \Sigma)$ is given by $h(Z) \sim \mathcal{N}(A\mu, A\Sigma A^T)$.

B.2.4 Proof of Corollary 3.5.2

Corollary 3.5.2 follows immediately once we can show that $K_\tau(A_{ij}, X)$ takes $O(1)$ time. It suffices to show $k_\tau(x^{(n)}, e_i)$ and $k_\tau(x^{(n)}, e_i + e_j)$ take $O(1)$ time. Since k_τ is a sum of polynomial kernels, $k_\tau(x, y)$ only depends on $x, y \in \mathbb{R}^p$ through the inner product $x^T y$. Hence, for vectors $\tilde{x}, \tilde{y} \in \mathbb{R}^M$, $k_\tau(\tilde{x}, \tilde{y})$ is well-defined and just depends on $\tilde{x}^T \tilde{y}$. Now, $k_\tau(x^{(n)}, e_i) = k_\tau(x_i^{(n)}, 1)$ and $k_\tau(x^{(n)}, e_i + e_j) = k_\tau((x_i^{(n)}, x_j^{(n)}), (1, 1))$. Since $k_\tau(x_i^{(n)}, 1)$ and $k_\tau((x_i^{(n)}, x_j^{(n)}), (1, 1))$ do not depend on p, these terms each take $O(1)$ time to compute.

B.2.5 The General Kernel Interaction Trick

In this section, we generalize the kernel interaction trick, namely show how to access the distribution of arbitrary components of θ. First, we require some new notation. For $E \subseteq \{1, \cdots, p\}$, $|E| = M$, define

$$\theta_E \coloneqq (\theta_{x_{i_1}}, \cdots, \theta_{x_{i_M}}, \theta_{x_{i_1} x_{i_2}}, \cdots, \theta_{x_{i_{M-1}} x_{i_M}}), \quad i_j \in E. \tag{B.6}$$

We show how to compute $\theta_E \mid \tau, D$ from the GP posterior predictive distribution. Without any lost of generality, we may assume $E = \{1, \cdots, M\}$ by relabeling the covariates.

Theorem B.2.2. *(General kernel interaction trick) Let $H_\tau \coloneqq (K_\tau + \sigma^2 I_N)^{-1}$ and*

$$A_M \coloneqq [e_1, -e_1, \cdots e_M, -e_M, e_1 + e_2, \cdots, e_{M-1} + e_M]^T.$$

Let $K_\tau(A_M, X) = K_\tau(X, A_M)^T$ be the matrix formed by taking the kernel between each row of A_M with each row of X. Let

$$\begin{aligned} a_i &\coloneqq (0, 0, \cdots, 1/2, -1/2, \cdots, 0, 0, \cdots, 0) \in \mathbb{R}^{2M + \frac{M(M-1)}{2}} \\ a_{ij} &\coloneqq (0, 0, \cdots, 1/2, -1/2, \cdots, -1, \cdots, 0, 0, \cdots, 1, \cdots, 0) \in \mathbb{R}^{2M + \frac{M(M-1)}{2}} \end{aligned} \tag{B.7}$$

for $i < j$. That is, a_i has non-zero entries at e_i and $-e_i$ and a_{ij} has non-zero entries *at e_i, $-e_i$, $-e_j$, and $e_i + e_j$. Let*

$$R_M \coloneqq [a_1 \cdots a_M \quad a_{12} \cdots a_{(M-1)M}]^T. \tag{B.8}$$

Then, $\theta_E \mid \tau, D$ is a multivariate Gaussian distribution with mean $R_M K_\tau(A_M, X) H_\tau Y$ and covariance matrix

$$R_M \left[K_\tau(A_{ij}, A_{ij}) - K_\tau(A_{ij}, X) H_\tau K_\tau(X, A_{ij})\right] R_M^T.$$

Proof. Following the proof of Theorem 3.5.1,

$$\begin{aligned} g(A^M) \mid D, \tau \sim \mathcal{N}(\mu_{g_M}, \Sigma_M) \quad \text{s.t.} \quad \mu_{g_M} &\coloneqq K_\tau(A^M, X) H_\tau Y, \\ \Sigma_M &\coloneqq \Big[K_\tau(A^M, A^M) - K_\tau(A^M, X) H_\tau K_\tau(X, A^M)\Big]. \end{aligned} \tag{B.9}$$

Similar to Eq. (B.5),

$$\theta_{x_i} = \frac{g(e_1)}{2} - \frac{g(-e_1)}{2} = a_i^T g(A^M) \quad \text{and} \quad \theta_{x_i x_j} = \frac{g(e_1)}{2} - \frac{g(-e_1)}{2} - g(e_j) + g(e_{ij}) = a_{ij}^T g(A^M). \tag{B.10}$$

The proof follows from Eq. (B.9), Eq. (B.10), and recalling that an affine transformation $h : x \mapsto R_M^T x$ of a multivariate Gaussian distribution $Z \sim \mathcal{N}(\mu, \Sigma)$ is given by $h(Z) \sim \mathcal{N}(R_M \mu, R_M \Sigma R_M^T)$.

□

Corollary B.2.3. *Given K_τ, the distribution $\theta_E \mid \tau, D$ takes $O(M^2)$ time and memory to compute.*

Proof. The proof is identical to the one provided in Appendix B.2.4. □

B.2.6 Proof of Proposition 3.6.1

See Appendix B.3.2.

B.3 Example Bayesian Interaction Models

In the following subsections, we show how to solve Eq. (3.9) for several classes of models.

B.3.1 Block-Degree Priors

Suppose we would like to set the prior variance of all terms with the same degree equal. That is, we would like to use a prior of the form

$$\begin{aligned}
\eta \in \mathbb{R}^3 &\sim p(\eta) \\
\theta_{x_i} \mid \eta &\sim \mathcal{N}(0, \eta_1^2) \\
\theta_{x_i x_j} \mid \eta &\sim \mathcal{N}(0, \eta_2^2) \\
\theta_{x_i^2} \mid \eta &\sim \mathcal{N}(0, \eta_3^2) \\
\theta_0 \mid c^2 &\sim \mathcal{N}(0, c^2).
\end{aligned} \tag{B.11}$$

To find the corresponding kernel, let $\lambda = (\frac{1}{\sqrt[4]{2}}\sqrt{\eta_2}, \cdots, \frac{1}{\sqrt[4]{2}}\sqrt{\eta_2})$, $M_1 = 1$ and $M_2 = 0$. Then, $\text{diag}(S)_{(ij)} = \eta_2^2$. Setting $\psi_i^2 = \eta_3^2 - \frac{1}{2}\eta_2^2$, implies that $\text{diag}(S)_{(ii)} = \eta_3^2$. Finally, letting $\alpha_i^2 = \tau_1^2 - \frac{2\eta_2}{\sqrt{2}}$ and $A = c^2 - 1$ implies that $\text{diag}(S)_{(i)} = \eta_1^2$ and $\text{diag}(S)_{(0)} = c^2$ as desired. We may equivalently re-write the induced kernel as

$$k_{block,\eta}(x, y) = \frac{\eta_2^2}{2} k^1_{\text{poly},2}(x, y) + (\eta_3^2 - \frac{\eta_2^2}{2}) k^0_{\text{poly},1}(x \odot x, y \odot y) + \left(\eta_1^2 - \eta_2^2\right) k^0_{\text{poly},1}(x, y) + c^2 - \frac{\eta_2^2}{2}. \tag{B.12}$$

Hence, Eq. (B.11) admits a kernel that only takes $O(p)$ time to compute.

B.3.2 Sparsity Priors

By Lemma B.2.1, the sparsity prior model provided in Eq. (3.11) equals $k_{block,\eta}(\kappa \odot x, \kappa \odot y)$.

B.4 SKIM Model Details

We provide the full hierarchical form of SKIM next. SKIM is based closely on the *regularized horseshoe prior* Piironen and Vehtari [2017] and the model proposed in

Griffin and Brown [2017]:

$$\begin{aligned}
m^2 &\sim \text{InvGamma}(\alpha_1, \beta_1) \qquad \xi^2 \sim \text{InvGamma}(\alpha_2, \beta_2) \qquad c^2 \sim \text{InvGamma}(\alpha_3, \beta_3) \\
\psi^2 &\sim \text{InvGamma}(\alpha_4, \beta_4) \qquad \phi := \frac{s}{p-s}\frac{\sigma}{\sqrt{N}} \qquad \sigma \sim \mathcal{N}^+(0, \alpha_5) \\
\kappa_i &= \frac{m\lambda_i}{\sqrt{m^2 + \eta_1^2 \lambda_i^2}} \qquad \lambda_i \sim C^+(0,1) \\
\eta_1 &\sim C^+(0, \phi) \qquad \eta_2 = \frac{\eta_1^2}{m^2}\xi \qquad \eta_3 = \frac{\eta_1^2}{m^2}\psi \\
\theta_{x_i} \mid \eta, \kappa &\sim \mathcal{N}(0, \eta_1^2 \kappa_i^2) \\
\theta_{x_j} \mid \eta, \kappa &\sim \mathcal{N}(0, \eta_1^2 \kappa_j^2) \\
\theta_{x_i x_j} \mid \eta, \kappa &\sim \mathcal{N}(0, \eta_2^2 \kappa_i^2 \kappa_j^2) \\
\theta_{x_i^2} \mid \eta, \kappa &\sim \mathcal{N}(0, \eta_3^2 \kappa_i^4) \\
\theta_0 \mid c^2 &\sim \mathcal{N}(0, c^2),
\end{aligned}$$

where s, α_i, and β_i are user-specified hyperparameters, $C^+(0,1)$ is a half-Cauchy distribution, and $\mathcal{N}^+$ is a half-normal distribution. More details, such as selecting the hyperparameters, desirable properties, and interpretations of SKIM, are provided below.

B.4.1 SKIM Details

Recall that we are primarily interested in the case when θ is sparse and satisfies strong-hierarchy. In order to promote sparsity in the main effects, we require two ingredients: (1) a prior on the *global shrinkage* parameter η_1 and (2) a prior on the *local shrinkage* parameters contained in $\kappa \in \mathbb{R}^p$ Carvalho et al. [2009], Piironen and Vehtari [2017]. Conditional on η_1 and κ,

$$\theta_{x_i} \mid \kappa, \eta_1 \sim \mathcal{N}(0, \eta_1^2 \kappa_i^2), \quad i = 1, \cdots, p. \tag{B.13}$$

η_1 controls the overall sparsity level of the model; in particular, the model becomes sparser as η_1 decreases. If we expect s non-zero main effects, then setting $\eta_1 = \frac{s}{p-s}\frac{\sigma}{\sqrt{N}}$ will yield an expected prior sparsity level of s by Piironen and Vehtari [2017, Equation 3.12]. However, we often do not know exactly how to select s. Hence, Piironen and Vehtari [2017] instead draw,

$$\phi := \frac{s}{p-s}\frac{\sigma}{\sqrt{N}} \qquad \eta_1 \sim C^+(0, \phi), \tag{B.14}$$

to express the uncertainty of not knowing the true main effect sparsity level.

The prior variance of θ_{x_i} is non-negligible only when κ_i is large enough to escape the global shrinkage of η_1. Hence, we draw κ_i from a heavy-tailed distribution so that certain main effects can escape global shrinkage. Carvalho et al. [2009] suggest drawing κ_i from a half-Cauchy distribution since this distribution has fat tails and desirable shrinkage properties. However, such a prior often leads to undesirable numerical stability issues when using gradient-based MCMC methods such as NUTS [Piironen and Vehtari, 2017]. As a result, Piironen and Vehtari [2017] instead propose the *regularized horseshoe* prior, which truncates the half-Cauchy distribution to have support only on $[0, m)$ instead of $[0, \infty)$. This truncation (empirically) leads to better mixing properties, and is achieved by setting

$$\kappa_i = \frac{m\lambda_i}{\sqrt{m^2 + \eta_1^2 \lambda_i^2}}, \qquad \lambda_i \sim C^+(0, 1). \tag{B.15}$$

As $\lambda_i \to \infty$, $\kappa_i \to \frac{m}{\eta_1}$. Hence, as $\lambda_i \to \infty$, the prior variance of θ_{x_i} equals m. Since we might not know the scale m of the non-zero main effects, we place a prior on m, namely,

$$m^2 \sim \text{InvGamma}(\alpha_1, \beta_1) \tag{B.16}$$

for hyperparameters α_1 and α_2.

Next, we model the interactions. If strong-hierarchy holds, sparsity comes for free; if there are only $s \ll p$ non-zero main effects, then there are at most $\frac{s(s-1)}{2} \ll p^2$ possible pairwise interactions. We must be careful, however, because strong-hierarchy trivially holds; our main effect estimates will, with probability one, never equal zero because the prior variances of the main effects are greater than 0 with probability one by Eq. (B.13) and our choice of priors. Instead, we aim for a relaxed version of strong-hierarchy. Namely, that the prior variance of an interaction $\theta_{x_i x_j}$ is large only if θ_{x_i} and θ_{x_j} are both large. Notice that the prior variances on θ_{x_i} and θ_{x_j} are large only when κ_i and κ_j are sufficiently far from zero. Hence, it suffices to make the prior variance of $\theta_{x_i x_j}$ large only when κ_i and κ_j are both large.

Let $\tilde{\kappa}_i^2 = \frac{\eta_1^2}{m^2}\kappa_i^2$. Then, $0 \leq \tilde{\kappa}_i^2 \leq 1$ and $\tilde{\kappa}_i$ approaches 1 as $\lambda_i \to \infty$. Since, $\tilde{\kappa}_i^2$ and $\tilde{\kappa}_j^2$ are bounded by 1, $\tilde{\kappa}_i^2 \tilde{\kappa}_j^2$ will only be close to 1 when each term is close to one. That is, when both λ_i *and* λ_j are large, or equivalently when κ_i and κ_j are both large. Hence, it suffices to let

$$\begin{aligned} \theta_{x_i x_j} \mid \eta_1, \kappa &\sim \mathcal{N}(0, \xi^2 \tilde{\kappa}_i^2 \tilde{\kappa}_j^2) \\ &= \mathcal{N}(0, \eta_2^2 \kappa_i^2 \kappa_j^2) \quad \text{for} \quad \eta_2 := \frac{\eta_1^2}{m^2}\xi, \end{aligned} \tag{B.17}$$

to promote strong-hierarchy, where ξ has the interpretation of the scale of the non-zero interaction effects; as λ_i and λ_j tend to infinity, the prior variance of $\theta_{x_i x_j}$ approaches ξ^2. Since we might not know this scale, we draw

$$\xi^2 \sim \text{InvGamma}(\alpha_2, \beta_2), \tag{B.18}$$

for some choice of hyperparameters α_2 and β_2. Our choice of prior for $\theta_{x_i^2}$ is analogous to the above reasoning for the choice of prior on $\theta_{x_i x_j}$.

Main difference between SKIM and the model proposed in Griffin and Brown [2017]: Unlike in the model proposed in Griffin and Brown [2017], SKIM does not assume sparsity between the interactions once the main effects are known. In particular, suppose, without any loss of generality, that the first s components of λ are large, while the remaining $p-s$ components are very close to zero. Then, the only interactions with non-negligible prior variance are the interactions between the first s variables. The number of such interactions is $O(s^2)$.

Unlike in Griffin and Brown [2017], SKIM does not assume sparsity among these $O(s^2)$ interactions. We do not assume such sparsity because the true sparsity level s is often very small (e.g., as in genome-wide association studies), which means that s^2 is small. Hence, once we have identified which of the s variables have non-zero main effects, estimating $O(s^2)$ interactions from N datapoints is not statistically difficult relative to actually identifying the s non-zero main effects. In particular, the mean-squared error of estimating $O(s^2)$ parameters from N datapoints is $O\left(\sqrt{\frac{s^2}{N}}\right)$ by standard Bernstein-von Mises results and a union bound. Thus, if $s = o(\sqrt{N})$, we can accurately estimate $O(s^2)$ parameters.

B.5 Variable Selection Procedure

Suppose we sample $\tau^{(t)} \sim p(\tau \mid D)$ via our kernel interaction sampler. Then, we use these $\tau^{(t)}$ samples to perform variable selection in the following way. Without any loss of generality, suppose we are deciding whether or not to include the main effect θ_{x_i}. Below we will show how to construct an interval $(c_{\text{lower}}, c_{\text{upper}})$ for θ_{x_i}. If this interval does not contain zero, we select θ_{x_i}. This interval is constructed by averaging the posterior means and standard deviations of θ_{x_i} associated with each sampled $\tau^{(t)}$:

$$\mu_T := \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{p(\theta_{x_i} \mid D, \tau^{(t)})}[\theta_{x_i}] \quad \sigma_T := \frac{1}{T}\sum_{t=1}^{T} \mathrm{SD}_{p(\theta_{x_i} \mid D, \tau^{(t)})}[\theta_{x_i}]$$

$$c_{\text{lower}} := \mu_T - z\sigma_T \quad c_{\text{upper}} := \mu_T + z\sigma_T. \tag{B.19}$$

Here, $\mathrm{SD}_{q(\theta)}[\theta]$ denotes the standard deviation of θ with respect to $q(\theta)$. In our experiments, we set $z = 2.59$ which corresponds to the 99.5th percentile of a standard normal distribution.

Heuristic justification of our variable selection procedure. We might expect that the posterior $p(\theta_{x_i} \mid D)$ has two modes: one mode near zero when the prior variance of θ_{x_i} is small and another mode when the prior variance is large. Thus, the posterior mean μ_T will "shrink" the estimate of θ_{x_i} towards zero, where the amount of shrinkage depends on the posterior mass of each mode. To understand the variability of the posterior mean, we effectively average the variability within each mode in Eq. (B.19). This averaging of variability within modes has the advantage of not artificially increasing the variance (due to the modes being separated by regions of low-probability) but has the disadvantage of potentially underestimating our uncertainty. For example, suppose $\theta_{x_i} \mid D \overset{d}{=} .5\mathcal{N}(0, .1) + .5\mathcal{N}(2, .1)$. Then, $\mathbb{E}_{p(\theta_{x_i} \mid D)}[\theta_{x_i}] = 1$

and $\mathrm{SD}_{p(\theta_{x_i}|D)}[\theta_{x_i}] = 1$. In this case, we would not select θ_{x_i} if we required that the posterior mean is further than twice the posterior standard deviation. If we instead averaged the variance within the modes (which would equal .1), we would select θ_{x_i} as we do in Eq. (B.19).

While we found good empirical performance of our variable selection procedure in Section 6.5 (e.g., based on FDR) we nevertheless think that variable selection in multimodal posteriors is challenging, and an area of active research. An interesting future research direction would be to develop even better variable selection strategies for sparse interaction models or to rigorously understand the tradeoffs between different variable selection procedures.

B.6 Woodbury Identity and the Matrix Determinant Lemma

To compute the determinant in Eq. (3.5), one can perform a Cholesky decomposition of $\Sigma_{N,\tau} \in \mathbb{R}^{\dim(\Phi_2)\times\dim(\Phi_2)}$. Computing $\Sigma_{N,\tau}$ takes $O(p^4N)$ time and $O(p^2N+p^4)$ memory. Computing the Cholesky decomposition of $\Sigma_{N,\tau}$ takes $O(p^6)$ time and requires $O(p^4)$ memory. This factorization can be avoided through the Woodbury matrix lemma and matrix determinant lemma.

The Woodbury matrix identity implies that

$$(A^{-1} + UU^T)^{-1} = A - AU(I_K + U^T AU)^{-1}U^T A, \tag{B.20}$$

where $A \in \mathbb{R}^{M\times M}$, $U \in \mathbb{R}^{M\times K}$, and I_K is the $K \times K$ identity matrix. The matrix determinant lemma implies that

$$\det(A^{-1} + UU^T) = \det(I + U^T AU)\det(A^{-1}). \tag{B.21}$$

Then, by the Woodbury identity,

$$\Sigma_{\tau,N} = (\Sigma_\tau^{-1}+\frac{1}{\sigma^2}\Phi_2(X)^T\Phi_2(X))^{-1} = \Sigma_\tau-\Sigma_\tau\Phi_2(X)^T(I_N+\Phi_2(X)\Sigma_\tau\Phi_2(X)^T)^{-1}\Phi_2(X)\Sigma_\tau. \tag{B.22}$$

Computing $p(D \mid \tau)$ requires computing $\det(\Sigma_{\tau,N})$. By the matrix determinant lemma,

$$\det(\Sigma_{\tau,N}) = (\det(I_N + \Phi_2(X)\Sigma_\tau\Phi_2(X)^T)\det(\Sigma_\tau^{-1}))^{-1}. \tag{B.23}$$

When Σ_τ is diagonal, the determinant equals the product of the diagonal, and its inverse equals one over the diagonal. Both of these quantities can be computed in $O(p^2)$ time. Hence, the time complexity for computing $\det(\Sigma_{\tau,N})$ is dominated by computing $\det(I_N + \Phi_2(X)\Sigma_\tau\Phi_2(X)^T)$, which takes $O(N^2p^2 + N^3)$ time and $O(Np^2)$ memory to store $\Phi_2(X)$.

B.7 Standard Polynomial Kernel

The feature map induced by the standard degree two polynomial kernel is given by

$$\begin{aligned}\Phi^{c}_{\text{poly},2}(x) &\coloneqq (x_1^2, \cdots, x_p^2, \sqrt{2}x_1x_2, \cdots, \sqrt{2}x_{p-1}x_p, \sqrt{2c}x_1, \cdots, \sqrt{2c}x_p, c)\\ &= a_{poly,2} \odot \Phi_2(x),\ a_{poly,2} \coloneqq (1, \cdots, 1, \sqrt{2}, \cdots, \sqrt{2}, \sqrt{2c}, \cdots, \sqrt{2c}, c).\end{aligned} \tag{B.24}$$

Hence, Eq. (B.24) implies that

$$\operatorname{diag}(\Sigma_{poly,2}) = a_{poly,2} \odot a_{poly,2}. \tag{B.25}$$

Eq. (B.25) shows that the prior covariance of the interaction terms are given higher prior variance than the main effects when $c \leq 1$, which is often undesirable. Furthermore, this prior does not promote sparsity, which is typically expected in high-dimensional problems.

Appendix C

Appendix for "The SKIM-FA Kernel: High-Dimensional Variable Selection and Non-Linear Interaction Discovery in Linear Time"

C.1 Proofs

C.1.1 Proof of Proposition 4.3.2

It suffices to prove that $\mathcal{H}_V^o$ is an RKHS. First we prove that $\mathcal{H}_V^o$ is a Hilbert space. Since $\mathcal{H}_V^o \subset \mathcal{H}_V$, it suffices to show that $\mathcal{H}_V^o$ is a vector space and complete. To show that $\mathcal{H}_V^o$ is a vector space, take arbitrary $f, g \in \mathcal{H}_V^o$ and $\alpha, \beta \in R$. We want to show $\alpha f + \beta g \in \mathcal{H}_V^o$. Take an arbitrary $f_A \in \mathcal{H}_A$, $A \subsetneq V$. Then,

$$\begin{aligned} \langle \alpha f + \beta g, f_A \rangle_\mu &= \alpha \langle f, f_A \rangle_\mu + \beta \langle g, f_A \rangle_\mu \\ &= 0 \end{aligned}$$

since $f, g \in \mathcal{H}_V^o$. Hence, $\mathcal{H}_V^o$ is a vector space.

Suppose towards a contradiction that $\mathcal{H}_V^o$ is not complete. Then, since $\mathcal{H}_V$ is complete, there exists an $f' \in \mathcal{H}_V \setminus \mathcal{H}_V^o$ and Cauchy sequence $\{f_n\}_{n=1}^\infty$ such that $\lim_{n\to\infty} \|f' - f_n\|_{\mathcal{H}_V} = 0$, where $f_n \in \mathcal{H}_V^o$ and $\|\cdot\|_{\mathcal{H}_V}$ denotes the induced RKHS norm for $\mathcal{H}_V$. Then, there exists an $\epsilon > 0$ and $f_A \in \mathcal{H}_A$, $A \subsetneq V$ such that

$$\begin{aligned} \epsilon &= \langle f', f_A \rangle_\mu \\ &= \langle f' + f_m - f_m, f_A \rangle_\mu \\ &= \langle f' - f_m, f_A \rangle_\mu + \langle f_m, f_A \rangle_\mu \\ &= \langle f' - f_m, f_A \rangle_\mu \\ &\leq \|f' - f_m\|_\mu \|f_A\|_\mu \quad \text{(by Cauchy-Schwarz).} \end{aligned} \tag{C.1}$$

To reach a contraction, it suffices to show that there exists an $m < \infty$ such that

$\|f' - f_m\|_\mu < \frac{\epsilon}{\|f_A\|_\mu}$. To obtain this inequality, we upper bound $\|\cdot\|_\mu$ in terms of $\|\cdot\|_{\mathcal{H}_V}$. Let r_V be the reproducing kernel for $\mathcal{H}_V$. Then, for $f \in \mathcal{H}_V$,

$$\begin{aligned} |f(x)|^2 &= |\langle f, r_V(x,\cdot)\rangle_{\mathcal{H}_V}|^2 \quad \text{(by the reproducing property)} \\ &\leq \|f\|^2_{\mathcal{H}_V} r_V(x,x)^2 \quad \text{(by Cauchy-Schwarz).} \end{aligned} \tag{C.2}$$

Then,

$$\begin{aligned} \|f\|^2_\mu &= \int |f(x)|^2 d\mu \\ &\leq \|f\|^2_{\mathcal{H}_V} \int r_V(x,x)^2 d\mu \end{aligned} \tag{C.3}$$

Since $\mathcal{H}_V$ belongs to the space of square integrable functions, $\int r_V(x,x)^2 d\mu = M_V < \infty$. Hence,

$$\|f' - f_m\|_\mu \leq M_V \|f' - f_m\|^2_{\mathcal{H}_V} < \infty. \tag{C.4}$$

Since $\|f' - f_m\|^2_{\mathcal{H}_V} \to 0$, there exists an m such that $\|f' - f_m\|_\mu < \frac{\epsilon}{\|f_A\|_\mu}$. Hence, $\mathcal{H}^o_V$ is complete.

To complete the proof it suffices to show that the evaluation functional on $\mathcal{H}^o_V$ is a bounded operator. Since $\mathcal{H}_V$ is an RKHS there exists an $M_x < \infty$ such that for all $f \in \mathcal{H}_V$

$$|f(x)| \leq M_x \|f\|_{\mathcal{H}_V}. \tag{C.5}$$

Since $\mathcal{H}^o_V \subset \mathcal{H}_V$, then for all $g \in \mathcal{H}^o_V$,

$$|g(x)| \leq M_x \|g\|_{\mathcal{H}_V}. \tag{C.6}$$

C.1.2 Proof of Lemma 4.3.5

$$\begin{aligned} f^{(M)}(x) &= \sum_{m=1}^{M} \alpha_m k_\theta(x_m, x) \\ &= \sum_{m=1}^{M} \left(\sum_{V:|V|\leq Q} \theta_V k_V(x_m, x) \right) \\ &= \sum_{V:|V|\leq Q} \theta_V \left(\sum_{m=1}^{M} k_V(x_m, x) \right) \\ &= \sum_{V:|V|\leq Q} f_V(x). \end{aligned} \tag{C.7}$$

It remains to show that $f_V \in \mathcal{H}^o_V$. For all $m \in [M]$, $k_V(x_m, \cdot) \in \mathcal{H}^o_V$. Hence, $\theta_V \sum_{m=1}^{M} k_V(x_m, x) \in \mathcal{H}^o_V$ since $\mathcal{H}^o_V$ is a Hilbert space.

C.1.3 Proof of Proposition 4.4.1

We prove the claim using a constructive proof with $p = 2$ variables. Consider the function

$$f(x_1, x_2) = 1 + (x_1 - x_2)^{2k} I(|x_1| \leq M) I(|x_2| \leq M). \tag{C.8}$$

Suppose the joint distribution of (x_1, x_2) under μ equals

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right).$$

Then, the joint distribution of (x_1, x_2) under $\mu_\otimes$ equals

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right).$$

By symmetry,

$$\begin{aligned} \mathbb{E}_{\mu_\otimes}[f(x_1, x_2)] &= 2\mu_2(x_2 < 0)\mathbb{E}_{\mu_\otimes}[f(x_1, x_2) \mid x_2 < 0] \\ &\geq 2\mu_1(x_1 > c)\mu_2(x_2 < 0)\mathbb{E}_{\mu_\otimes}[f(x_1, x_2) \mid x_1 > c, x_2 < 0] \\ &= \mu_1(x_1 > c)\mathbb{E}_{\mu_\otimes}[f(x_1, x_2) \mid x_1 > c, x_2 < 0] \\ &\geq \mu_1(x_1 > c)c^{2k} I(|c| < M). \end{aligned} \tag{C.9}$$

Under μ, we may assume without loss of generality that

$$\begin{aligned} x_1 &\sim \mathcal{N}(0, 1) \\ \epsilon &\sim \mathcal{N}(0, 1) \quad \text{s.t.} \quad \epsilon \perp\!\!\!\perp x_1 \\ x_2 &= \rho x_1 + \sqrt{1 - \rho^2}\epsilon. \end{aligned}$$

Then,

$$\begin{aligned} \lim_{\rho \to 1} \mathbb{E}_\mu f(x_1, x_2) &= 1 + \lim_{\rho \to 1} \int (x_1 - x_2)^{2k} I(|x_1| \leq M) I(|x_2| \leq M) d\mu(x_1, x_2) \\ &= 1 + \lim_{\rho \to 1} \int (x_1 - \rho x_1 - \sqrt{1 - \rho^2}\epsilon)^{2k} I(|x_1| \leq M) I(|x_2| \leq M) d\mu_\otimes(x_1, \epsilon) \\ &= 1 + \int \lim_{\rho \to 1} (x_1 - \rho x_1 - \sqrt{1 - \rho^2}\epsilon)^{2k} I(|x_1| \leq M) I(|x_2| \leq M) d\mu_\otimes(x_1, \epsilon) \\ &= 1, \end{aligned} \tag{C.10}$$

where the second to last line follows from he Dominated Convergence Theorem since $f(x_1, x_2)$ is uniformly bounded by $(2M)^{2k}$. Since $\mathbb{E}_\mu f(x_1, x_2) > 1$ for $0 \leq \rho < 1$, there exists a sequence $\{\rho_k\}_{k=1}^\infty$ such that for all $k \in \mathbb{N}$, $1 < \mathbb{E}_{\mu_k} f(x_1, x_2) < 2$ and $0 < \rho_k < 1$, where μ_k sets $\rho = \rho_k$. Pick k' large enough so that $f_{\{\emptyset\}}^{\mu_\otimes} > 2$. Then, for

$k \geq k'$,

$$\begin{aligned} \frac{|f_{\{\emptyset\}}^{\mu_\otimes} - f_{\{\emptyset\}}^{\mu_k}|}{|f_{\{\emptyset\}}^{\mu_k}|} &\geq \frac{|f_{\{\emptyset\}}^{\mu_\otimes} - f_{\{\emptyset\}}^{\mu_k}|}{2} \\ &= \frac{f_{\{\emptyset\}}^{\mu_\otimes} - f_{\{\emptyset\}}^{\mu_k}}{2} \\ &> \frac{f_{\{\emptyset\}}^{\mu_\otimes} - 2}{2} \end{aligned} \tag{C.11}$$

Let $k^* = \max\left(k', \left\lceil .5\sqrt[c]{\frac{2(\Delta+1)}{\mu_1(x_1>c}}\right\rceil\right)$. Then, by Eq. (C.9) and Eq. (C.11), $\frac{|f_{\{\emptyset\}}^{\mu_\otimes} - f_{\{\emptyset\}}^{\mu_{k^*}}|}{|f_{\{\emptyset\}}^{\mu_{k^*}}|} > \Delta$.

C.1.4 Proof of Lemma 4.3.4

By equation 2.25 of Rasmussen and Williams [2006, Chapter 2], Eq. (4.6) equals the posterior predictive mean of the following Bayesian model:

$$\begin{aligned} f &\sim GP(0, k_\theta) \\ y \mid f, x &\sim \mathcal{N}(f(x), \sigma_{\text{noise}}^2 = \lambda). \end{aligned}$$

We may re-write k_θ as,

$$\begin{aligned} k_\theta(x, \tilde{x}) &= \sum_{V:|V|\leq Q} \theta_V \Phi_V^T(x) \Phi_V^T(\tilde{x}) \\ &= \sum_{V:|V|\leq Q} \Phi_V^T(x)[\theta_V I_{B^V \times B^V}] \Phi_V^T(\tilde{x}) \\ &= \sum_{V:|V|\leq Q} \Phi_V^T(x) \Sigma_V \Phi_V^T(\tilde{x}), \end{aligned}$$

where $\Sigma_V = \theta_V I_{B^V \times B^V}$. Then, by Rasmussen and Williams [2006, Chapter 2.1.2] and the additive property of kernels, $f \sim GP(0, k_\theta)$ has the same distribution as drawing a set of regression coefficients $\Theta_V \sim \mathcal{N}(0, \Sigma_V)$ and setting $f = \sum_{V:|V|\leq Q} \Theta_V^T \Phi_V(\cdot)$. Hence, the posterior predictive mean of the Gaussian process at a point x equals $\sum_{V:|V|\leq Q} \hat{\Theta}_V^T \Phi_V(x)$.

C.1.5 Proof of Theorem 4.3.9

$$\begin{aligned}
k_{\text{SKIM-FA}}(x,\tilde{x}) &= \sum_{V:|V|\leq Q}\left[\eta_{|V|}^2\prod_{i\in V}\kappa_i^2\right]k_V(x,\tilde{x}) \\
&= \sum_{V:|V|\leq Q}\left[\eta_{|V|}^2\prod_{i\in V}\kappa_i^2\right]\prod_{i\in V}k_i(x_i,\tilde{x}_i) \\
&= \sum_{V:|V|\leq Q}\left[\eta_{|V|}^2\prod_{i\in V}\kappa_i^2 k_i(x_i,\tilde{x}_i)\right] \qquad \text{(C.12)} \\
&= \sum_{q=1}^{Q}\sum_{V:|V|=Q}\left[\eta_{|V|}^2\prod_{i\in V}\kappa_i^2 k_i(x_i,\tilde{x}_i)\right] \\
&= \sum_{q=1}^{Q}\eta_q^2\sum_{V:|V|=q}\left[\prod_{i\in V}\kappa_i^2 k_i(x_i,\tilde{x}_i)\right]
\end{aligned}$$

Let $\tilde{k}_i(\cdot,\cdot)=\kappa_i^2 k_i(\cdot,\cdot)$. Then, Vapnik [1995, pg. 199] shows that

$$\bar{k}_q \coloneqq \sum_{V:|V|=q}\prod_{i\in V}\tilde{k}_i = \frac{1}{q}\sum_{s=1}^{q}(-1)^{s+1}\bar{k}_{q-s}k^s, \qquad \text{(C.13)}$$

where $k^s(x,\tilde{x})=\sum_{i=1}^p[\tilde{k}_i(x_i,\tilde{x}_i)]^s$ and $\bar{k}_0(x,\tilde{x})=1$. The result follows from Eq. (C.12) and Eq. (C.13).

C.1.6 Proof of Corollary 4.3.10

Computing and storing $k^1(x,\tilde{x}),\cdots,k^Q(x,\tilde{x})$ takes $O(pQ)$ time and requires $O(Q)$ memory, respectively. After computing and storing $\bar{k}_1(x,\tilde{x}),\cdots,\bar{k}_q(x,\tilde{x})$, $\bar{k}_{q+1}(x,\tilde{x})$ takes $O(q+1)$ time. Hence, computing all $\bar{k}_1(x,\tilde{x}),\cdots,\bar{k}_Q(x,\tilde{x})$ terms takes $O(Q^2)$ time given $k^1(x,\tilde{x}),\cdots,k^Q(x,\tilde{x})$. Since $Q<p$, computing $k_{\text{SKIM-FA}}(x,\tilde{x})$ takes $O(pQ)$ time.

C.1.7 Proof of Proposition 4.5.1

$$\begin{aligned}
\frac{\partial L}{\partial \tilde{U}_i^{(t)}} &= \frac{\partial L}{\partial \kappa_i^{(t)}}\frac{\partial \kappa_i}{\partial U_i^{(t)}}\frac{\partial U_i^{(t)}}{\partial \tilde{U}_i^{(t)}} \\
&= \frac{\partial L}{\partial \kappa_i^{(t)}}I(U_i^{(t)}>c)\frac{2\tilde{U}_i^{(t)}}{(\tilde{U}_i^{(t)}+1)^2}.
\end{aligned}$$

Since $\kappa_i^{(t)} = 0$, that implies $U_i^{(t)} \leq c$. Hence, $\frac{\partial L}{\partial \tilde{U}_i^{(t)}} = 0$. Consequently,

$$\begin{aligned} \tilde{U}_i^{(t+1)} &= \tilde{U}_i^{(t)} - \gamma \frac{\partial L}{\partial \tilde{U}_i^{(t)}} \\ &= \tilde{U}_i^{(t)}. \end{aligned} \tag{C.14}$$

By Eq. (C.14), $\kappa_i^{(t')} = 0$ for all $t' \geq t$.

C.1.8 Proof of Lemma 4.4.4

It suffices to prove that any $f_V \in \mathcal{H}_V$ is square-integrable with respect to any probability measure. Since ϕ_{ib} is a continuous function on a compact set, there exists a $0 < M_{ib} < \infty$ such that $|\phi_{ib}|$ is bounded by M_{ib}. Without loss of generality, assume $V = \{1, \cdots, q\}$. Then, there exists coefficients $c_{b_1,\cdots,b_q} \in \mathbb{R}$ such that

$$\begin{aligned} f_V(x_V) &= \sum_{b_1 \in [B_1]} \cdots \sum_{b_q \in [B_q]} c_{b_1,\cdots,b_q} \prod_{i=1}^{q} \phi_{ib_i}(x_i) \\ &\leq \sum_{b_1 \in [B_1]} \cdots \sum_{b_q \in [B_q]} c_{b_1,\cdots,b_q} M_*^q \\ &< \infty \end{aligned}$$

for all x_V, where $M_* = \max_{i \in [p]} \max_{b \in [B_i]} M_{ib} < \infty$ since $B_i < \infty$. Hence, for any probability measure μ,

$$\begin{aligned} \int |f_V(x_V)|^2 d\mu &< \int \left(\sum_{b_1 \in [B_1]} \cdots \sum_{b_q \in [B_q]} c_{b_1,\cdots,b_q} M_*^q \right)^2 d\mu \\ &= \left(\sum_{b_1 \in [B_1]} \cdots \sum_{b_q \in [B_q]} c_{b_1,\cdots,b_q} M_*^q \right)^2 \\ &< \infty. \end{aligned} \tag{C.15}$$

C.1.9 Proof of Theorem 4.4.5

Let

$$\begin{aligned} \tilde{f}_{ij} &= f_{\{i,j\}}^{\mu_\otimes} - [\Psi_{ij}^i \Phi_i + \Psi_{ij}^j \Phi_j + \Psi_{ij}^0] \\ \tilde{f}_i &= f_{\{i\}}^{\mu_\otimes} + \sum_{j>i} \Psi_{ij}^i \Phi_i + \sum_{j<i} \Psi_{ji}^i \Phi_i^T(x_i) \\ \tilde{f}_\emptyset &= f_\emptyset^{\mu_\otimes} + \sum_{i<j} \Psi_{ij}^0. \end{aligned} \tag{C.16}$$

We start by proving that $f = \tilde{f}_\emptyset + \sum_{i=1}^p \tilde{f}_i + \sum_{i,j=1}^p \tilde{f}_{ij}$. Expanding each component,

$$\begin{aligned}
\tilde{f}_\emptyset + \sum_i \tilde{f}_i + \sum_{i<j} \tilde{f}_{ij} &= \tilde{f}_\emptyset + \sum_i \tilde{f}_i + \sum_{i<j} \left[f^{\mu_\otimes}_{\{i,j\}} - [\Psi^i_{ij}\Phi_i + \Psi^j_{ij}\Phi_j + \Psi^0_{ij}] \right] \\
&= f^{\mu_\otimes}_\emptyset + \sum_i \tilde{f}_i + \sum_{i<j} \left[f^{\mu_\otimes}_{\{i,j\}} - [\Psi^i_{ij}\Phi_i + \Psi^j_{ij}\Phi_j] \right] \\
&= f^{\mu_\otimes}_\emptyset + \sum_i \left[f^{\mu_\otimes}_{\{i\}} + \sum_{j>i} \Psi^i_{ij}\Phi_i + \sum_{j<i} \Psi^i_{ji}\Phi_i \right] + \\
&\quad \sum_{i<j} \left[f^{\mu_\otimes}_{\{i,j\}} - [\Psi^i_{ij}\Phi_i + \Psi^j_{ij}\Phi_j] \right] \\
&= f^{\mu_\otimes}_\emptyset + \sum_i f^{\mu_\otimes}_{\{i\}} + \sum_{i<j} f^{\mu_\otimes}_{\{i,j\}} + \\
&\quad \sum_i \sum_{j>i} \Psi^i_{ij}\Phi_i + \sum_i \sum_{j<i} \Psi^i_{ji}\Phi_i - \sum_{i<j} \left[\Psi^i_{ij}\Phi_i + \Psi^j_{ij}\Phi_j \right] \\
&= f + \sum_i \sum_{j>i} \Psi^i_{ij}\Phi_i + \sum_i \sum_{j<i} \Psi^i_{ji}\Phi_i - \sum_i \sum_{j>i} \left[\Psi^i_{ij}\Phi_i + \Psi^j_{ij}\Phi_j \right] \\
&= f + \sum_i \sum_{j<i} \Psi^i_{ji}\Phi_i - \sum_i \sum_{j>i} \Psi^j_{ij}\Phi_j \\
&= f + \sum_i \sum_{j<i} \Psi^i_{ji}\Phi_i - \sum_j \sum_{j<i} \Psi^i_{ji}\Phi_j \\
&= f.
\end{aligned}$$

We now prove that there exists unique coefficients, $\Psi^i_{ij} \in \mathbb{R}^{1\times B_i}$, $\Psi^j_{ij} \in \mathbb{R}^{1\times B_j}$, $\Psi^0_{ij} \in \mathbb{R}$, such that $\tilde{f}_{ij}$ belongs to the orthogonal complement of the Hilbert space $\mathcal{H}^{ij}_{\text{add}} := \text{span}\{1, \{\phi_{ib}\}_{b=1}^{B_i}, \{\phi_{jb}\}_{b=1}^{B_j}\}$. Let $\mathcal{H}^{ij} := \text{span}\{1, \{\phi_{ib}\}_{b=1}^{B_i}, \{\phi_{jb}\}_{b=1}^{B_j}\}, \{\phi_{ib}\phi_{jb'}\}_{b\in[B_i],b'\in[B_j]}\}$. Then, $f^{\mu_\otimes}_{\{i,j\}} \in \mathcal{H}^{ij}$ and $\mathcal{H}^{ij}_{\text{add}}$ is a closed convex subspace of $\mathcal{H}^{ij}$. Therefore, by the Hilbert Projection Theorem, there exists unique $\bar{f}_{ij} \in \mathcal{H}^{ij}_{\text{add}}$ and $f^\perp_{ij} \in \mathcal{H}^{ij}$ such that

$$\begin{aligned}
f^{\mu_\otimes}_{\{i,j\}} = \bar{f}_{ij} + f^\perp_{ij} \quad \text{s.t.} \\
\langle g, f^\perp_{ij} \rangle_\mu = 0 \quad \forall g \in \mathcal{H}^{ij}_{\text{add}}.
\end{aligned} \tag{C.17}$$

Since $\text{span}\{1, \{\phi_{ib}\}_{b=1}^{B_i}, \{\phi_{jb}\}_{b=1}^{B_j}\}$ is a linearly independent basis of $\mathcal{H}^{ij}_{\text{add}}$, there exists unique coefficients,$\Psi^i_{ij} \in \mathbb{R}^{1\times B_i}$, $\Psi^j_{ij} \in \mathbb{R}^{1\times B_j}$, $\Psi^0_{ij} \in \mathbb{R}$, such that $\bar{f}_{ij} = \Psi^i_{ij}\Phi_i^T(x_i) + \Psi^j_{ij}\Phi_j^T(x_j) + \Psi^0_{ij}$.

To complete the proof, we need to show that $\tilde{f}_{ij} = f^\mu_{\{i,j\}}, \tilde{f}_i = f^\mu_{\{i\}}, \tilde{f}_\emptyset = f^\mu_\emptyset$. It suffices to show that

$$\begin{aligned}
&\int_{x_i} \tilde{f}_i \, d\mu_i = 0 \\
&\int_{x_i, x_j} \tilde{f}_{ij} \, d\mu_i = 0 \\
&\int_{x_i, x_j} \tilde{f}_i \tilde{f}_{ij} \, d\mu(x_i, x_j) = 0.
\end{aligned} \tag{C.18}$$

The last two equalities in Eq. (C.18) follow directly from Eq. (C.17). For the first equality in Eq. (C.18), notice that

$$\begin{aligned}
\int_{x_i} \tilde{f}_i \, d\mu_i &= \mathbb{E}_{\mu_i} \tilde{f}_i \\
&= \mathbb{E}_{\mu_i} \left[f^{\mu_\otimes}_{\{i\}} + \sum_{j>i} \Psi^i_{ij} \Phi_i + \sum_{j<i} \Psi^i_{ji} \Phi_i \right] \\
&= \mathbb{E}_{\mu_i} f^{\mu_\otimes}_{\{i\}} + \sum_{j>i} \mathbb{E}_{\mu_i} [\Psi^i_{ij} \Phi_i] + \sum_{j<i} \mathbb{E}_{\mu_i} [\Psi^i_{ji} \Phi_i] \\
&= \sum_{j>i} \Psi^i_{ij} \mathbb{E}_{\mu_i} [\Phi_i] + \sum_{j<i} \Psi^i_{ji} \mathbb{E}_{\mu_i} [\Phi_i] \\
&= 0,
\end{aligned}$$

where the last equation follows from the fact that the components of Φ_i span $\mathcal{H}^o_i$ (and hence are all zero mean).

C.1.10 Proof of Proposition 4.5.2

As shown in the proof of Theorem 4.4.5, $\Psi^i_{ij} \in \mathbb{R}^{1 \times B_i}, \Psi^j_{ij} \in \mathbb{R}^{1 \times B_j}, \Psi^0_{ij} \in \mathbb{R}$ equal the unique set of coefficients such that $\bar{f}_{ij} = \Psi^i_{ij} \Phi_i + \Psi^j_{ij} \Phi_j + \Psi^0_{ij}$ for $\bar{f}_{ij}$ defined in Eq. (C.17) and also shown below:

$$\begin{aligned}
f^{\mu_\otimes}_{\{i,j\}} &= \bar{f}_{ij} + f^\perp_{ij} \quad \text{s.t.} \\
&\langle g, f^\perp_{ij} \rangle_\mu = 0 \quad \forall g \in \mathcal{H}^{ij}_{\text{add}}.
\end{aligned}$$

Let $y^{(w)}_{ij} = f^{\mu_\otimes}_{\{i,j\}}(x^{(m)}_i, x^{(w)}_j)$ and $\epsilon^{(w)}_{ij} = f^\perp_{ij}(x^{(w)}_i, x^{(w)}_j)$, where $x^{(w)} \overset{\text{iid}}{\sim} \mu$. Then,

$$y^{(w)}_{ij} = \Psi^i_{ij} \Phi_i(x^{(w)}_i) + \Psi^j_{ij} \Phi_j(x^{(w)}_j) + \Psi^0_{ij} + \epsilon^{(w)}_{ij} \quad x^{(w)} \overset{\text{iid}}{\sim} \mu. \tag{C.19}$$

Then, Eq. (C.19) is a special case of the random design linear model under misspecification studied in Hsu et al. [2014]. Hence, by Hsu et al. [2014, Theorem 11] we can consistently recover Ψ^i_{ij}, Ψ^j_{ij}, Ψ^0_{ij} by using ordinary least-squares. Hence Algorithm 5 recovers Ψ^i_{ij}, Ψ^j_{ij}, Ψ^0_{ij} as $W \to \infty$.

C.2 Literature Review

Finite Basis Expansion Methods. Stone [1994] introduced the hierarchical functional decomposition and derived statistical rates of convergence by approximating $\mathcal{H}$ using a finite B-spline tensor product basis. Huang [1998] later extended this result to general tensor product families such as wavelets, polynomials, etc. There have been a number of specific Bayesian and frequentist methods that fall within the general class of models described in Huang [1998]; see, for example, Wei et al. [2019], Scheipl et al. [2012], Curtis et al. [2014]. Unfortunately, since these methods explicitly generate the tensor product basis, they are computationally intractable for even moderately sized problems.

Linear models trivially fall within this class as well. For $Q = 1$, the Lasso and the many related techniques provide fast variable selection and estimation in high-dimensional linear models [Chen et al., 1998, Candes and Tao, 2007, Nakagawa et al., 2016]. For $Q = 2$, the hierarchical Lasso [Bien et al., 2013] extends the Lasso to model interactions, and there have been many variants of this model; see, for example, Lim and Hastie [2015], Shah [2016]. However, these methods take at least $O(p^2)$ time since they explicitly model all main and interaction effects.

Two-Stage & Forward-Stage Approaches. Instead of modeling interactions jointly, a common heuristic (similar in spirit to forward stepwise regression) is greedily adding interactions such as in multivariate additive regression splines (MARS) or GA2M [Lou et al., 2013]. Another common approach is performing computationally cheap variable selection methods designed for generalized additive models (e.g., Lasso or SpAM [Liu et al., 2008]) to identify a sparse set of relevant variables. By restricting to a small set of variables, one can then apply more computationally intensive interactions techniques such as RKHS ANOVA methods.

Either of the two approaches above requires some form of strong-hierarchy, namely that all interactions have non-zero main effects, to consistently identify the correct set of variables. While some problems have strong main effects, in other applications this may not be the case. For example, in genome-wide associate studies, fitting an additive-only model to predict an individual's height from genetics only has an R^2 of about 5% even though height is well-predicted by parents' heights (thought to be between $80\% - 90\%$) [Maher, 2008]. This discrepancy, more generally called the problem of *missing heritability*, remains an open challenge in biology for understanding complex diseases based on genetics. One explanation for missing heritability is not modeling genetic interactions [Maher, 2008, Aschard, 2016, Slim et al., 2018, Greene et al., 2010]. In other words, the main effects might be weak, or in the extreme case some genes might only have interaction effects. Hence, from a purely variable selection standpoint, modeling interactions could help better identify genes that are risk-factors for certain diseases.

In the orthogonal μ case, the statistical benefit from modeling interactions can be easily seen from the decomposition in Eq. (4.4). Suppose $Q = 2$ and that main effects total signal variance equals 5, the pairwise signal variance equals equals 90, and the noise variance equals 5. Then, the R^2 for an additive-only model is 5%

while the R^2 for interaction model is 90%. Since the achievable signal increases (and necessarily the effective noise variance decreases), performing variable selection in a lower signal-to-noise regime might offset the statistical price of modeling more parameters.

C.3 Zero Mean Kernels and Finite-Basis Functions

In this section, we show how we construct k_i, i.e., the reproducing kernel for $\mathcal{H}_i^o$. We construct k_i by first generating a finite-dimensional basis for $\mathcal{H}_i$. Then, we normalize each basis function to be zero mean and unit variance so that the normalized basis functions span $\mathcal{H}_i^o$. For a more general approach to construct zero mean kernels (e.g., even when $\mathcal{H}_i$ is infinite-dimensional) see Durrande et al. [2013].

Construction. For each covariate dimension i, consider a set of linearly independent basis functions $\{\phi_{ib}\}_{b=1}^{B_i}$ such that

$$\mathcal{H}_i = \mathrm{span}\{1, \phi_{i1}, \cdots, \phi_{iB_i}\}.$$

Let $\tilde{\phi}_{ib} = \frac{\phi_{ib} - \mathbb{E}_\mu[\phi_{ib}]}{\sqrt{\mathrm{Var}_\mu[\phi_{ib}]}}$. Then,

$$\mathcal{H}_i^o = \mathrm{span}\{\tilde{\phi}_{i1}, \cdots, \tilde{\phi}_{iB_i}\}, \quad \Phi_i := [\tilde{\phi}_{i1}, \cdots, \tilde{\phi}_{iB_i}].$$

Hence, $k_i(x_i, \tilde{x}_i) = \Phi_i(x_i)^T \Phi_i(\tilde{x}_i)$ is the reproducing kernel for $\mathcal{H}_i^o$. In many instances, we do not actually know the joint distribution of the covariates. In this case, we approximate μ with the empirical distribution $\hat{\mu}$ of the datapoints:

$$\begin{aligned}
\tilde{\phi}_{ib} &= \frac{\phi_{ib} - \mathbb{E}_{\hat{\mu}}[\phi_{ib}]}{\sqrt{\mathrm{Var}_{\hat{\mu}}[\phi_{ib}]}} \quad \text{s.t.} \\
\hat{\mu} &= \frac{1}{N}\sum_{n=1}^{N} \delta_{x^{(n)}} \\
\mathbb{E}_{\hat{\mu}}[\phi_{ib}] &= \frac{1}{N}\sum_{n=1}^{N} \phi_{ib}(x_i^{(n)}) \\
\mathrm{Var}_{\hat{\mu}}[\phi_{ib}] &= \frac{1}{N}\sum_{n=1}^{N} \phi_{ib}^2(x_i^{(n)}) - \mathbb{E}_{\hat{\mu}}[\phi_{ib}].
\end{aligned}$$

C.4 MARS ANOVA Procedure

We show how to perform the functional ANOVA decomposition of $\hat{f}$ with respect to $\hat{\mu}_\otimes = \hat{\mu}_1 \otimes \cdots \otimes \hat{\mu}_p$, where $\hat{f}$ denotes the regression function fit from MARS and μ_i the empirical distribution of covariate i: $\hat{\mu}_i = \frac{1}{N}\sum_{n=1}^{N} \delta_{x_i^{(n)}}$. Under $\hat{\mu}_\otimes$, the functional

ANOVA decomposition of $\hat{f}$ equals

$$\begin{aligned}
\hat{f}_{\emptyset} &= \mathbb{E}_{\hat{\mu}_{\otimes}}[\hat{f}] \\
\hat{f}_{\{i\}}(x_i) &= \mathbb{E}_{\hat{\mu}_{\otimes}}[\hat{f} \mid x_i = x_i] - \hat{f}_{\emptyset} \\
\hat{f}_{\{i,j\}}(x_i, x_j) &= \mathbb{E}_{\hat{\mu}_{\otimes}}[\hat{f} \mid x_i = x_i, x_j = x_j] - \hat{f}_{\{i\}}(x_i) - \hat{f}_{\{j\}}(x_i) - \hat{f}_{\emptyset},
\end{aligned} \tag{C.20}$$

which is also shown in Durrande et al. [2013, Equation 5]. We show how to compute each of the expectations in Eq. (C.20). The intercept $\hat{f}_{\emptyset}$ equals the sample average of the fitted values (i.e., $\hat{f}$ applied to each of the N training datapoints). Let X denote the $N \times p$ matrix of training data. Let X^i equal the matrix obtained by setting all values in the ith column of X equal to x_i and the remaining columns unchanged. Then,

$$\mathbb{E}_{\hat{\mu}_{\otimes}}[\hat{f} \mid x_i = x_i] = \frac{1}{N} \sum_{n=1}^{N} \hat{f}(X_n^i),$$

where X_n^i is the nth row of X_n^i. Similarly, let X^{ij} equal the matrix obtained by setting all values in the ith and jth columns of X equal to x_i and x_j respectively, and the remaining columns unchanged. Then,

$$\mathbb{E}_{\hat{\mu}_{\otimes}}[\hat{f} \mid x_i = x_i, x_j = x_j] = \frac{1}{N} \sum_{n=1}^{N} \hat{f}(X_n^{ij}).$$

C.5 Additional Experimental Results

Table C.1: Variable Selection Performance for Main Effects Only Setting.

Method	# **Covariates**	# **Correct Selected**	# **Wrong Selected**	# **Correct Not Se**
HierLasso	250	4	0	1
SKIM -FA	250	4	0	1
Pairs Lasso	250	4	5	1
SPAM-2Stage	250	5	52	0
MARS	250	5	58	0
SKIM-FA	500	5	13	0
SPAM-2Stage	500	5	28	0
Pairs Lasso	500	4	39	1
HierLasso	500	4	48	1
MARS	500	5	64	0
SKIM-FA	1000	3	0	2
HierLasso	1000	4	5	1
Pairs Lasso	1000	4	6	1
SPAM-2Stage	1000	5	15	0
MARS	1000	5	70	0

Table C.2: Estimation Performance for Main Effects Only Setting.

Method	**p**	**Correct Selected SSE (Main)**	**Correct Not Selected SSE (Main)**	**Wrong Selected SSE (Main)**	**Correct Selected SSE (Pair)**	**Correct Not Selected SSE (Pair)**	**Wrong Selected SSE (Pair)**	**Total SSE**	**Total SSE ÷ Signal Variance**
MARS-EMP	250	0.31	0.00	2.11	0.00	0.00	2.24	4.66	0.23
SPAM-2Stage	250	2.77	0.00	1.99	0.00	0.00	0.08	4.84	0.24
SKIM-FA	250	2.75	4.02	0.00	0.00	0.00	0.39	7.16	0.36
MARS-VANILLA	250	73.17	0.00	2.37	0.00	0.00	9.89	85.43	4.27
SPAM-2Stage	500	2.82	0.00	1.25	0.00	0.00	0.04	4.11	0.21
SKIM-FA	500	2.75	0.00	0.49	0.00	0.00	1.40	4.64	0.23
MARS-EMP	500	0.38	0.00	2.37	0.00	0.00	2.19	4.95	0.25
MARS-VANILLA	500	35.67	0.00	3.62	0.00	0.00	9.22	48.51	2.43
SPAM-2Stage	1000	2.67	0.00	0.78	0.00	0.00	0.02	3.46	0.17
MARS-EMP	1000	0.45	0.00	2.68	0.00	0.00	2.39	5.51	0.28
SKIM-FA	1000	2.70	8.10	0.00	0.00	0.00	0.24	11.03	0.55
MARS-VANILLA	1000	16.14	0.00	1.56	0.00	0.00	10.33	28.02	1.40

Table C.3: Variable Selection Performance for Equal Main and Interaction Effects Setting.

Method	**# of Covariates**	**# Correct Selected**	**# Wrong Selected**	**# Correct Not**
SKIM-FA	250	5	1	0
HierLasso	250	5	25	0
SPAM-2Stage	250	5	37	0
MARS	250	5	84	0
Pairs Lasso	250	5	89	0
SKIM-FA	500	5	0	0
SPAM-2Stage	500	5	29	0
HierLasso	500	5	30	0
MARS	500	5	69	0
Pairs Lasso	500	5	182	0
SKIM-FA	1000	5	0	0
SPAM-2Stage	1000	5	15	0
HierLasso	1000	5	40	0
MARS	1000	5	71	0
Pairs Lasso	1000	5	213	0

Table C.4: Estimation Performance for Equal Main and Interaction Effects Setting.

Method	**p**	**Correct Selected SSE (Main)**	**Correct Not Selected SSE (Main)**	**Wrong Selected SSE (Main)**	**Correct Selected SSE (Pair)**	**Correct Not Selected SSE (Pair)**	**Wrong Selected SSE (Pair)**	**Total SSE**	**Total SSE ÷ Signal Variance**
SKIM-FA	250	1.62	0.00	0.08	0.52	0.00	0.17	2.39	0.12
SPAM-2Stage	250	1.63	0.00	1.72	8.84	0.00	0.11	12.30	0.62
MARS-EMP	250	0.71	0.00	4.44	2.17	0.00	5.69	13.01	0.65
MARS-VANILLA	250	24.91	0.00	5.28	17.13	0.00	18.03	65.35	3.27
SKIM-FA	500	1.52	0.00	0.00	0.41	0.00	0.00	1.93	0.10
SPAM-2Stage	500	1.62	0.00	3.74	2.16	0.00	5.47	12.99	0.65
MARS-EMP	500	0.71	0.00	4.69	1.63	0.96	6.57	14.56	0.73
MARS-VANILLA	500	11.36	0.00	13.22	15.62	0.96	23.55	64.71	3.24
SKIM-FA	1000	1.54	0.00	0.00	0.29	0.00	0.00	1.82	0.09
SPAM-2Stage	1000	1.67	0.00	1.07	0.41	0.00	2.16	5.31	0.27
MARS-EMP	1000	0.61	0.00	3.84	1.70	0.00	2.52	8.67	0.43
MARS-VANILLA	1000	454.88	0.00	3.16	21.46	0.00	13.22	492.72	24.64

Table C.5: Variable Selection Performance for Weak Main Effects Setting.

Method	**# Covariates**	**# Correct Selected**	**# Wrong Selected**	**# Correct Not Se**
SKIM-FA	250	5	6	0
MARS	250	5	75	0
SPAM-2Stage	250	4	77	1
Pairs Lasso	250	5	123	0
HierLasso	250	5	160	0
SKIM-FA	500	5	16	0
SPAM-2Stage	500	1	21	4
HierLasso	500	5	62	0
Pairs Lasso	500	5	85	0
MARS	500	2	132	3
SKIM-FA	1000	5	9	0
SPAM-2Stage	1000	1	41	4
MARS	1000	5	75	0
HierLasso	1000	5	120	0
Pairs Lasso	1000	5	144	0

Table C.6: Estimation Performance for Weak Main Effects Setting.

Method	**p**	**Correct Selected SSE (Main)**	**Correct Not Selected SSE (Main)**	**Wrong Selected SSE (Main)**	**Correct Selected SSE (Pair)**	**Correct Not Selected SSE (Pair)**	**Wrong Selected SSE (Pair)**	**Total SSE**	**Total SSE ÷ Signal Variance**
SKIM-FA	250	0.45	0.00	0.95	0.73	0.00	0.77	2.89	0.14
MARS-EMP	250	1.46	0.00	4.02	4.83	0.00	4.67	14.97	0.75
SPAM-2Stage	250	0.09	0.05	2.22	10.72	7.73	0.42	21.23	1.06
MARS-VANILLA	250	22497.35	0.00	7.31	148073.29	0.00	18.55	170596.50	8529.83
SKIM-FA	500	0.69	0.00	2.05	1.50	0.00	1.37	5.61	0.28
SPAM-2Stage	500	0.27	0.20	4.09	0.00	19.46	0.08	24.11	1.21
MARS-EMP	500	0.41	0.15	21.92	0.00	19.46	15.56	57.51	2.88
MARS-VANILLA	500	0.10	0.15	323788.65	0.00	19.46	324588.33	648396.70	32419.83
SKIM-FA	1000	0.72	0.00	1.37	0.61	0.00	0.63	3.33	0.17
MARS-EMP	1000	0.67	0.00	5.86	3.37	0.00	5.63	15.52	0.78
SPAM-2Stage	1000	0.16	0.20	6.69	0.00	18.33	0.31	25.69	1.28
MARS-VANILLA	1000	23.62	0.00	3.18	23.16	0.00	15.43	65.39	3.27

Table C.7: Proxy Ground Truth Effects and Signal Variances for the Bike Sharing Dataset.

Effect	**Signal Variance**
Hour	0.382
Air Temp.	0.104
Humidity	0.024
Windspeed	0.002
Hour x Air Temp.	0.047
Hour x Humidity	0.01
Hour x Windspeed	0.002
Air Temp. x Humidity	0.012
Air Temp. x Windspeed	0.005
Humidity x Windspeed	0.003

Table C.8: Variable Selection Performance for the Bike Sharing Dataset.

Method	**# Covariates**	**# Original Selected**	**# Wrong Selected**
SKIM-FA	250	2	0
HierLasso	250	3	7
Pairs Lasso	250	3	29
MARS	250	3	96
SPAM-2Stage	250	4	97
SKIM-FA	500	2	0
HierLasso	500	3	8
SPAM-2Stage	500	3	22
Pairs Lasso	500	3	39
MARS	500	4	109
SKIM-FA	1000	3	0
HierLasso	1000	3	5
SPAM-2Stage	1000	3	8
Pairs Lasso	1000	3	76
MARS	1000	3	119

Table C.9: Estimation Performance for the Bike Sharing Dataset.

Method	**# Noise**	**Correct Selected SSE (Main)**	**Correct Not Selected SSE (Main)**	**Wrong Selected SSE (Main)**	**Correct Selected SSE (Pair)**	**Correct Not Selected SSE (Pair)**	**Wrong Selected SSE (Pair)**	**Total SSE**
SKIM-FA	250	0.15	0.027	0	0.019	0.038	0	0.233
SPAM-2Stage	250	0.149	0	0.172	0.091	0	0.01	0.422
MARS-EMP	250	0.209	0.002	0.476	0.052	0.026	0.344	1.11
MARS-Vanilla	250	6.522	0.002	1.644	1.036	0.026	2.2	11.431
SKIM-FA	500	0.148	0.027	0	0.019	0.038	0	0.231
SPAM-2Stage	500	0.15	0.002	0.057	0.081	0.009	0.002	0.302
MARS-EMP	500	0.225	0	0.529	0.052	0.026	0.3	1.131
MARS-Vanilla	500	5.564	0	0.5	1.037	0.026	2.085	9.212
SKIM-FA	1000	0.145	0.002	0	0.107	0.009	0	0.263
SPAM-2Stage	1000	0.149	0.002	0.027	0.081	0.009	0.000	0.269
MARS-EMP	1000	0.214	0.002	0.485	0.054	0.026	0.245	1.026
MARS-Vanilla	1000	6.556	0.002	0.796	0.947	0.026	1.882	10.209

Appendix D

Appendix for "The DeCAMFounder: Non-Linear Causal Discovery in the Presence of Hidden Variables"

D.1 Proofs

D.1.1 Proof of Proposition 5.2.1

Proof. Let $O = \{x_1, \cdots, x_p\}$ and $\mathbb{P}(x, h)$ be Markov with respect to a DAG G. Then, there exist functions g_j and f_j such that

$$\begin{aligned} h_i &= g_i(\mathrm{Pa}_G(h_i), h_i') \quad \forall i \in [K] \\ x_j &= f_j(\mathrm{Pa}_G(x_i), \epsilon_j) \quad \forall j \in [p], \end{aligned} \tag{D.1}$$

where the noises $h' \perp\!\!\!\perp \epsilon$. We prove the claim by inducting on the number of confounders K. For $K = 1$, $\mathrm{Pa}_G(h_1) \subset O$. Hence, h_1 is only a function of h_1' and the observed nodes. Consider the graph G' formed by removing node h_1 (and all corresponding incoming and outgoing arrows) in G and adding a new node h_1'. Let $\mathrm{Pa}_{G'}(h_1') = \emptyset$ and $\mathrm{Ch}_{G'}(h_1^{'}) = \mathrm{Ch}_G(h_1)$, where $\mathrm{Ch}_G(h_1)$ denotes the children of node h in the graph G. For every $x_i \in \mathrm{Ch}_G(h_1)$, add $\mathrm{Pa}_G(h_1)$ to the parent set of x_i in G'. Then, G' is a DAG and h' is a source. Furthermore, the partial order induced by G' equals the partial order induced by G on the subset of observed nodes $\{x_1, \cdots, x_p\}$. It suffices to show that $\mathbb{P}(x, h')$ is Markov with respect to G'. Then, for any $x_j \in \mathrm{Ch}_{G'}(h')$,

$$\begin{aligned} x_j &= f_j(\mathrm{Pa}_G(x_j) \setminus h_1, h_1, \epsilon_j) \\ &= f_j(\mathrm{Pa}_G(x_j) \setminus h_1, g_1(\mathrm{Pa}_G(h_1), h_1'), \epsilon_j). \end{aligned} \tag{D.2}$$

Hence, x_j functionally only depends on the parent set specified by G'. Since the parent sets of G' agree with G on the remaining set of observed nodes, the claim holds for $K = 1$.

Assume that for any set of p observed nodes and $K - 1$ confounders, we can always construct such a DAG G'. Suppose that there are K total confounders. Without loss of generality, suppose that $\text{Pa}_G(h_1) \subset O$ (i.e., h_1 comes before $h_2, \cdots, h_K$ in the causal ordering). Treat $x_1, \cdots, x_p, h_2, \cdots, h_K$ as the set of observed nodes. Then, h_1 is the only confounder. Hence, by the inductive hypothesis, there exists a DAG G' and exogenous h_1' such that $x_1, \cdots, x_p, h_1', h_2, \cdots, h_K$ factorizes according to G'. To complete the proof, treat $x_1, \cdots, x_p, h_1'$ as the set of observed nodes, and $h_2, \cdots, h_K$ as the set of confounders. Then, there are $K-1$ total confounders. Applying the inductive hypothesis again, there exists a DAG G'' such that $x_1, \cdots, x_p, h_1', h_2'', \cdots, h_K''$ factorizes according to G'' and $h_2'', \cdots, h_K''$ are sources. Since we picked h_1 to come before $h_2, \cdots, h_p$ in the causal ordering in G, h_1' remains a source in G''. Hence, $h_1', h_2'', \cdots, h_K''$ are all sources as desired.

□

D.1.2 Proof of Lemma 5.4.4

Proof. By Eq. (5.2),

$$x = (I - B)^{-1}\epsilon + (I - B)^{-1}\Theta h. \tag{D.3}$$

Hence,

$$\begin{aligned} x_j &= \sum_{x_i \in \text{Pa}_{G^*}(x_j)} B_{ij} x_i + \Theta_i^T h \\ &= \sum_{x_i \in \text{Pa}_{G^*}(x_j)} B_{ij}[(I-B)^{-1}\epsilon + (I-B)^{-1}\Theta h]_i + \Theta_j^T h. \end{aligned} \tag{D.4}$$

Taking expectations with respect to h gives

$$\begin{aligned} s_j &= \mathbb{E}[x_j \mid h] \\ &= \mathbb{E}\left[\sum_{i \in \text{Pa}_{G^*}(x_j)} B_{ij}[(I-B)^{-1}\epsilon + (I-B)^{-1}\Theta h]_i + \Theta_j^T h \mid h\right] \\ &= \sum_{i \in \text{Pa}_{G^*}(x_j)} \mathbb{E}\left[B_{ij}[(I-B)^{-1}\epsilon + (I-B)^{-1}\Theta h]_i \mid h\right] + \Theta_j^T h \\ &= \sum_{i \in \text{Pa}_{G^*}(x_j)} [(I-B)^{-1}\Theta h]_i + \Theta_j^T h \qquad (\text{since } \epsilon \perp\!\!\!\perp h), \end{aligned} \tag{D.5}$$

which completes the proof. □

D.1.3 Proof of Theorem 5.4.1

Lemma D.1.1. *For every $j \in [p]$, there exists an f_j such that*

$$x_j = \epsilon_j + d_j + f_j(\epsilon_1, d_1, \cdots, \epsilon_{j-1}, d_{j-1}) \tag{D.6}$$

for the SEM in Eq. (5.1).

Proof. The proof follows by inducting on the number of nodes in G^*. For $p = 1$, $x_1 = d_1 + \epsilon$, and Eq. (D.6) trivially holds. For $p = 2$,

$$\begin{aligned} x_2 &= d_2 + \epsilon_2 + f_{12}(x_1) \\ &= d_2 + \epsilon_2 + f_{12}(d_1 + \epsilon_1). \end{aligned} \tag{D.7}$$

The claim holds by setting $f_2(\epsilon_1, d_1) = f_{12}(d_1 + \epsilon_1)$. Suppose that Eq. (D.6) holds for all DAGs with at most $p - 1$ nodes. Then it suffices to prove that Eq. (D.6) holds for all DAGs with p nodes. For this, note that

$$\begin{aligned} x_p &= d_p + \epsilon_p + \sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(x_i) \\ &= d_p + \epsilon_p + \sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(\epsilon_i + d_i + f_j(\epsilon_1, d_1, \cdots, \epsilon_{i-1}, d_{i-1})), \end{aligned} \tag{D.8}$$

where the last line follows from the inductive hypothesis. Thus by setting

$$f_p(\epsilon_1, d_1, \cdots, \epsilon_{j-1}, d_{p-1}) = \sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(\epsilon_i + d_i + f_j(\epsilon_1, d_1, \cdots, \epsilon_{i-1}, d_{i-1})),$$

the claim follows. □

Corollary D.1.2. *For an SEM in the form of Eq.* (5.1)*, it holds that*

$$\mathbb{E}[x_j \mid h] = \mathbb{E}[x_j \mid d_1, \cdots, d_j].$$

We prove Theorem 5.4.1 below using Corollary D.1.2.

Proof. By Eq. (5.1), it suffices to show that there exists an r_j such that $d_j = s_j - r_j(s_1, \cdots, s_{j-1})$ for every $j \in [p]$. We prove this claim by inducting on the number of nodes in G^*. For $p = 1$, $x_1 = d_1 + \epsilon_1$. Since $h \perp\!\!\!\perp \epsilon$, $s_1 = d_1$, and the claim holds by setting $r_1 = 0$. For $p = 2$, $x_2 = \epsilon_2 + d_2 + f_{12}(x_1)$, where f_{12} may equal 0 if x_1 is not a parent of x_2. Then,

$$\begin{aligned} s_2 &= \mathbb{E}[x_2 \mid h] \\ &= \mathbb{E}[x_2 \mid d_1, d_2] \quad \text{(by Corollary D.1.2)} \\ &= \mathbb{E}[d_2 + \epsilon_2 + f_{12}(x_1) \mid d_1, d_2] \\ &= d_2 + \mathbb{E}[f_{12}(x_1) \mid d_1] \\ &= d_2 + \mathbb{E}[f_{12}(x_1) \mid s_1] \quad \text{(since } s_1 = d_1\text{)}. \end{aligned} \tag{D.9}$$

Hence, $d_2 = s_2 - \mathbb{E}[f_{12}(x_1) \mid s_1]$. The claim holds by setting $r_2(s_1) = \mathbb{E}[f_{12}(x_1) \mid s_1]$. Suppose that $d_j = s_j - r_j(s_1, \cdots, s_{j-1})$ for all SEMs in the form of Eq. (5.1) with at most $p - 1$ nodes. It suffices to show that that there exists an r_p such that $d_p = s_p - r_p(s_1, \cdots, s_{p-1})$ for an arbitrary SEM in the form of Eq. (5.1) with p nodes.

For this, consider the subgraph formed from $x_1, \cdots, x_{p-1}$. Since x_p is a sink node, $\mathbb{P}(x_1, \cdots, x_{p-1})$ factorizes according to a DAG. Hence, by the inductive hypothesis, there exists $\{r_j\}_{j=1}^{p-1}$ such that

$$d_j = s_j - r_j(s_1, \cdots, s_{j-1}) \quad \forall j = 1, \cdots, p-1. \tag{D.10}$$

Now, note that

$$\begin{aligned}
s_p &= \mathbb{E}[x_p \mid h] \\
&= \mathbb{E}[x_p \mid d_1, d_2, \cdots, d_p] \quad \text{(by Corollary D.1.2)} \\
&= \mathbb{E}[d_p + \epsilon_p + \sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(x_i) \mid d_1, d_2, \cdots, d_p] \\
&= d_p + \mathbb{E}[\sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(x_i) \mid d_1, d_2, \cdots, d_{p-1}] \\
&= d_p + \mathbb{E}[\sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(x_i) \mid \{s_i - r_i(s_1, \cdots, s_{i-1})\}_{i=1}^{p-1}] \quad \text{(by Eq. (D.10))}.
\end{aligned} \tag{D.11}$$

Thus by setting

$$r_p(s_1, \cdots, s_{p-1}) = \mathbb{E}[\sum_{x_i \in \mathrm{Pa}_{G^*}(x_p)} f_{ij}(x_i) \mid \{s_i - r_i(s_1, \cdots, s_{i-1})\}_{i=1}^{p-1}], \tag{D.12}$$

the result follows. $\square$

D.1.4 Proof of Proposition 5.4.7

We follow Friedman and Nachman [2000] to compute the marginal likelihood. By Theorem 5.4.1, the marginal likelihood decomposes as

$$\begin{aligned}
\mathbb{P}(X \mid G, S) &= \int \mathbb{P}(X \mid G, \Omega_G) d\mathbb{P}(\Omega_G) \\
&= \int \prod_{j=1}^{p} \mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{C_j}, \Omega_G) d\mathbb{P}(\Omega_G) \\
&= \int \prod_{j=1}^{p} \mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{C_j}, \{f_{ij}\}_{i \in \mathrm{Pa}_G(x_j)}, r_j) d\mathbb{P}(\{f_{ij}\}_{i \in \mathrm{Pa}_G(x_j)}, r_j) \\
&= \prod_{j=1}^{p} \int \mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{C_j}, \{f_{ij}\}_{i \in \mathrm{Pa}_G(x_j)}, r_j) d\mathbb{P}(\{f_{ij}\}_{i \in \mathrm{Pa}_G(x_j)}, r_j) \\
&= \prod_{j=1}^{p} \mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{C_j}).
\end{aligned}$$

Hence,

$$\log \mathbb{P}(X \mid G, S) = \sum_{j=1}^{p} \log \mathbb{P}(X_j - S_j \mid X_{\mathrm{Pa}_G(x_j)}, S_{C_j}).$$

The proof now follows from Equation 2.30 of Rasmussen and Williams [2006].

D.2 Score Fuction Details

In our experiments, we used an RBF kernel for the k_{η_j} and $k_{\theta_{ij}}$. The kernel hyperparameters η_j and θ_{ij} refer to the unknown lengthscales in an RBF kernel. We implemented this model using the Gaussian process package `GPyTorch` [Gardner et al., 2018] to fit the kernel hyperparameters (i.e., by maximizing the log marginal likelihood via gradient ascent). We used a total of 100 iterations using the `Adam` optimizer with a learning rate of 0.01. See the `scores.py` file in the "decamfound" folder on the Github repository for our `python` code.

D.3 Generating Simulated Data

For node x_j with sampled trend types f_{ij} and g_{kj} and weights θ_{ij} and θ'_{kj}, let

$$O_j = \sum_{i \in \mathrm{Pa}_{G^*}(x_j)} \theta_{ij} f_{ij}(x_i) \qquad C_j = \sum_{k=1}^{K} \theta'_{kj} g_{kj}(h_k)$$

represent the (unnormalized) variation explained by the observed and confounder nodes, respectively. We wish to find normalization constants $c_{\text{par},j}$ and $c_{\text{confound},j}$ such that

$$\mathrm{Cov}(c_{\text{par},j} O_j + c_{\text{confound},j} C_j) = 1 - \sigma^2_{\text{noise}}$$

and

$$\mathrm{Cov}(c_{\text{confound},j} C_j) = \sigma^2_{\text{confound}}.$$

Since it may be difficult to analytically solve for these normalization constants, we find them inductively using a Monte Carlo approach. In particular, suppose we have computed the normalization constants $c_{\text{par},i}$ and $c_{\text{confound},j}$ for $i < j$. Then, we take many Monte Carlo samples (we use 10,000 in the experiments) from the marginal distribution over $h, x_1, x_2, \ldots, x_{j-1}$. This allows us to estimate $\mathrm{Cov}(C_j)$ and solve for $c_{\text{confound},j}$. Similarly, we may use these samples to estimate $\mathrm{Cov}(O_j, C_j)$, which along with the value of $c_{\text{confound},j}$ and the estimate of $\mathrm{Var}(C_j)$, allows us to solve for $c_{\text{par},j}$ in the first equation. There are several edge cases depending on if x_j has no observed and/or confounder parents:

1. If x_j has no observed or confounder parents (i.e., is a source), then set $c_{\text{par},j} = 0$ and $c_{\text{confound},j} = 0$. Set the noise variance for x_j equal to 1.

2. If x_j has no confounder parents but at least one observed node parent, then set the signal variance for x_j equal to $\sigma^2_{\text{signal}} + \sigma^2_{\text{confound}}$.

3. If x_j has at least one confounder parents but no observed node parents, then set the noise variance for x_j equal to $\sigma^2_{\text{noise}} + \sigma^2_{\text{signal}}$.

See the `generate_synthetic_data.py` file in our Github repository for the code.

D.4 Additional Figures and Experiments

D.4.1 Synthetic Data Experiments

In Section 6.5, we reported on the proportion of times the incorrect parent set was selected over the true parent set. We report on the following additional metric to understand how confidently wrong (or correct) each method is:

1. **Log Odds (Wrong vs. True)**: each score equals the log marginal likelihood (technically an approximation for BIC) of the parent set. Assuming a uniform prior over the set of all parent sets, the log odds (LO) between the wrong and true parent set reduces into the difference between scores:

$$\begin{aligned}\mathrm{LO}_i &= \log \frac{\mathbb{P}(X \mid P_i)\mathbb{P}(P_\mathrm{i})}{\mathbb{P}(X \mid P_\mathrm{correct})\mathbb{P}(P_\mathrm{correct})} \\ &= \mathrm{score}(P_i) - \mathrm{score}(P_\mathrm{correct}).\end{aligned} \tag{D.13}$$

   We report $\max_{i \in [M]} \mathrm{LO}_i$ for the Wrong Parent Addition task in Fig. D-1 and for the Correct Parent Deletion task in Fig. D-2. A higher value is worse (i.e., it indicates that a method places higher confidence in an incorrect parent set than the true parent set).

Scoring Candidate DAG Results. We score a candidate set of $M = 100$ incorrect DAGs $\mathcal{G}$, built from randomly adding or deleting edges multiple times, starting from the true DAG (i.e., the natural extension of our two parent set evaluation tasks). Since scoring a single graph takes $O(pN^3)$ for the non-linear methods, we consider fewer settings (i.e., fix the confounding variance to be equal to the signal variance), and only do 10 total simulations instead of 25. These results are shown in Fig. D-3. We report on two metrics:

1. **Avg. Posterior SHD:** equals $\sum_{m=1}^{M} \mathrm{SHD}(G_m, G^*)\mathbb{P}(G \mid X)$, where SHD denotes the structural hamming distance to the true graph, and $\mathbb{P}(G \mid X)$ equals the posterior probability of a graph computed from renormalizing the log marginal likelihood scores. Lower is better.

2. **SHD Between MAP and True DAG:** reports the SHD from the true DAG for the highest scoring graph (i.e., the maximum a posteriori estimate). Lower is better.

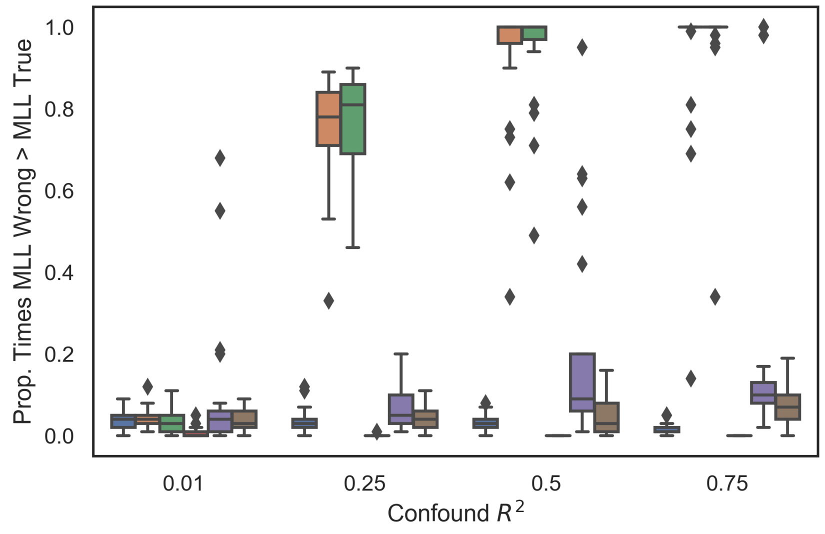
Fig. D-3 shows that linear methods, even if they account for confounding, suffer in the non-linear setting. This advantage of modeling non-linearities agrees with the results, for example, in Bühlmann et al. [2014]. Fig. D-3 also shows that, as in the Correct Parent Deletion Task, LRPS suffers even in the linear setting because the induced undirected graph is very sparse (and hence often favors deleting true edges). FInally, Fig. D-3 shows that both CAM-OBS (trivially) and our method are robust to both non-linearities and confounding.

D.4.2 Ovarian Cancer Dataset

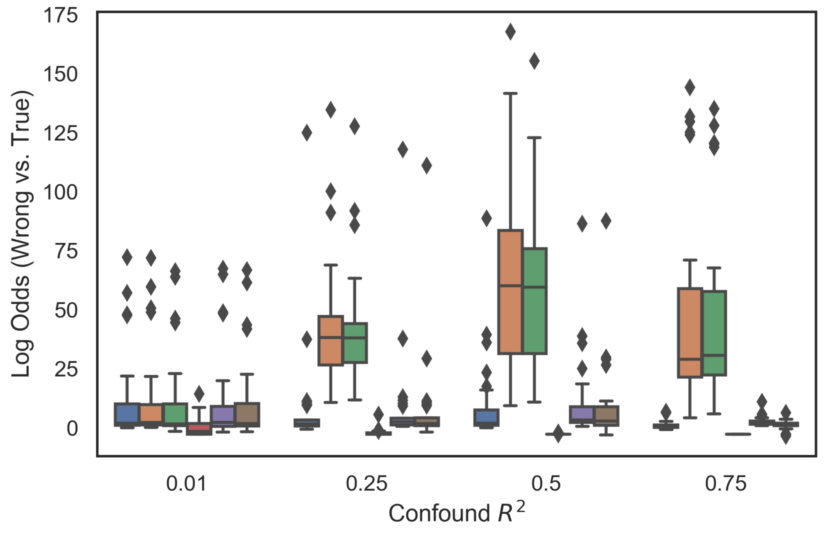
Scree Plot. Analyzing the spectrum of the data matrix consisting of the 486 observed genes in Fig. D-4 shows that there are about 7 spiked eigenvalues.

Pervasive TF Gene Correlations. 7/15 TFs have edges with more than 75 genes according to NetBox (i.e., these TFs might play the role of pervasive confounders). We summarize the absolute value of the correlations between these 7 TFs and the 486 genes in Fig. D-5.

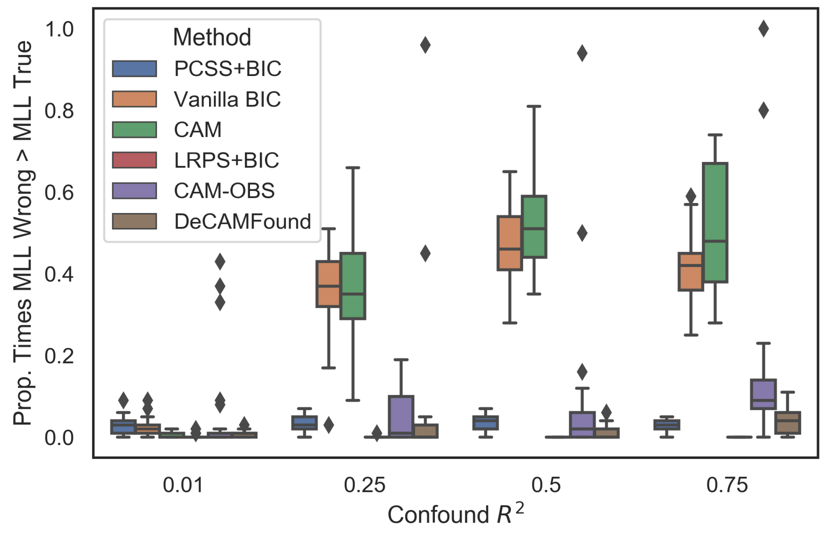
Removing the Effect of a Latent TF via PCSS. See Fig. D-7.



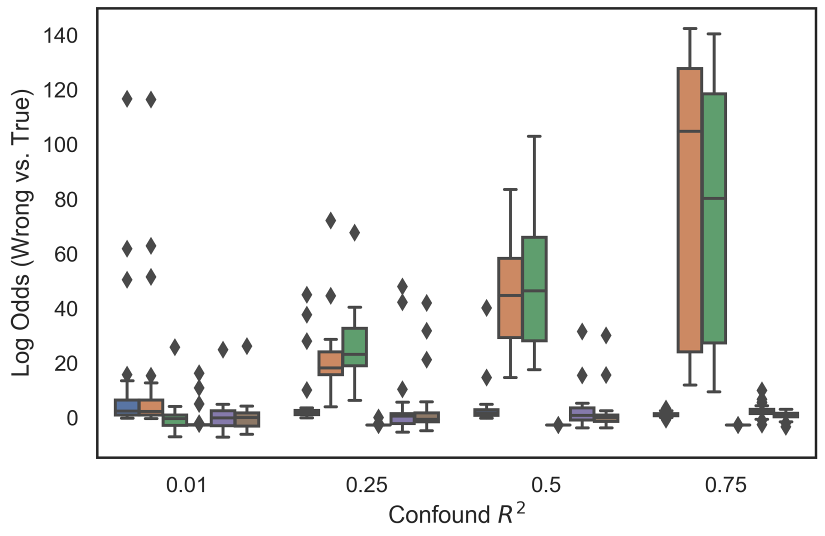
(a) Linear SEM



(b) Linear SEM

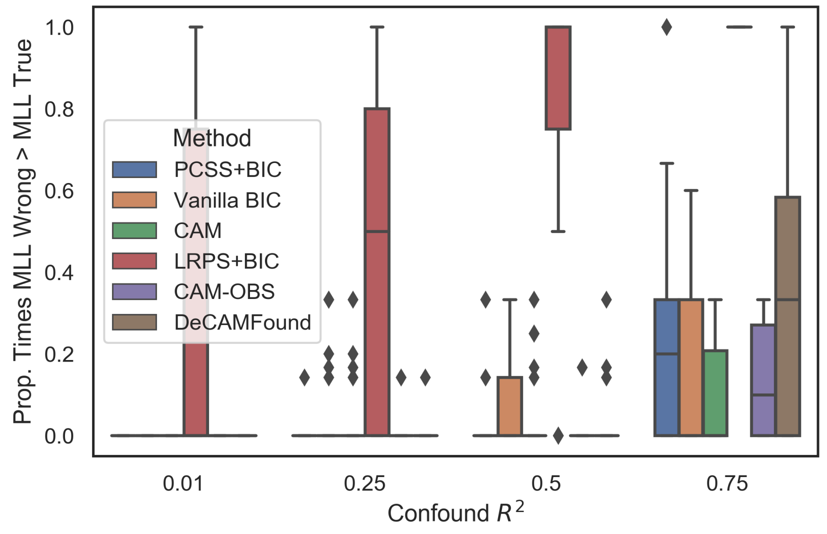


(c) Non-Linear SEM

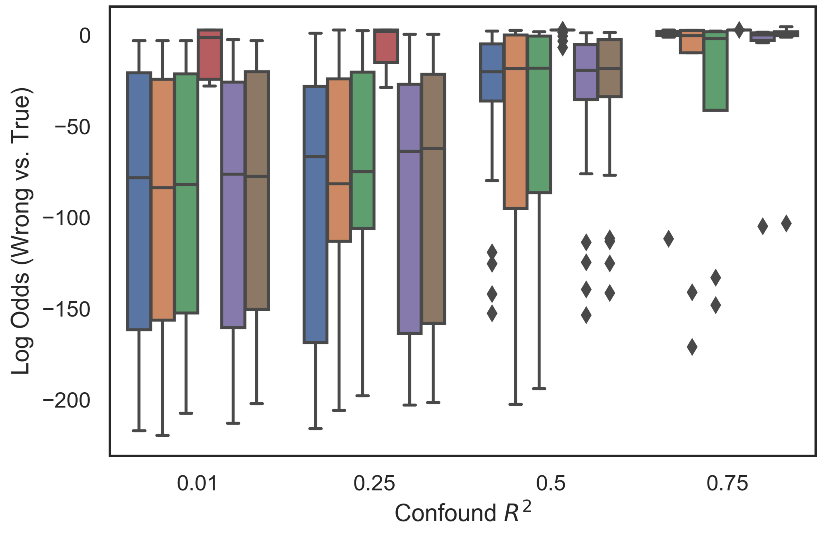


(d) Non-Linear SEM

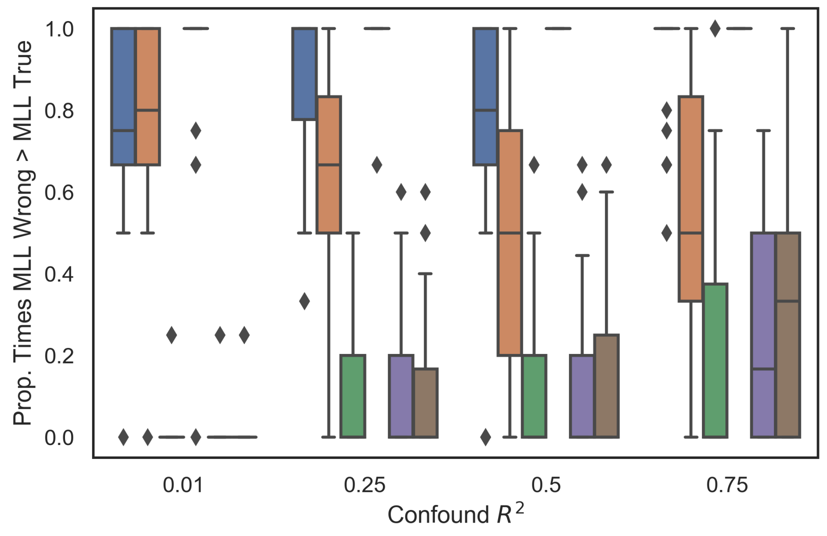
Figure D-1: Results for the Wrong Parent Addition task. 25 total simulations per dataset configuration were performed. See Section 5.5.1.2 and Appendix D.4.1 for a description of the performance metrics.



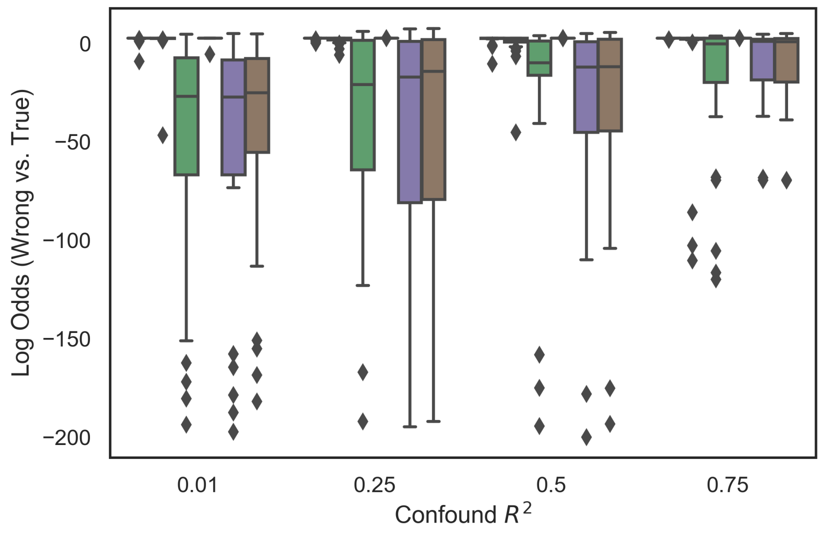
(a) Linear SEM



(b) Linear SEM

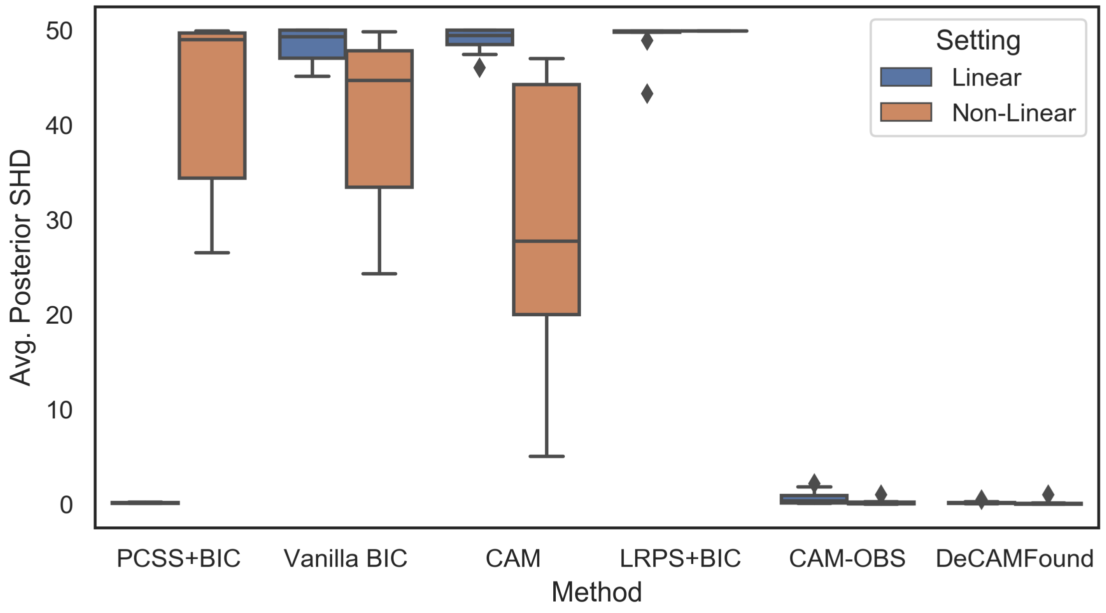


(c) Non-Linear SEM

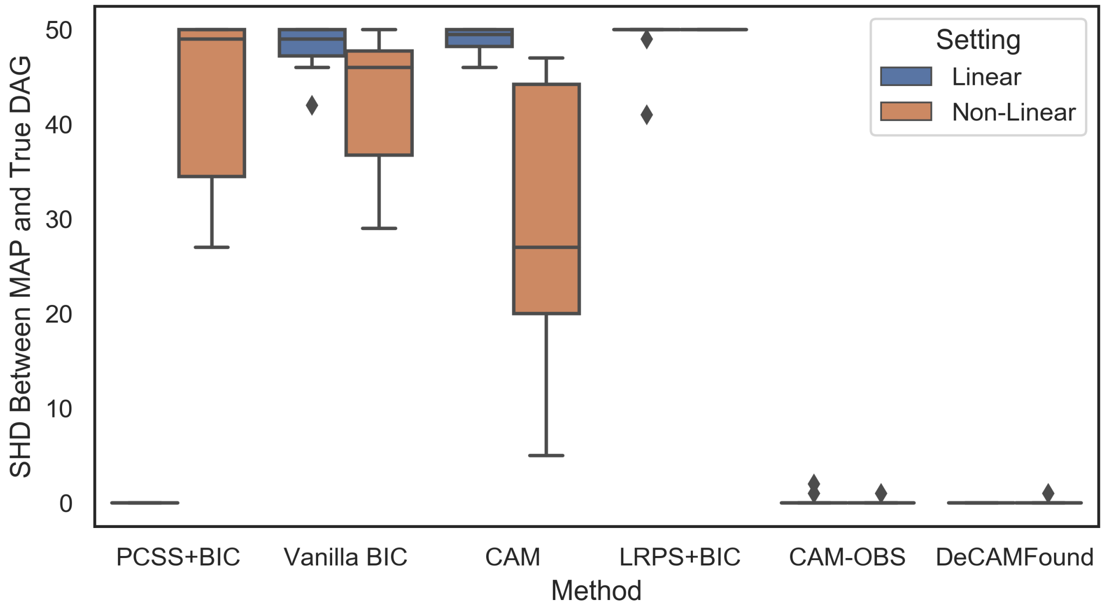


(d) Non-Linear SEM

Figure D-2: Results for the Correct Parent Deletion task. 25 total simulations per dataset configuration were performed. See Section 5.5.1.2 and Appendix D.4.1 for a description of the performance metrics.



(a) SHD



(b) Average SHD

Figure D-3: Results for the candidate DAG scoring task. 10 total simulations per dataset configuration (i.e., linear / non-linear) were performed.

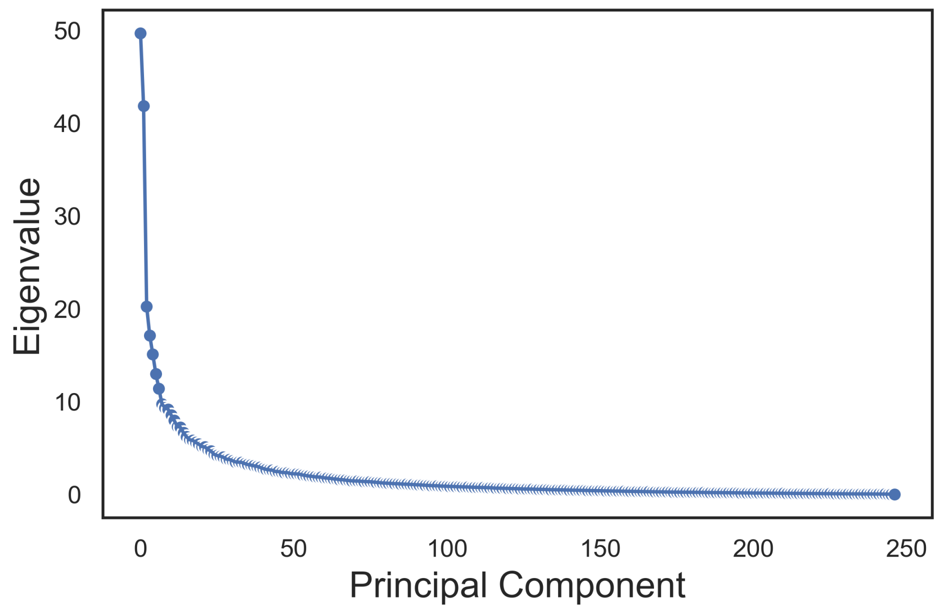


Figure D-4: PCA scree plot when the input data matrix consists of the 486 observed genes. Based on this scree plot, we select $K = 7$ components for the spectral methods.

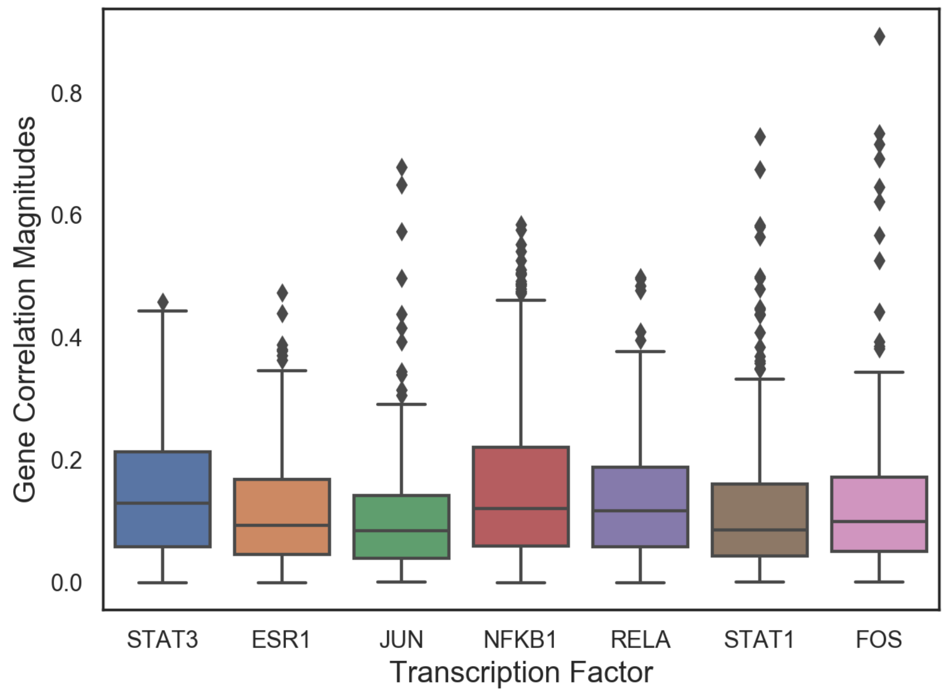
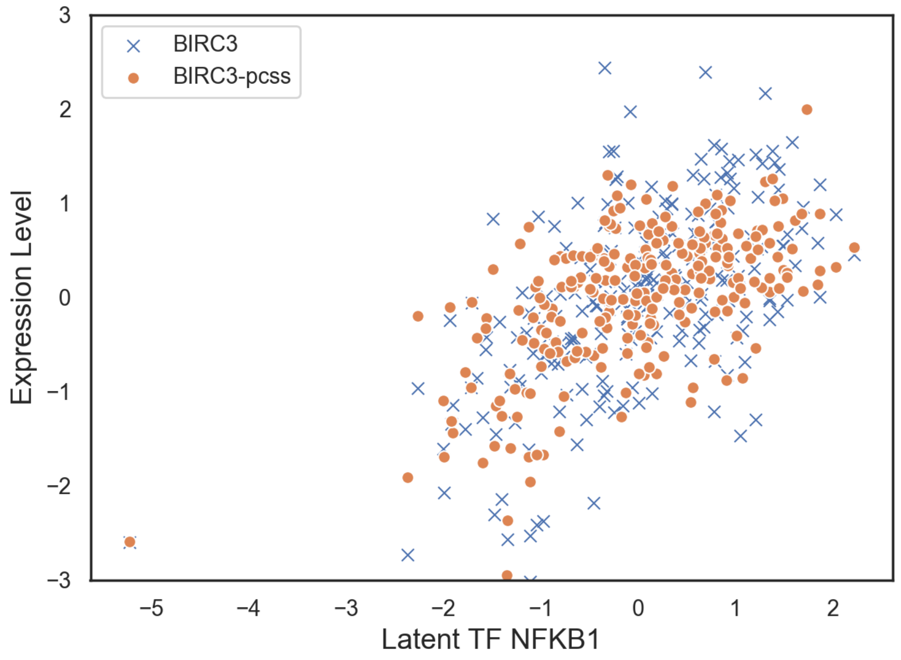


Figure D-5: Absolute value of correlations between TFs and the 486 observed genes. Only TFs with at least 75 edges are shown.



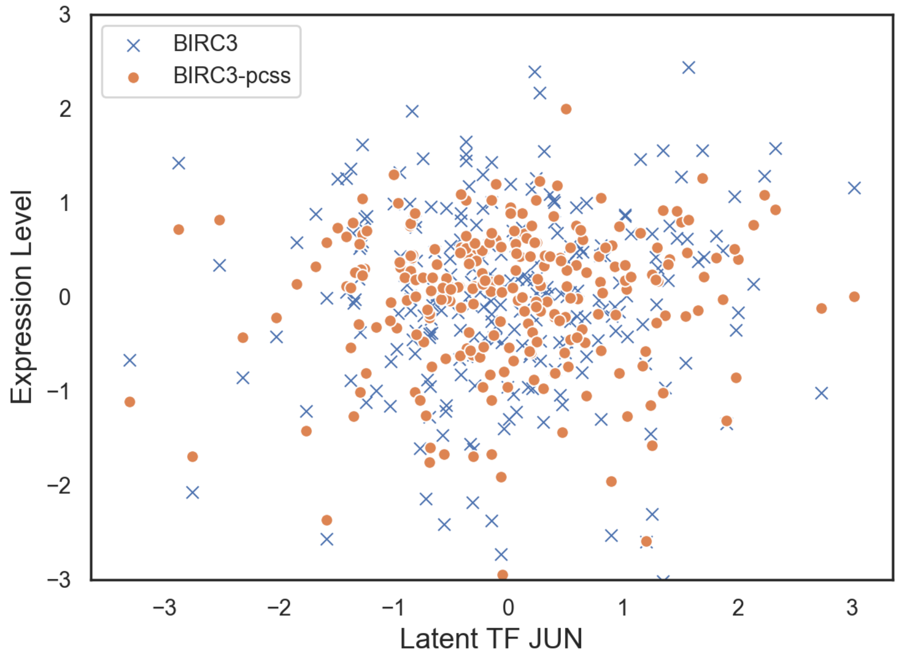


Figure D-6: Out of the 15 latent TFs, BIRC3 has the highest absolute correlation with NFKB1 and the smallest absolue correlation with JUN. BIRC3-pcss refers to the total latent confounding variation estimated for that gene via PCSS. BIRC3 refers to the actual observed values of the gene. BIRC3-pcss is correlated with NFKB1 (which is correlated with BIRC3) and not correlated with JUN.

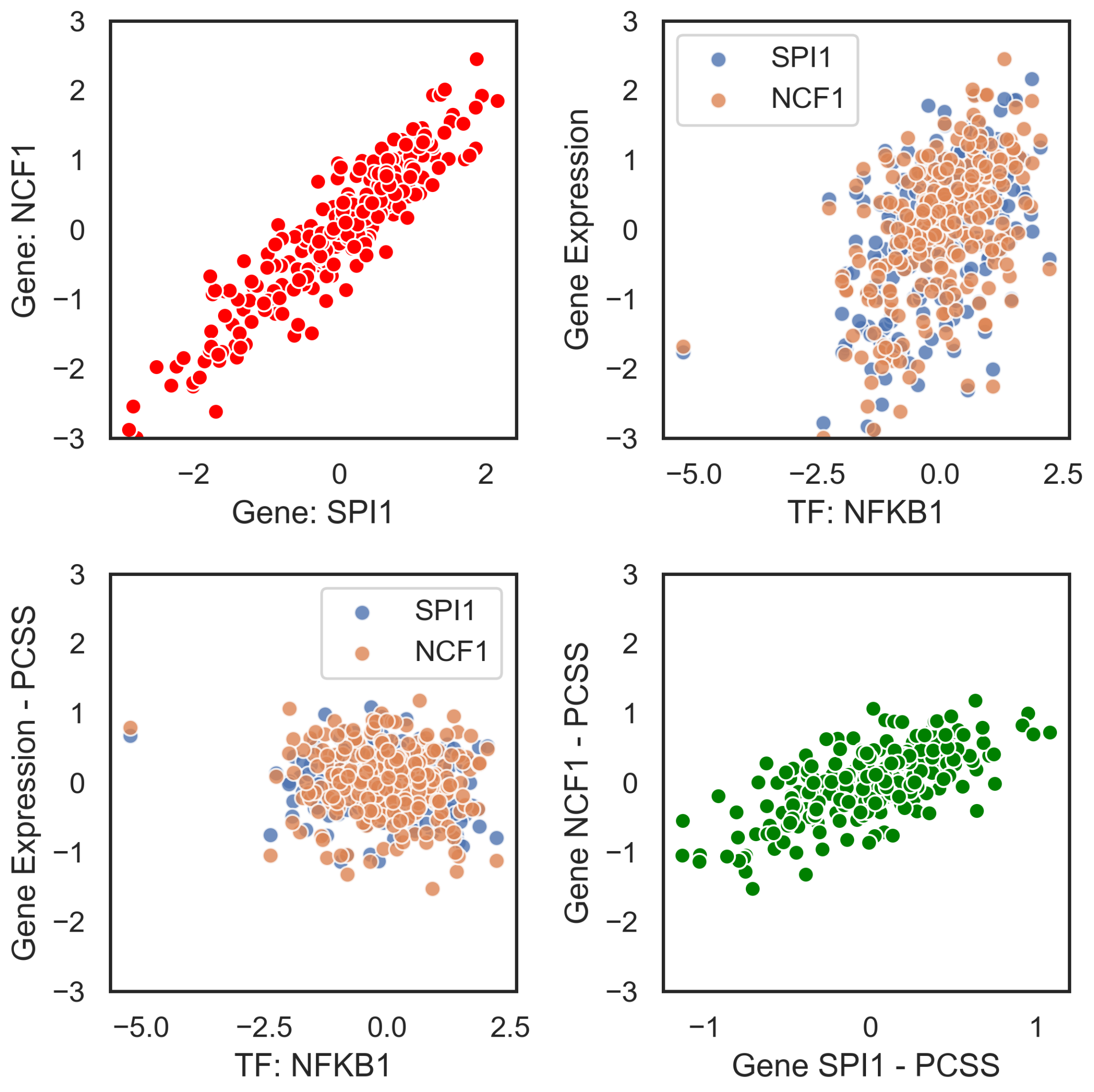


Figure D-7: Top left: scatterplot of two genes that are conditionally independent given each parents' gene neighborhood sets and TFs but dependent when removing the TFs. Top right: scatter plot of each gene for the TF that has the highest correlation with both genes. Bottom left: correlation with the transcription factor after removing the estimated confounder sufficient statistics from each gene. Bottom right: weaker correlation after removing the confounder sufficient statistics from each gene. Since both genes are still marginally dependent given the TFs without conditioning on the parent sets, the genes are still correlated in the bottom right figure.

Appendix E

Appendix for “ABCD-Strategy: Budgeted Experimental Design for Targeted Causal Structure Discovery”

E.1 Proofs

E.1.1 Proof of Proposition 6.3.2

Given infinite samples per intervention $I \in \mathcal{I}$, G^* is recovered up to its $\mathcal{I}$-Markov equivalence class. Hence, the resulting entropy after placing an infinite number of samples at each intervention is equal to $\log_2 |\mathrm{Ess}^{\mathcal{I}}(G)|$ when the true DAG is G. Since the true DAG is unknown, this entropy must be averaged over our prior distribution on $\mathcal{G}$, which is uniform. Hence, the entropy after observing an infinite number of samples per intervention in $\mathcal{I}$ equals $\frac{1}{|\mathcal{G}|}\sum_{G\in\mathcal{G}} \log_2 |\mathrm{Ess}^{\mathcal{I}}(G)|$. Minimizing this entropy over all possible interventions sets of size at most K completes the proof.

E.1.2 Proof of Theorem 6.3.4

Let

$$\mathcal{I}^\infty := \{I \in \mathcal{I}^* : \sum_{b=1}^{\infty} |\tilde{I} \in \xi_b : \tilde{I} = I| = \infty \ \mu^* a.s.\},$$

where ξ_b denotes the interventions selected at batch b by $U^f_{\mathrm{M.I}}$. Since $\mathcal{I}^*$ is finite, $\mathcal{I}^\infty$ is non-empty. When $|\mathcal{I}^\infty| > 1$, $\mathcal{I}^\infty$ is a conservative family of targets since $\mathcal{I}^*$ is a family of single-node interventions. Hence, we identify the $\mathcal{I}^\infty$-MEC of G^* in the limit of an infinite number of batches and samples [Hauser and Bühlmann, 2012]. Assume $|\mathcal{I}^\infty| > 1$. If $f(G)$ is identifiable in $\mathrm{Ess}^{\mathcal{I}^\infty}(G^*)$, then

$$\mathbb{P}\left(f(G) \mid D_B\right) \xrightarrow{\mu^* \text{ a.s.}} \mathbb{1}(f(G) = f(G^*)).$$

Hence, it suffices to show that the interventions $U^f_{\mathrm{M.I}}$ selects infinitely often identifies $f(G)$ in the limiting interventional essential graph $\mathrm{Ess}^{\mathcal{I}^\infty}(G^*)$. Suppose towards a contradiction that $f(G)$ were not fully identifiable in $\mathrm{Ess}^{\mathcal{I}^\infty}(G^*)$. By definition of almost sure convergence, there exists some $b^* < \infty$ such that any $\tilde{I} \in \mathcal{I}^* \setminus \mathcal{I}^\infty$ is never selected again after batch b^* with probability one since $\mathcal{I}^*$ is finite. Maximizing $U^f_{\text{M.I.}}$ is equivalent to minimizing the conditional entropy,

$$H^b_\xi(f \mid Y_\xi) := \mathbb{E}_{y \sim \mathbb{P}(y|D_b,\xi)}\ H(f \mid D_b, Y = y). \tag{E.1}$$

If $b > b^*$, then

$$\operatorname{argmin} \xi \in \mathbb{Z}^{\mathcal{I}^*} \cap C_b\ H^b_\xi(f \mid Y_\xi) = \operatorname{argmin} \xi \in \mathbb{Z}^{\mathcal{I}^\infty} \cap C_b\ H^b_\xi(f \mid Y_\xi) \tag{E.2}$$

since any batch b after b^* never selects an intervention in $\tilde{I} \in \mathcal{I}^* \setminus \mathcal{I}^\infty$. Since f is not identifiable in $\mathrm{Ess}^{\mathcal{I}^\infty}(G^*)$, that implies

$$\lim_{b\to\infty} H^b_{\xi_\infty}(f \mid Y_{\xi_\infty}) \to L > 0.$$

Since $\mathcal{I}^*$ consists of all single-node interventions, $\mathcal{I}^*$ can identify $f(G)$ [Hauser and Bühlmann, 2012]. Hence, there must be some $\tilde{I} \in \mathcal{I}^* \setminus \mathcal{I}^\infty$ and $\epsilon > 0$ such that

$$\lim_{b\to\infty} H^b_{\xi_\infty \cup \tilde{I}_\infty}(f) < L - \epsilon, \tag{E.3}$$

where $\tilde{I}_\infty$ denotes selecting $\tilde{I}$ infinitely many times. But Eq. (E.3) implies that there must exist some batch $b > b^*$ such that the conditional entropy of the design $\tilde{\xi} = \{\tilde{I}\}$ is uniformly smaller than the conditional entropy of any $\xi \in \mathbb{Z}^{\mathcal{I}^\infty}$. But this is a contradiction because then $\tilde{I}$ would be selected again after some batch $b > b^*$ and Eq. (E.2) would no longer hold.

For $|\mathcal{I}^\infty| = 1$, we no longer have a conservative family of targets. However, a nearly identical argument works by noting that, in the limit, we learn the observational equivalence class of the $\mathcal{I}^\infty$ mutilated graph of G^*.

E.1.3 Consistency Counterexample

Suppose we know the Markov equivalence class of G^* and the goal is to fully recover G^*. Suppose $C_b = \{\xi : \|\xi\|_0 = K\}$, where $\|\cdot\|_0$ counts the number of unique interventions in ξ. Since there is no constraint on the number of samples, only on the number of unique interventions, we may allocate an infinite number of samples per intervention within each batch. This constraint is equivalent to the one examined in Ghassami et al. [2018]. The scores in both Ness et al. [2018] and Ghassami et al. [2018] select interventions by maximizing the expected number of oriented edges in the interventional Markov equivalence classes. In particular, the utility function in Ness et al. [2018] is equivalent to maximizing,

$$U(\mathcal{I}; D) = \sum_{G\in\mathcal{G}} A(\mathrm{Ess}^{\mathcal{I}}(G))\mathbb{P}(G), \tag{E.4}$$

where $A(\mathrm{Ess}^{\mathcal{I}}(G))$ equals the additional number of edges oriented relative to the observational Markov equivalence class. Suppose G^* equals the graph in Fig. E-1

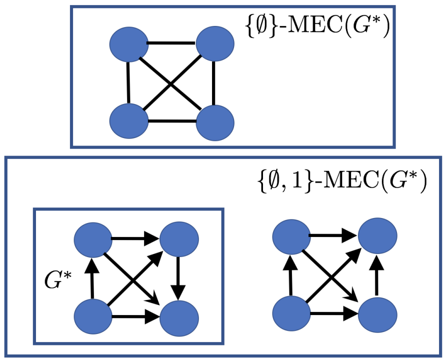


Figure E-1: Each box represents the members of the interventional Markov equivalence classes. For G^* given in the bottom left box, the observational Markov equivalence class has no edges oriented. The top box represents the essential graph of the observational Markov equivalence class. The interventional Markov equivalence class for an intervention at node one consists of two DAGs given in the bottom box.

and that $K = 1$ unique interventions are allowed within each batch. Assume that $\mathcal{I}^* = \{\{1\}, \cdots, \{4\}\}$ and that we start with a uniform prior over $\mathcal{G}$. Then, since all arrows are undirected in the observational Markov equivalence class, symmetry implies $U(\{j\}; \emptyset) = U(\{j\}; \emptyset)$ for all $i, j \in 1, \cdots, 4$. Without any loss of generality suppose intervention one is selected in batch one. We show that every subsequent batch will select intervention $\{1\}$. If only $\{1\}$ were selected, $U(\mathcal{I}; D)$ would not be consistent since the $\{\emptyset, \{1\}\}$-MEC(G^*) contains two graphs, as shown at the bottom of Fig. E-1. After batch one, the posterior is supported on these two graphs since an infinite number of samples are allocated to the intervention at node one.

The utility function in Eq. (E.4) scores interventions relative to the observational equivalence class, which causes the consistency issue. In particular, the posterior in batch two is only supported on the two DAGs given in the bottom box of Fig. E-1. The score of $\{1\}$ equals 5 in batch two while the scores of interventions $\{2\}, \{3\}, \{4\}$ equal $4, 3, 4$, respectively. Hence, in batch two, intervention $\{1\}$ will be selected again, but the posterior will remain the same since the $\{\emptyset, \{1\}\}$ interventional Markov equivalence class of G^* is already known.

An easy way to fix Eq. (E.4) (for this given counterexample) would be to only select interventions not selected in previous batches. This modification would fix the issue with the counterexample, namely prevent intervention one from being selecting infinitely often. However, when one can only allocate a finite number of samples per batch, this modification would not lead to a consistent estimator. In particular, if a certain intervention is done in some batch, and that intervention must be conducted in order to identify f, then only placing finitely many samples to that intervention in that batch and never placing any more samples in subsequent batches will not lead to a consistent method.

E.1.4 Proof of Theorem 6.4.1

Definition E.1.1. [Soma and Yoshida, 2016] Let E be a finite set. A function $f : \mathbb{Z}^E \to \mathbb{R}$ is *diminishing returns submodular* (DR-submodular) if for $x \leq y$

$$f(x + \chi_e) - f(x) \geq f(y + \chi_e) - f(y),\ x, y \in \mathbb{Z}^E \tag{E.5}$$

where $e \in E$ and χ_e is the ith unit vector.

Lemma E.1.2. *$\tilde{U}^f_{M.I.}(\xi; D)$ is DR-submodular.*

Proof. $f(G) = G$ so we omit f in $\tilde{U}^f_{\text{M.I.}}$ to simplify notation. Since the sum of submodular functions is submodular, it suffices to show

$$\begin{aligned}\mathbb{E}_{y|G,\hat{\theta}^G_{\text{MLE}},\xi}\ \tilde{U}_{\text{M.I.}}(y, \xi; D) &= H(G) - H(G \mid Y_\xi) \\ &= I((G, \hat{\theta}^G_{\text{MLE}}), Y_\xi)\end{aligned} \tag{E.6}$$

is DR-submodular, where I is the mutual information. Consider an $A \subseteq B \in \mathbb{Z}^{\mathcal{I}^*}$. Take any $C \in \mathcal{I}^*$. Since entropy decreases with more conditioning,

$$\begin{aligned}H(Y_C \mid Y_A) - H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}})) \geq \\ H(Y_C \mid Y_B) - H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}})).\end{aligned} \tag{E.7}$$

By conditional independence,

$$\begin{aligned}H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}})) &= H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}}), Y_A) \\ &= H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}}), Y_B).\end{aligned} \tag{E.8}$$

Hence, Eq. (E.7) may be rewritten as,

$$\begin{aligned}I((G, \hat{\theta}^G_{\text{MLE}}), Y_C \mid Y_A) = \\ H(Y_C \mid Y_A) - H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}}), Y_A) \geq \\ H(Y_C \mid Y_B) - H(Y_C \mid (G, \hat{\theta}^G_{\text{MLE}}), Y_B) = \\ I((G, \hat{\theta}^G_{\text{MLE}}), Y_C \mid Y_B).\end{aligned} \tag{E.9}$$

Eq. (E.9) implies

$$\begin{aligned}&I((G, \hat{\theta}^G_{\text{MLE}}), Y_A + Y_C) - I((G, \hat{\theta}^G_{\text{MLE}}), Y_A) \\ &\geq I((G, \hat{\theta}^G_{\text{MLE}}), Y_B + Y_C) - I((G, \hat{\theta}^G_{\text{MLE}}), Y_B)\end{aligned} \tag{E.10}$$

as desired. □

The proof of Theorem 6.4.1 then follows directly from Lemma E.1.2 and Soma et al. [2014, Theorem 2.4].

E.1.5 Proof of Proposition 6.4.2

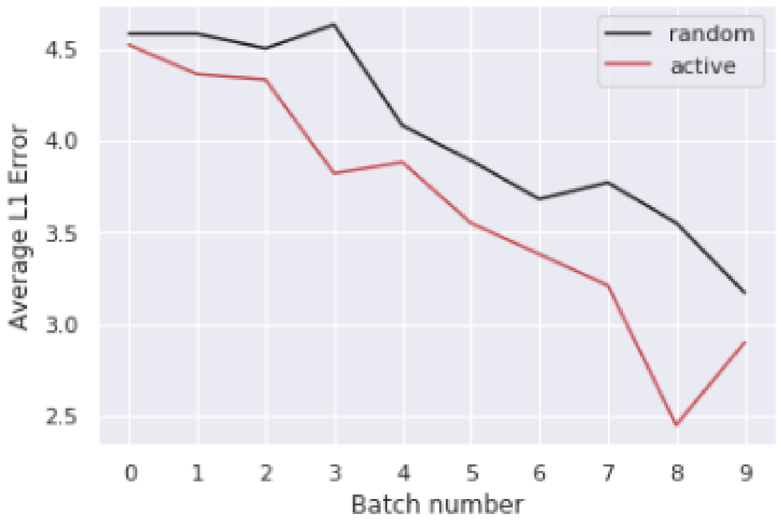
For each graph $G \in \mathcal{G}_T$, compute the associated edge weights $\hat{\theta}^G_{\text{MLE}}$. Computing each $\hat{\theta}^G_{\text{MLE}}$ takes $O(p\kappa^3)$ time using the formula given in Hauser and Bühlmann [2012, pg. 17]. Since there are T DAGs, the total time to compute the MLE estimates of the edge weights of each DAG is $O(Tp\kappa^3)$. Sampling from a multivariate Gaussian with bounded indegree with known adjacency matrix takes $O(p\kappa)$ time. $\hat{U}^f_{\text{M.I.}}$ requires a total of $|\mathcal{I}^*|MN_bT^2$ samples. Hence, the total computation time of sampling all the y_{mt} in Eq. (6.14) is $O(|\mathcal{I}^*|MN_b\kappa pT^2)$. Evaluating $\hat{U}^f_{\text{M.I.}}$ takes $O(MT^2)$ time using these samples, which is of lower computational complexity than computing $\hat{U}^f_{\text{M.I.}}$. Hence, the total runtime is $O(p\kappa^3 + |\mathcal{I}^*|MN_b\kappa pT^2)$.

E.1.6 Constraint on the Number of Unique Interventions

If we are only allowed to allocate at most K unique interventions per batch, we modify Algorithm 10 by allocating $\frac{N_b}{K}$ samples per intervention in Algorithm 9. Once an intervention is selected, that intervention is removed from I^* and another one is greedily selected from the remaining set. With this strategy, Algorithm 9 will terminate after K iterations. Hence, there will be at most K unique interventions as desired.

E.1.7 DREAM4 Supplementary Figures

We applied our targeted experimental design strategy towards learning the *downstream pathways* of select genes from a 10-node network from the DREAM4 challenge. We observed a modest improvement over the random strategy for some central genes in the network (Fig. E-2, top). However, the results are subject to high variations (Fig. E-2, bottom), which we surmise to be due to the small size of the observational dataset. Nevertheless, these preliminary results illustrate the promise of applying targeted experimental design to real, large-scale biological datasets.



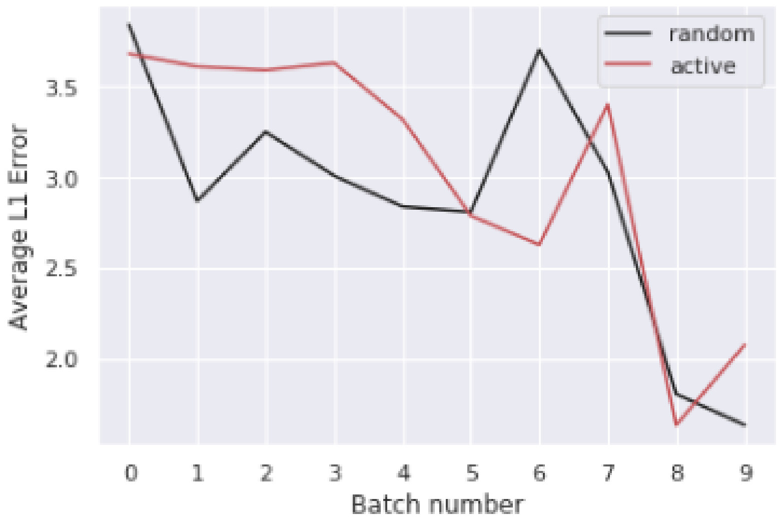


Figure E-2: Performance of intervention strategies on predicting the descendants of genes 6 (top) and 8 (bottom).

Bibliography

C. 1000 Genomes Project. A global reference for human genetic variation. *Nature*, 526:68–74, 2015.

A. Agarwal, D. Shah, D. Shen, and D. Song. On robustness of principal component regression. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

R. Agrawal and T. Broderick. High-dimensional variable selection and non-linear interaction discovery in linear time. *In Preparation*, 2021.

R. Agrawal, T. Broderick, and C. Uhler. Minimal I-MAP MCMC for scalable structure discovery in causal DAG models. In *International Conference on Machine Learning*, 2018.

R. Agrawal, T. Campbell, J. Huggins, and T. Broderick. Data-dependent compression of random features for large-scale kernel approximation. In *International Conference on Artificial Intelligence and Statistics*, 2019a.

R. Agrawal, J. Huggins, B. Trippe, and T. Broderick. The kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. In *International Conference on Machine Learning*, 2019b.

R. Agrawal, C. Squires, K. Yang, K. Shanmugam, and C. Uhler. ABCD-strategy: Budgeted experimental design for targeted causal structure discovery. In *International Conference on Artificial Intelligence and Statistics*, 2019c.

R. Agrawal, B. Trippe, J. Huggins, and T. Broderick. The kernel interaction trick: Fast Bayesian discovery of pairwise interactions in high dimensions. In *International Conference on Machine Learning*, 2019d.

R. Agrawal, U. Roy, and C. Uhler. Covariance matrix estimation under total positivity for portfolio selection. *Journal of Financial Econometrics*, 2021a.

R. Agrawal, C. Squires, N. Prasad, and C. Uhler. The DeCAMFounder: Non-linear causal discovery in the presence of hidden variables. *Under Review*, 2021b.

M. Amjad, D. Shah, and D. Shen. Robust synthetic control. *Journal of Machine Learning Research*, 19(22):1–51, 2018.

S. A. Andersson, D. Madigan, and M. D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25(2):505–541, 1997.

H. Aschard. A perspective on interaction effects in genetic association studies. *Genetic Epidemiology*, 2016.

H. Avron, V. Sindhwani, J. Yang, and M. W. Mahoney. Quasi-Monte Carlo feature maps for shift-invariant kernels. *Journal of Machine Learning Research*, pages 1–38, 2016.

H. Avron, M. Kapralov, C. Musco, C. Musco, A. Velingker, and A. Zandieh. Random Fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *International Conference on Machine Learning*, 2017.

M. Balcan, A. Blum, and S. Vempala. On kernels, margins, and low-dimensional mappings. In *Algorithmic Learning Theory*, pages 1–12, 2008.

R. F. Barber and E. J. Candès. Controlling the false discovery rate via knockoffs. *The Annals of Statistics*, 43(5):2055–2085, 10 2015.

Y. Benjamini and T. P. Speed. Summarizing and correcting the GC content bias in high-throughput sequencing. *Nucleic Acids Research*, 40(10), 2012.

D. Bernstein, B. Saeed, C. Squires, and C. Uhler. Ordering-based causal structure learning in the presence of latent variables. *Proceedings of Machine Learning Research*, 108:4098–4108, 2020.

A. Beskos, N. Pillai, G. Roberts, J. Sanz-Serna, and A. Stuart. Optimal tuning of the hybrid Monte Carlo algorithm. *Bernoulli*, 19(5A):1501–1534, 11 2013.

J. Bien, J. Taylor, and R. Tibshirani. A Lasso for hierarchical interactions. *The Annals of Statistics*, 41(3):1111–1141, 2013.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Workshop on Computational Learning Theory*, pages 144–152, 1992.

P. Bühlmann, J. Peters, and J. Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. *Annals of Statistics*, 42(6):2526–2556, 12 2014.

J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, Mar. 2010.

T. Campbell and T. Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. In *International Conference on Machine Learning*, 2018.

T. Campbell and T. Broderick. Automated scalable Bayesian inference via Hilbert coresets. *Journal of Machine Learning Research*, 2019.

E. Candes and T. Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

B. Carpenter, D. Lee, M. A. Brubaker, A. Riddell, A. Gelman, B. Goodrich, J. Guo, M. Hoffman, M. Betancourt, and P. Li. Stan: A probabilistic programming language, 2019.

C. Carvalho, N. Polson, and J. Scott. Handling sparsity via the horseshoe. In *International Conference on Artificial Intelligence and Statistics*, 2009.

E. Cerami, E. Demir, N. Schultz, B. S. Taylor, and C. Sander. Automated network analysis identifies core pathways in glioblastoma. *PLOS ONE*, 5:1–10, 02 2010.

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10:273–304, 1995.

V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. Sparse and low-rank matrix decompositions. In *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 962–967, 2009.

V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky. Latent variable graphical model selection via convex optimization. *Annals of Statistics*, 40(4):1935–1967, 08 2012.

W. Chang, C. Li, Y. Yang, and B. Poczos. Data-driven random Fourier features using Stein effect. In *International Joint Conference on Artificial Intelligence*, pages 1497–1503, 2017.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, pages 33–61, 1998.

D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

H. Chipman. Bayesian variable selection with related predictors. *The Canadian Journal of Statistics*, 24(1):17–36, 1996.

H. Cho, B. Berger, and J. Peng. Reconstructing causal biological networks through active learning. *PLoS ONE*, 2016.

S. Choy and C. Chan. Scale mixtures distributions in insurance applications. *Journal of the IAA*, 33:93–104, 2003.

K. Chwialkowski, H. Strathmann, and A. Gretton. A kernel test of goodness of fit. In *International Conference on Machine Learning*, 2016.

D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *Annals of Statistics*, 40(1):294–321, 02 2012.

C. Cortes, M. Mohri, and A. Talwalkar. On the impact of kernel approximation on learning accuracy. In *International Conference on Artificial Intelligence and Statistics*, 2010.

S. M. Curtis, S. Banerjee, and S. Ghosal. Fast bayesian model assessment for nonparametric additive regression, 2014.

A. Daniely, R. Frostig, V. Gupta, and Y. Singer. Random features for compositional kernels. *arXiv:1703.07872*, 2017.

C. Davis and W. Kahan. The rotation of eigenvectors by a perturbation III. *SIAM J. Numer. Anal*, 7:1–46, 1970.

A. Dixit, O. Parnas, B. Li, J. Chen, C. Fulco, L. Jerby-Arnon, N. Marjanovic, D. Dionne, T. Burks, R. Raychowdhury, B. Adamson, T. Norman, E. Lander, J. Weissman, N. Friedman, and A. Regev. Perturb-seq: dissecting molecular circuits with scalable single-cell RNA profiling of pooled genetic screens. *Cell*, pages 1853–1866, 2016.

P. Drineas and M. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, pages 2153–2175, 2005.

N. Durrande, D. Ginsbourger, O. Roustant, and L. Carraro. ANOVA kernels and RKHS of zero mean functions for model-based sensitivity analysis. *Journal of Multivariate Analysis*, 115(C):57–67, 2013.

A. El Alaoui and M. Mahoney. Fast randomized kernel methods with statistical guarantees. In *Advances in Neural Information Processing Systems*, 2015.

B. Ellis and W. H. Wong. Learning causal Bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103:778–789, 2008.

J. Fan, Y. Liao, and M. Mincheva. Large covariance estimation by thresholding principal orthogonal complements. *Journal of the Royal Statistical Society Series B*, 75(4):603–680, September 2013.

J. H. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1): 1–67, 03 1991.

N. Friedman and D. Koller. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–125, 2003.

N. Friedman and I. Nachman. Gaussian process networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, page 211–219, 2000.

N. Friedman, M. Goldszmidt, and A. J. Wyner. Data analysis with Bayesian networks: A bootstrap approach. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7(3-4):601–620, 2000a.

N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. In *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*, pages 127–135, 2000b.

B. Frot, P. Nandy, and M. H. Maathuis. Robust causal structure learning with some hidden variables. *Journal of the Royal Statistical Society: Series B*, 81(3):459–487, 2019.

J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson. GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. In *Advances in Neural Information Processing Systems*. 2018.

D. Geiger and D. Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

A. Gelman, J. Hill, and M. Yajimam. Why we (usually) don't have to worry about multiple comparisons. *Journal of Research on Educational Effectiveness*, pages 189–211, 2012.

E. George and R. McCulloch. Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, 88(423):881–889, 1993.

A. Ghassami, S. Salehkaleybar, N. Kiyavash, and E. Bareinboim. Budgeted experiment design for causal structure learning. In *International Conference on Machine Learning*, 2018.

S. B. Gillispie and M. D. Perlman. Enumerating Markov equivalence classes of acyclic digraph models. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, 2001.

C. Greene, N. Sinnott-Armstrong, D. S. Himmelstein, P. Park, J. Moore, and B. Harris. Multifactor dimensionality reduction for graphics processing units enables genome-wide testing of epistasis in sporadic ALS. *Bioinformatics*, 26(5):694–695, 2010.

A. Gretton, K. Fukumizu, C. H., L. Song, B. Schölkopf, and A. Smola. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems*, pages 585–592, 2008.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, pages 723–773, 2012.

J. Griffin and P. Brown. Hierarchical shrinkage priors for regression models. *Bayesian Analysis*, 12:135–159, 2017.

M. Grzegorczyk and D. Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71: 265–305, 2008.

C. Gu and G. Wahba. Smoothing spline ANOVA with component-wise bayesian "confidence intervals". *Journal of Computational and Graphical Statistics*, 2(1): 97–117, 1993.

S. Gunn and J. Kandola. Structural modelling with sparse kernels. *Machine Learning*, 48:137–163, 2004.

L. Gyorfi, M. Kohler, A. K. Krzyzak, and H. Walk. A distribution-free theory of nonparametric regression. *Journal of the American Statistical Association*, 98(464): 1084–1084, 2003.

N. Halko, P. Martinsson, and J. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, pages 217–288, 2011.

H. Hamdan, J. Nolan, M. Wilson, and K. Dardia. Using scale mixtures of normals to model continuously compounded returns. *Journal of Modern Applied Statistical Methods*, 2005.

R. Hamid, Y. Xiao, A. Gittens, and D. DeCoste. Compact random feature maps. In *International Conference on International Conference on Machine Learning*, 2014.

B. S. Harrington and C. M. Annunziata. NF-kB signaling in ovarian cancer. *Cancers (Basel)*, 8, 08 2019.

A. Hauser and P. Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13(1):2409–2464, 2012.

A. Hauser and P. Bühlmann. Two optimal strategies for active learning of causal models from interventional data. *International Journal of Approximate Reasoning*, 55:926–939, 2014.

A. Hauser and P. Bühlmann. Jointly interventional and observational data: estimation of interventional Markov equivalence classes of directed acyclic graphs. *Journal of the Royal Statistical Society Series B*, 77(1):291–318, 2015.

D. Heckerman, C. Meek, and G. Cooper. A Bayesian approach to causal discovery. Technical report, Microsoft Research, 1997.

M. Hoffman and A. Gelman. The No-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1): 1593–1623, 2014.

T. Hofmann, B. Schölkopf, and A. Smola. Kernel methods in machine learning. *The Annals of Statistics*, pages 1171–1220, 2008.

J. Honorio and Y.-J. Li. The error probability of random Fourier features is dimensionality independent. *arXiv:1710.09953*, 2017.

G. Hooker. Generalized functional anova diagnostics for high-dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16(3): 709–732, 2007.

T. Horel, T. Campbell, L. Masoero, R. Agrawal, A. Papachristos, and D. Roithmayr. The contagiousness of police violence. *In Preparation*.

P. Hoyer, D. Janzing, J. M. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems*, volume 21, 2009.

C. Hsieh, K. Chang, C. Lin, S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *International Conference on Machine Learning*, pages 408–415, 2008.

D. Hsu, S. M. Kakade, and T. Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3):569–600, 2014.

J. Z. Huang. Projection estimation in multiple regression with application to functional anova models. *Annals of Statistics*, 26(1):242–272, 02 1998. doi: 10.1214/aos/1030563984.

P. Huang, H. Avron, T. Sainath, V. Sindhwani, and B. Ramabhadran. Kernel methods match deep neural networks on TIMIT. In *International Conference on Acoustics, Speech and Signal Processing*, pages 205–209, May 2014.

J. Huggins, M. Kasprzak, T. Campbell, and T. Broderick. Practical bounds on the error of bayesian posterior approximations: A nonasymptotic approach. *arXiv:1809.09505*, 2018.

W. Johnson, J. Lindenstrauss, and G. Schechtman. Extensions of Lipschitz maps into Banach spaces. *Israel Journal of Mathematics*, pages 129–138, 1986.

S. Kakade and G. Shakhnarovich. Lecture notes in large scale learning, 2009. URL `http://ttic.uchicago.edu/~gregory/courses/LargeScaleLearning/lectures/jl.pdf`.

M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007.

P. Kar and H. Karnick. Random feature maps for dot product kernels. In *International Conference on Artificial Intelligence and Statistics*, pages 583–591, 2012.

M. Koivisto and K. Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

J. Kuipers and G. Moffa. Partition MCMC for inference on acyclic digraphs. *Journal of the American Statistical Association*, 112:282–299, 2017.

J. Kuipers, G. Moffa, and D. Heckerman. Addendum on the scoring of Gaussian directed acyclic graphical models. *The Annals of Statistics*, 42:1689–1691, 2014.

M. Kusner, J. Loftus, C. Russell, and R. Silva. Counterfactual fairness. In *Advances in Neural Information Processing Systems*. 2017a.

M. J. Kusner, J. Loftus, C. Russell, and R. Silva. Counterfactual fairness. In *Advances in Neural Information Processing Systems*, volume 30, 2017b.

G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *Journnal of Machine Learning Research*, page 27–72, 2004.

S. Lauritzen. *Graphical Models.* Oxford University Press, 1996.

Q. Le, T. Sarlos, and A. Smola. Fastfood - approximating kernel expansions in loglinear time. In *International Conference on Machine Learning*, 2013.

J. T. Leek and J. D. Storey. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLOS Genetics*, 3(9), September 2007.

P. Li, T. Hastie, and K. Church. Very sparse random projections. In *International Conference on Knowledge Discovery and Data Mining*, pages 287–296, 2006.

M. Lim and T. Hastie. Learning interactions via hierarchical group-lasso regularization. *Journal of Computational and Graphical Statistics*, 24(3):627–654, 2015.

W. Lim, R. Du, B. Dai, K. Jung, L. Song, and H. Park. Multi-scale Nystrom method. In *International Conference on Artificial Intelligence and Statistics*, 2018.

Y. Lin and H. H. Zhang. Component selection and smoothing in multivariate nonparametric regression. *Annals of Statistics*, 34(5):2272–2297, 10 2006.

I. Lipkovich, A. Dmitrienko, and R. D'Agostino. Tutorial in biostatistics: data-driven subgroup identification and analysis in clinical trials. *Statistics in Medicine*, 36: 136–196, 2017.

H. Liu, L. Wasserman, J. D. Lafferty, and P. K. Ravikumar. Spam: Sparse additive models. In *Advances in Neural Information Processing Systems*, pages 1201–1208. 2008.

Y. Lou, R. Caruana, J. Gehrke, and G. Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–631. ACM, 2013.

D. MacKay. *Information Theory, Inference & Learning Algorithms.* Cambridge University Press, 2002.

D. J. C. MacKay. *Introduction to Monte Carlo Methods*, pages 175–204. Springer Netherlands, 1998.

D. Madigan and J. York. Bayesian graphical models for discrete data. *International Statistical Review*, 63:215–232, 1995.

B. Maher. Personal genomes: The case of the missing heritability. *Nature*, pages 18–21, 2008.

C. Margossian, A. Vehtari, D. Simpson, and R. Agrawal. Hamiltonian Monte Carlo using an embedded Laplace approximation. In *Neural Information Processing Systems*, 2020.

F. Marguerite and W. Philip. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, pages 95–110, 1956.

S. Mendelson. On the performance of kernel classes. *Journal of Machine Learning Research*, pages 759–771, 2003.

J. M. Mooij, D. Janzing, J. Peters, and B. Schölkopf. Regression by dependence minimization and its application to causal inference in additive noise models. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382, pages 745–752, 2009.

J. M. Mooij, J. Peters, D. Janzing, J. Zscheischler, and B. Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, 17(32):1–102, 2016.

G. Morota and D. Gianola. Kernel-based whole-genome prediction of complex traits: a review. *Frontiers in Genetics*, 5:363, 2014.

K. Murphy. Active learning of causal Bayes net structure. Technical report, 2001.

C. Musco and C. Musco. Recursive sampling for the Nyström method. In *Advances in Neural Information Processing Systems*, 2017.

K. Nakagawa, S. Suzumura, M. Karasuyama, K. Tsuda, and I. Takeuchi. Safe pattern pruning: An efficient approach for predictive pattern mining. In *International Conference on Knowledge Discovery and Data Mining*, 2016.

R. O. Ness, K. Sachs, P. Mallick, and O. Vitek. A Bayesian active learning experimental design for inferring signaling networks. *Journal of Computational Biology*, 25(7): 709–725, 2018.

T. Niinimaki, P. Parviainen, and M. Koivisto. Structure discovery in Bayesian networks by sampling partial orders. *Journal of Machine Learning Research*, 17:2002–2048, 2016.

J. Pearl. Causality: Models, reasoning, and inference. *Econometric Theory*, 19 (675-685):46, 2003.

J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2nd edition, 2009.

J. Pennington, F. Yu, and S. Kumar. Spherical random features for polynomial kernels. In *Advances in Neural Information Processing Systems*, pages 1846–1854, 2015.

J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 15(1):2009–2053, 2014. ISSN 1532-4435.

J. Piironen and A. Vehtari. Sparsity information and regularization in the horseshoe and other shrinkage priors. *Electronic Journal of Statistics*, 11:5018–5051, 2017.

N. S. Pillai, A. M. Stuart, and A. H. Thiéry. Optimal scaling and diffusion limits for the Langevin algorithm in high dimensions. *The Annals of Applied Probability*, 22 (6):2320–2356, 12 2012.

A. Price, N. Patterson, R. Plenge, M. Weinblatt, N. Shadick, and D. Reich. Principal components analysis corrects for stratification in genome-wide association studies. *Nature Genetics*, 2006.

J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems*, 2007.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2008.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

T. Richardson and P. Spirtes. Ancestral graph markov models. *Annals of Statistics*, 30(4):962–1030, 08 2002.

D. Risso, K. Schwartz, G. Sherlock, and S. Dudoit. GC-content normalization for RNA-seq data. *BMC Bioinformatics*, 12(1):480, Dec 2011.

J. Robins, M. A. Hernan, and B. Brumback. Marginal structural models and causal inference in epidemiology. *Epidemiology*, 11:550–60, 2000a.

J. M. Robins, M. A. Hernan, and B. Brumback. Marginal structural models and causal inference in epidemiology, 2000b.

A. Rudi and L. Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, 2017.

A. Rudi, R. Camoriano, and L. Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, 2015.

W. Rudin. *Functional Analysis*. International series in pure and applied mathematics. Tata McGraw-Hill, 1974.

W. Rudin. *Fourier Analysis on Groups*, chapter The Basic Theorems of Fourier Analysis. Wiley, 1994.

Y. Samo and S. Roberts. Generalized spectral kernels. *arXiv:1506.02236*, 2015.

C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *International Conference on Machine Learning*, pages 515–521, 1998.

T. Schaffter, D. Marbach, and D. Floreano. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, 27:2263–2270, 2011.

F. Scheipl, L. Fahrmeir, and T. Kneib. Spike-and-slab priors for function selection in structured additive regression models. *Journal of the American Statistical Association*, 107(500):1518–1532, 2012.

B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

B. Schölkopf, A. Smola, and K. Müller. Kernel principal component analysis. In *Artificial Neural Networks*, pages 583–588, 1997.

R. Shah. Modelling interactions in high-dimensional data with backtracking. *Journal of Machine Learning Research*, 17(207):1–31, 2016.

R. D. Shah, B. Frot, G.-A. Thanei, and N. Meinshausen. Right singular vector projection graphs: fast high dimensional covariance matrix estimation under latent confounding. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(2):361–389, 2020.

W. Shen, Z. Yang, and J. Wang. Random features for shift-invariant kernels with moment matching. In *Association for the Advancement of Artificial Intelligence Conference*, 2017.

L. Slim, C. Chatelain, C. Azencott, and J. Vert. Novel methods for epistasis detection in genome-wide association studies. *bioRxiv:325993*, 2018.

L. Solus, Y. Wang, L. Matejovicova, and C. Uhler. Consistency guarantees for permutation-based causal inference algorithms. *Biometrika*, page asaa104, 2020.

T. Soma and Y. Yoshida. Maximizing monotone submodular functions over the integer lattice. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 325–336. Springer, 2016.

T. Soma, N. Kakimura, K. Inaba, and K. Kawarabayashi. Optimal budget allocation: Theoretical guarantee and efficient algorithm. In *International Conference on International Conference on Machine Learning*, 2014.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search.* The MIT Press, 2nd edition, 2000.

B. Sriperumbudur and N. Sterge. Approximate kernel PCA using random features: Computational vs. statistical trade-off. *arXiv:1706.06296*, 2017.

B. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, pages 1517–1561, 2010.

C. J. Stone. The use of polynomial splines and their tensor products in multivariate function estimation. *Annals of Statistics*, 22(1):118–171, 03 1994. doi: 10.1214/aos/1176325361.

G. Su, O. F. Christensen, T. Ostersen, M. Henryon, and M. S. Lund. Estimating additive and non-additive genetic variances and predicting genetic merits using genome-wide dense single nucleotide polymorphism markers. *PLOS ONE*, 7(9):1–7, 09 2012.

D. Sutherland and J. Schneider. On the error of random Fourier features. In *Conference on Uncertainty in Artificial Intelligence*, pages 862–871, 2015.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

A. Talwalkar. *Matrix Approximation for Large-scale Learning.* PhD thesis, New York University, 2010.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, pages 267–288, 1994.

M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 567–574, 2009.

S. Tong and D. Koller. Active learning for structure in Bayesian networks. In *International Joint Conference on Artificial Intelligence*, 2001.

B. Trippe, J. Huggins, R. Agrawal, and T. Broderick. LR-GLM: High-dimensional Bayesian inference using low-rank data approximations. In *International Conference on Machine Learning*, 2019.

R. Turner and M. Sahani. Two problems with variational expectation maximisation for time-series models. In *Bayesian Time series models*, pages 109–130. Cambridge University Press, 2011.

A. W. Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

V. Vapnik. *Statistical Learning Theory.* John Wiley & Sons, New York, 1998.

V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation and signal processing. In *Advances in Neural Information Processing Systems*, pages 281–287, 1997.

V. N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag, 1995.

T. S. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Uncertainty in Artificial Intelligence*, volume 6, page 255, 1991.

T. S. Verma and J. Pearl. An algorithm for deciding if a set of observed independencies has a causal explanation. In *Uncertainty in Artificial Intelligence*, 1992.

M. Wainwright and E. Simoncelli. Scale mixtures of Gaussians and the statistics of natural images. In *International Conference on Neural Information Processing Systems*, 1999.

W. Wang and J. Fan. Asymptotics of empirical eigenstructure for high dimensional spiked covariance. *Annals of Statistics*, 45(3):1342–1374, 06 2017.

Y. Wang and D. M. Blei. The blessings of multiple causes. *Journal of the American Statistical Association*, 114(528):1574–1596, 2019.

Y. Wang, L. Solus, K. Yang, and C. Uhler. Permutation-based causal inference algorithms with interventions. In *Advances in Neural Information Processing Systems*, pages 5824–5833, 2017.

R. Wei, B. Reich, J. Hoppin, and S. Ghosal. Sparse Bayesian additive nonparametric regression with application to health effects of pesticides mixtures. *Statistica Sinica*, 01 2019.

O. Weissbrod, D. Geiger, and S. Rosset. Multikernel linear mixed models for complex phenotype prediction. *Genome Research*, 2016.

C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, pages 682–688, 2001.

T. Wu, Y. Chen, T. Hastie, E. Sobel, and K. Lange. Genome-wide association analysis by Lasso penalized logistic regression. *Bioinformatics*, 25(6):714–721, 2009.

K. D. Yang, A. Katcoff, and C. Uhler. Characterizing and learning equivalence classes of causal DAGs under interventions. In *International Conference on Machine Learning*, 2018.

T. Yang, Y. Li, M. Mahdavi, R. Jin, and Z. Zhou. Nyström method vs random Fourier features - a theoretical and empirical comparison. In *Advances in Neural Information Processing Systems*, 2012.

Y. Yang, M. Pilanci, and M. J. Wainwright. Randomized sketches for kernels: Fast and optimal nonparametric regression. *The Annals of Statistics*, pages 991–1023, 2017.

F. Yu, A. Suresh, K. Choromanski, D. Holtmann-Rice, and S. Kumar. Orthogonal random features. In *Advances in Neural Information Processing Systems*, pages 1975–1983, 2016.

K. Zhang, J. Peters, D. Janzing, and B. Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Conference on Uncertainty in Artificial Intelligence*, pages 804–813, 2011.