Diss. ETH No. 25237

COMPUTATIONALLY EFFICIENT INFERENCE FOR LARGE-SCALE DATA.

A dissertation submitted to
ETH Zürich

for the degree of
Doctor of Sciences

presented by
GIAN-ANDREA THANEI
Master of Sciences in Mathematics, ETH Zürich
born June 13, 1989
citizen of Zürich, Switzerland

accepted on the recommendation of
Prof. Dr. Nicolai Meinshausen, examiner
Prof. Dr. Rajen Shah, co-examiner
Prof. Dr. Peter Bühlmann, co-examiner

2018

Before you judge a man, walk a mile in his shoes. After that who cares?... He's a mile away and you've got his shoes! – Anonymous.

Acknowledgments

I will always remember the time during my PhD as very enjoyable. Enjoyable for the things I learned, the experiences I made and the friends I got to know.

I am very thankful to my supervisors Nicolai and Rajen. They are both amazing researchers. Through their engaged and open-minded supervision, I was able to considerably improve my skills and understanding. Thank you!

I also want to thank my co-examiner Peter for sitting in my defense. Peter is a good sport and the spring of humorous atmosphere at SfS. Thank you!

Of course, research in solitude is not my thing - I had the great pleasure to work with Nicolai, Rajen, Ben, Dominik, Michael and Christina. Thank you!

A big thank you goes to the fun crew of G18: Dominik, Ema, Sholt, Niklas, Nicholas and Domagoj, who became really good friends. Life would be boring without my buddies Stefan, Philine, Andi K., Steven, Marcos, JP and many more. And above all, Mirjam. I am very grateful to my parents Ursula and Peter, my sisters, my aunt Anita and Helga & Dieter for all their support. Thank you!

Contents

Abstract ix

Zusammenfassung xi

Introduction 1

1 Random projections for large-scale regression 5
- 1.1 Introduction . . . 5
- 1.2 Theoretical results . . . 9
- 1.3 Averaged compressed least squares . . . 15
- 1.4 Discussion . . . 19
- 1.5 Appendix . . . 19

2 Fast interaction search in high-dimensional data 25
- 2.1 Introduction . . . 26
  - 2.1.1 Related work . . . 27
  - 2.1.2 Organisation of the paper . . . 30
- 2.2 The *xyz* algorithm for binary data . . . 30
  - 2.2.1 Optimality of minimal subsampling . . . 34
  - 2.2.2 The final version of xyz . . . 36
  - 2.2.3 Computational and statistical properties of xyz . . . 37
- 2.3 Interaction search on continuous data . . . 40
  - 2.3.1 Continuous $\mathbf{Y}$ and binary $\mathbf{X}$ . . . 40
  - 2.3.2 Continuous $\mathbf{Y}$ and continuous $\mathbf{X}$ . . . 41
- 2.4 Application to Lasso regression . . . 46
- 2.5 Experiments . . . 48
  - 2.5.1 Comparison of minimal subsampling and dense projections . . . 48
  - 2.5.2 Scaling . . . 49
  - 2.5.3 Run on SNP data . . . 50

2.5.4 Regression on artificial data 52
2.5.5 Regression on real data 54
2.6 Discussion 55
2.7 Appendix 56

3 Fast graphical model selection under latent confounding 79
3.1 Introduction 79
3.1.1 Related work 82
3.1.2 Organisation of the paper 84
3.2 Ridge, rowspace projections and latent confounding 84
3.2.1 Model setting 84
3.2.2 A brief review of ridge and rowspace projections . 85
3.2.3 Latent confounders in linear models 86
3.2.4 Why RP is only weakly affected by latent confounding 86
3.3 Learning graph structures with RP 88
3.3.1 Model setting 88
3.3.2 Nodewise RP 89
3.3.3 Graphical RP 89
3.3.4 SVD decomposition for numerical stability 90
3.3.5 Fast inner product computation and GRP-xyz . . 91
3.4 Experiments 92
3.4.1 Simulating a graphical model with latent confounders 92
3.4.2 Average precision 94
3.4.3 Asymptotic run time behaviour 96
3.5 Real data example: modelling gene regulatory networks across multiple tissues 97
3.5.1 The GTEX datasets 97
3.5.2 Results 99
3.6 Appendix 102
3.6.1 Proof of the RP-Rescaling lemma 102

Bibliography 103

Curriculum Vitae 109

Abstract

In many areas of science and industry, the volume of data doubles every year (Marx, 2013). Simultaneously decisions are increasingly made based on this data. These factors combined give rise to computational challenges in statistics. Classical statistical methods are often not directly applicable to large data sets due to their computational demands. At the same time, an accurate analysis is necessary to make informed decisions.

This thesis investigates computationally efficient approximations to many standard statistical techniques. These approximations most often come with a loss in statistical accuracy. The main theoretical contributions of this dissertation are in understanding this loss of accuracy.

We start with examining random projections (Thanei et al., 2016a) which allow us to represent data in lower dimensions. We then run classical methods on the projected data. This approach leads to computationally efficient estimators. Furthermore, under certain assumptions, we can show that the most important features are preserved in the projected data.

The idea of projecting data onto lower dimensions can also be applied to high-dimensional interaction search (Thanei et al., 2016b). We show that the performance of this type of interaction search strongly depends on the choice of the projection distribution. This leads us to define a projection distribution that manages the trade-off between run time and accuracy in an optimal way.

Interaction search is related to many different statistical problems, for example inference in graphical models. Graphical models are used to represent dependencies within data. The estimation of a graphical model can be computationally very intensive. Using today's methods, even estimating the graph of a precision matrix with a thousand variables can take a considerable amount of time. We introduce a new method (Thanei et al., 2018) that enables efficient graph estimation. Additionally, we demonstrate that this method is robust to pitfalls in statistical analysis such as confounding.

Zusammenfassung

In vielen Bereichen der Wissenschaft und Industrie verdoppelt sich das Datenvolumen jedes Jahr (Marx, 2013). Gleichzeitig werden viele Entscheidungen basierend auf Daten getroffen. Diese Entwicklungen führen zu grossen rechnerischen Herausforderungen in der Statistik. Klassische statistische Methoden sind aufgrund ihrer Rechenanforderungen oft nicht direkt auf grosse Datenmengen anwendbar. Zugleich ist eine genaue Analyse notwendig, um fundierte Entscheidungen zu treffen.

In dieser Arbeit werden rechnerisch effiziente Annäherungen zu einigen statistischen Standardtechniken untersucht. Solche Annäherungen haben oftmals einen Verlust an statistischer Genauigkeit zur Folge. Wir machen theoretische Beiträge um diesen Genauigkeitsverlust detailiert zu verstehen.

Zunächst untersuchen wir Zufallsprojektionen (Thanei u. a., 2016a), die es uns erlauben, Daten in niedrigeren Dimensionen darzustellen. Wir führen dann die klassischen Methoden auf den projizierten Daten aus. Dieser Ansatz führt zu rechnerisch effizienten Schätzern. Unter bestimmten Annahmen können wir ausserdem zeigen, dass die wichtigsten Informationen in den projizierten Daten erhalten bleiben.

Die Idee, Daten auf niedrigere Dimensionen zu projizieren, kann unter anderem auf die hochdimensionale Interaktionssuche angewendet werden (Thanei u. a., 2016b). Wir zeigen, dass die Rechenzeit dieser Art der Interaktionssuche stark von der Wahl der Projektionsverteilung abhängt. Dies führt uns dazu, eine Projektionsverteilung zu definieren, die den Kompromiss zwischen Rechenzeit und Genauigkeit optimal steuert.

Die Interaktionssuche ist mit vielen verschiedenen statistischen Problemen verbunden wie zum Beispiel Inferenz von grafischen Modellen. Grafische Modelle werden verwendet, um Abhängigkeiten innerhalb von Daten darzustellen. Die Schätzung eines grafischen Modells ist sehr rechenintensiv. Mit den heutigen Methoden kann bereits das Schätzen eines Grafen von tausend Variablen eine beträchtliche Zeit in Anspruch nehmen.

Wir führen eine neue Methode ein (Thanei u. a., 2018), mit der Grafen sehr effizient geschätzt werden können. Darüber hinaus argumentieren wir, dass diese Methode robust ist gegenüber Herausforderungen wie Messstörungen durch unbeobachtete Variablen.

Introduction

This thesis is a collection of three manuscripts on computationally efficient inference for large-scale data. The focus of the first chapter is on prediction using linear models. The second and third chapters focus on variable screening. The goal of all these investigations is to develop computationally efficient and accurate estimators.

By large-scale data, we either mean that the data has many samples (large n) or that the data is high-dimensional (large p). Each setting leads to its own computational and statistical challenges.

For the large sample case, the computational issue can usually be alleviated through subsampling. This entails computing estimates on small subsamples of the data and then aggregating those estimates. The aggregation step is most often some form of averaging. Many estimators based on subsampling and subsequent aggregation behave similar to estimators computed directly on the full data. The subsampling scheme, however, allows for a straightforward parallelization. Hence the challenges of this type of optimization are mostly of implementational nature.

High-dimensionality is a more challenging problem from a statistical point of view. If only a few samples are available but there are many measurements per sample, subsampling will not be of much help. Additionally, reducing the number of variables without losing important information is not straightforward. In high-dimensional problems, it is crucial to reduce the dimension of the data while conserving the dependency structure of the variables. In this thesis, we develop methods that exploit dimension reduction techniques but preserve the essential dependency structure among variables.

In the first chapter, we consider the problem of using variables $\mathbf{X}$ to predict responses $\mathbf{Y}$ with a linear model. Under suitable assumptions, we show that by randomly projecting $\mathbf{X}$ onto a matrix of smaller dimension $\mathbf{X\Phi}$ (where $\mathbf{\Phi}$ is a matrix with random Gaussian entries) and then using $\mathbf{X\Phi}$ to predict $\mathbf{Y}$ we gain a run time improvement of a quadratic order whilst retaining accuracy (Thanei et al., 2016a).

In Chapter 2, we consider interaction search in high-dimensional data. For example, in genome-wise association studies (GWAS) interactions can indicate whether the presence of two genotypes (encoded by mutations) affects a phenotype. This problem is computationally difficult as one has to scan through all possible pairs of variables which results in a run time that is quadratic in the number of variables. For GWAS data it is typical to have millions of variables but only a few hundred patients (samples). Scanning all possible pairs can take a few days or up to a few weeks with a desktop computer.

Interaction search is equivalent to closest pair search (Thanei et al., 2016b). To solve the closest pair search problem, we first project the data down to a single dimension. In the projected space, the closest pair search can be conducted in a run time that is linear in the number of variables. This computationally cheap closest pair search is repeated many times over to increase detection power. We call this procedure the xyz algorithm. We show that the xyz algorithm has subquadratic run time in the number of variables. In practice, the xyz algorithm is able to scan GWAS data (with one million variables) for strong interactions within a few minutes.

We make an important theoretical discovery regarding the choice of the projection distribution. Namely, we show that using projection matrices with Gaussian entries, whose usage is motivated by the Johnson-Lindenstrauss Lemma, does not result in run time improvements overall. We define a minimal subsampling projection and show that using the xyz algorithm with this minimal subsampling projection leads to a minimal run time for a given detection power.

Due to the massive computational cost, high-dimensional regression estimators, such as the Lasso, are not able to include interaction terms without imposing some structural assumptions. The xyz algorithm can be used in conjunction with coordinate-wise update algorithms to yield a new version of the Lasso that is able to include any type of interaction term with small computational cost.

In Chapter 3, we consider the problem of high-dimensional variable selection in the presence of latent confounders. Recent work indicates, that by implicitly projecting the signal $\boldsymbol{\beta}$ onto the rowspace (sample space) of the data $\mathbf{X}$ can give valuable information for variable selection. We show this procedure remains valid even in the presence of latent confounders (Thanei et al., 2018).

Through nodewise application, our findings naturally extend to graphical model learning. Current methods for learning graphical models in the presence of latent confounders have a run time that is at least quadratic, or most often cubic in the number of variables. Hence, these methods are not applicable to many domains facing the problem of high-dimensionality, such as gene expression data. Our estimator is able to infer the most important edges in the graph within a run time that is linear in the number of variables. This enables learning the graphical model of gene expression data in a matter of seconds compared to a matter of hours or days for established methods.

Chapter 1

Random projections for large-scale regression

Fitting linear regression models can be computationally very expensive in large-scale data analysis tasks if the sample size and the number of variables are very large. Random projections are extensively used as a dimension reduction tool in machine learning and statistics. We discuss the applications of random projections in linear regression problems, developed to decrease computational costs, and give an overview of the theoretical guarantees of the generalization error. It can be shown that the combination of random projections with least squares regression leads to similar recovery as ridge regression and principal component regression. We also discuss possible improvements when averaging over multiple random projections, an approach that lends itself easily to parallel implementation[1].

1.1 Introduction

Assume we are given a data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ (n samples of a p-dimensional random variable) and a response vector $\mathbf{Y} \in \mathbb{R}^n$. We assume a linear model for the data where $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ for some regression coefficient $\boldsymbol{\beta} \in \mathbb{R}^p$ and $\boldsymbol{\varepsilon}$ i.i.d. mean-zero noise. Fitting a regression model

---

[1] This chapter has been published in (Thanei et al., 2016a)

by standard least squares or ridge regression requires $\mathcal{O}(np^2)$ or $\mathcal{O}(p^3)$ flops. In the situation of large-scale (n, p very large) or high dimensional ($p \gg n$) data these algorithms are not applicable without having to pay a huge computational price.

Using a random projection, the data can be "compressed" either row- or column-wise. Row-wise compression was proposed and discussed in Dhillon et al. (2013b), McWilliams et al. (2014a), and Zhou et al. (2007). These approaches replace the least-squares estimator

$$\underset{\boldsymbol{\gamma}\in\mathbb{R}^p}{\operatorname{argmin}}\|\mathbf{Y}-\mathbf{X}\gamma\|_2^2 \qquad \text{with the estimator} \qquad \underset{\boldsymbol{\gamma}\in\mathbb{R}^p}{\operatorname{argmin}}\|\boldsymbol{\psi}\mathbf{Y}-\psi\mathbf{X}\gamma\|_2^2, \tag{1.1}$$

where the matrix $\boldsymbol{\Psi} \in \mathbb{R}^{m\times n}$ ($m \ll n$) is a random projection matrix and has, for example, i.i.d. $\mathcal{N}(0,1)$ entries. Other possibilities for the choice of $\boldsymbol{\Psi}$ are discussed below. The high-dimensional setting and ℓ_1-penalized regression is considered in Zhou et al. (2007), where it is shown that a sparse linear model can be recovered from the projected data under certain conditions. The optimization problem is still p-dimensional, however, and computationally expensive if the number of variables is very large.

Column-wise compression addresses this later issue by reducing the problem to a d-dimensional optimization with $d \ll p$ by replacing the least-squares estimator

$$\underset{\boldsymbol{\gamma}\in\mathbb{R}^p}{\operatorname{argmin}}\|\mathbf{Y}-\mathbf{X}\boldsymbol{\gamma}\|_2^2 \qquad \text{with the estimator} \qquad \boldsymbol{\Phi}\underset{\boldsymbol{\gamma}\in\mathbb{R}^d}{\operatorname{argmin}}\|\mathbf{Y}-\mathbf{X}\boldsymbol{\Phi}\boldsymbol{\gamma}\|_2^2, \tag{1.2}$$

where the random projection matrix is now $\boldsymbol{\Phi} \in \mathbb{R}^{p\times d}$ (with $d \ll p$). By right multiplication to the data matrix $\mathbf{X}$ we transform the data matrix to $\mathbf{X}\boldsymbol{\Phi}$ and thereby reduce the number of variables from p to d and thus reducing computational complexity. The Johnson-Lindenstrauss Lemma (Dasgupta and Gupta, 2003; Indyk and Motwani, 1998; Johnson and Lindenstrauss, 1984) guarantees that the distance between two transformed sample points is approximately preserved in the column-wise compression.

Random projections have also been considered under the aspect of preserving privacy (Blocki et al., 2012). By pre-multiplication with a random projection matrix as in (1.1) no observation in the resulting matrix can be identified with one of the original data points. Similarly, post-multiplication as in (1.2) produces new variables that do not reveal the realized values of the original variables.

In many applications the random projection used in practice falls under the class of Fast Johnson-Lindenstrauss Transforms (FJLT) (Ailon and Chazelle, 2006). One instance of such a fast projection is the Subsampled Randomized Hadamard Transform (SRHT) (Tropp, 2011). Due to its recursive definition, the matrix-vector product has a complexity of $\mathcal{O}(p\log(p))$, reducing the cost of the projection to $\mathcal{O}(np\log(p))$. Other proposals that lead to speedups compared to a Gaussian random projection matrix include random sign or sparse random projection matrices (Achlioptas, 2003). Notably, if the data matrix is sparse, using a sparse random projection can exploit sparse matrix operations. Depending on the number of non-zero elements in $\mathbf{X}$, one might prefer using a sparse random projection over a FJLT that cannot exploit sparsity in the data. Importantly, using $\mathbf{X\Phi}$ instead of $\mathbf{X}$ in our regression algorithm of choice can be disadvantageous if $\mathbf{X}$ is extremely sparse and d cannot be chosen to be much smaller than p. (The projection dimension d can be chosen by cross validation.) As the multiplication by $\mathbf{\Phi}$ "densifies" the design matrix used in the learning algorithm the potential computational benefit of sparse data is not preserved.

For OLS and row-wise compression as in (1.1), where n is very large and $p < m < n$, the SRHT (and similar FJLTs) can be understood as a subsampling algorithm. It preconditions the design matrix by rotating the observations to a basis where all points have approximately uniform leverage (Dhillon et al., 2013b). This justifies uniform subsampling in the projected space which is applied subsequent to the rotation in order to reduce the computational costs of the OLS estimation. Related ideas can be found in the way columns and rows of $\mathbf{X}$ are sampled in a CUR-matrix decomposition (Mahoney and Drineas, 2009). While the approach in Dhillon et al. (2013b) focuses on the concept of leverage, McWilliams et al. (2014a) propose an alternative scheme that allows for outliers in the data and makes use of the concept of influence (Cook, 1977). Here, random projections are used to approximate the influence of each observation which is then used in the subsampling scheme to determine which observations to include in the subsample.

Using random projections column-wise as in (1.2) as a dimensionality reduction technique in conjunction with (ℓ_2 penalized) regression has been considered in Lu et al. (2013), Kabán (2014) and Maillard and Munos (2009). The main advantage of these algorithms is the computational speedup while preserving predictive accuracy. Typically, a variance reduction is traded off against an increase in bias. In general, one disadvantage of reducing the dimensionality of the data is that the coefficients in the projected space are not interpretable in terms of the original variables. Naively, one could reverse the random projection operation by projecting the coefficients estimated in the projected space back into the original space as in (1.2). For prediction purposes this operation is irrelevant, but it can be shown that this estimator does not approximate the optimal solution in the original p-dimensional coefficient space well (Zhang et al., 2013). As a remedy, Zhang et al. (2013) propose to find the dual solution in the projected space to recover the optimal solution in the original space. The proposed algorithm approximates the solution to the original problem accurately if the design matrix is low-rank or can be sufficiently well approximated by a low-rank matrix.

Lastly, random projections have been used as an auxiliary tool. As an example, the goal of McWilliams et al. (2014b) is to distribute ridge regression across variables with an algorithm called Loco. The design matrix is split across variables and the variables are distributed over processing units (workers). Random projections are used to preserve the dependencies between all variables in that each worker uses a randomly projected version of the variables residing on the other workers in addition to the set of variables assigned to itself. It then solves a ridge regression using this local design matrix. The solution is the concatenation of the coefficients found from each worker and the solution vector lies in the original space so that the coefficients are interpretable. Empirically, this scheme achieves large speedups while retaining good predictive accuracy. Using some of the ideas and results outlined in the current manuscript, one can show that the difference between the full solution and the coefficients returned by Loco is bounded.

Clearly, row- and column-wise compression can also be applied simultaneously or column-wise compression can be used together with subsampling of the data instead of row-wise compression. In the remaining sections, we will focus on the column-wise compression as it poses more difficult challenges in terms of statistical performance guarantees. While row-wise compression just reduces the effective sample size and can be expected to work in general settings as long as the compressed dimension $m < n$ is not too small (Zhou et al., 2007), column-wise compression can only work well if certain conditions on the data are satisfied and we will give an overview of these results. If not mentioned otherwise, we will refer with compressed regression and random projections to the column-wise compression.

The structure of the manuscript is as follows: We will give an overview of bounds on the estimation accuracy in the following section 1.2, including both known results and new contributions in the form of tighter bounds. In Section 1.3 we will discuss the possibility and properties of variance-reducing averaging schemes, where estimators based on different realized random projections are aggregated. Finally, Section 1.4 concludes the manuscript with a short discussion.

1.2 Theoretical results

We will discuss in the following the properties of the column-wise compressed estimator as in (1.2), which is defined as

$$\hat{\boldsymbol{\beta}}_d^{\boldsymbol{\Phi}} \;=\; \boldsymbol{\Phi} \operatorname*{argmin}_{\boldsymbol{\gamma}\in\mathbb{R}^d} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\Phi}\boldsymbol{\gamma}\|_2^2, \tag{1.3}$$

where we assume that $\boldsymbol{\Phi}$ has i.i.d. $\mathcal{N}(0, 1/d)$ entries. This estimator will be referred to as the compressed least squares estimator (CLSE) in the following. We will focus on the unpenalized form as in (1.3) but note that similar results also apply to estimators that put an additional penalty on the coefficients $\boldsymbol{\beta}$ or $\boldsymbol{\gamma}$. Due to the isotropy of the random projection, a ridge-type penalty as in Lu et al. (2013) and McWilliams et al. (2014b) is perhaps a natural choice. An interesting summary of the bounds on random projections is on the other hand that the random projection as in (1.3) already acts as a regularization and the theoretical properties of (1.3) are very much related to the properties of a ridge-type estimator of the coefficient vector in the absence of random projections.

We will restrict discussion of the properties mostly to the mean squared error (MSE)

$$\mathbb{E}_{\boldsymbol{\Phi}}\left[\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\boldsymbol{\Phi}}\|_2^2)\right]. \tag{1.4}$$

First results on compressed least squares have been given in (Maillard and Munos, 2009) in a random design setting. It was shown that the bias of the estimator (1.3) is of order $\mathcal{O}(\log(n)/d)$. This proof used a modified version of the Johnson-Lindenstrauss Lemma. A recent result (Kabán, 2014) shows that the $\log(n)$-term is not necessary for fixed design settings where $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ for some $\boldsymbol{\beta} \in \mathbb{R}^p$ and $\boldsymbol{\varepsilon}$ is i.i.d. noise, centered $\mathbb{E}_{\boldsymbol{\varepsilon}}[\boldsymbol{\varepsilon}] = 0$ and with the variance $\mathbb{E}_{\boldsymbol{\varepsilon}}[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}'] = \sigma^2 I_{n\times n}$. We will work with this setting in the following.

The following result of (Kabán, 2014) gives a bound on the MSE for fixed design.

Theorem 1.2.1. *(Kabán, 2014) Assume fixed design and* $\mathrm{Rank}(\mathbf{X}) \geq d$. *Then*

$$\mathbb{E}_{\mathbf{\Phi}}\left[\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2)\right] \;\leq\; \sigma^2 d + \frac{\|\mathbf{X}\boldsymbol{\beta}\|_2^2}{d} + \mathrm{trace}(\mathbf{X}^T\mathbf{X})\frac{\|\boldsymbol{\beta}\|_2^2}{d}. \quad (1.5)$$

Proof. See Appendix. □

Compared with (Maillard and Munos, 2009), the result removes an unnecessary $\mathcal{O}(\log(n))$ term and demonstrates the $\mathcal{O}(1/d)$ behaviour of the bias. The result also illustrates the tradeoffs when choosing a suitable dimension d for the projection. Increasing d will lead to a $1/d$ reduction in the bias terms but lead to a linear increase in the estimation error (which is proportional to the dimension in which the least-squares estimation is performed). An optimal bound can only be achieved with a value of d hat depends on the unknown signal and in practice one would typically use cross-validation to make the choice of the dimension of the projection.

One issue with the bound in Theorem 1.2.1 is that the bound on the bias term in the noiseless case ($\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}$)

$$\mathbb{E}_{\mathbf{\Phi}}\left[\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2)\right] \;\leq\; \frac{\|\mathbf{X}\boldsymbol{\beta}\|_2^2}{d} + \mathrm{trace}(\mathbf{X}^T\mathbf{X})\frac{\|\boldsymbol{\beta}\|_2^2}{d} \quad (1.6)$$

is usually weaker than the trivial bound (by setting $\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}} = 0$) of

$$\mathbb{E}_{\mathbf{\Phi}}\left[\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2)\right] \;\leq\; \|\mathbf{X}\boldsymbol{\beta}\|_2^2 \quad (1.7)$$

for most values of $d < p$. By improving the bound, it is also possible to point out the similarities between ridge regression and compressed least squares.

The improvement in the bound rests on a small modification in the original proof in (Kabán, 2014). The idea is to bound the bias term of (1.4) by optimizing over the upper bound given in the foregoing theorem.

Specifically, one can use the inequality

$$\begin{aligned}\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta}-&\mathbf{X}\mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{X}^T\mathbf{X}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}\|_2^2]]\\ &\leq \min_{\hat{\boldsymbol{\beta}}\in\mathbb{R}^p} \mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta}-\mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^T\hat{\boldsymbol{\beta}}\|_2^2]],\end{aligned}$$

instead of

$$\begin{aligned}\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta}-&\mathbf{X}\mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{X}^T\mathbf{X}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}\|_2^2]]\\ &\leq \mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta}-\mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^T\boldsymbol{\beta}\|_2^2]].\end{aligned}$$

To simplify the exposition we will from now on always assume we have rotated the design matrix to an orthogonal design so that the Gram matrix is diagonal:

$$\mathbf{\Sigma} = \mathbf{X}^T\mathbf{X} = diag(\lambda_1, ..., \lambda_p). \tag{1.8}$$

This can always be achieved for any design matrix and is thus not a restriction. It implies, however, that the optimal regression coefficients $\boldsymbol{\beta}$ are expressed in the basis in which the Gram matrix is orthogonal, this is the basis of principal components. This will turn out to be the natural choice for random projections and allows for easier interpretation of the results.
Furthermore note that in Theorem 1.2.1 we have the assumption $\mathrm{Rank}(\mathbf{X}) \geq d$, which tells us that we can apply the CLSE in the high dimensional setting $p \gg n$ as long as we choose d small enough (smaller than $\mathrm{Rank}(\mathbf{X})$, which is usually equal to n) in order to have uniqueness.
With the foregoing discussion on how to improve the bound in Theorem 1.2.1 we get the following theorem:

Theorem 1.2.2. *Assume* $\mathrm{Rank}(\mathbf{X}) \geq d$*, then the MSE (1.4) can be bounded above by*

$$\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta}-\mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2]] \leq \sigma^2 d + \sum_{i=1}^p \beta_i^2\lambda_i w_i \tag{1.9}$$

where

$$w_i = \frac{(1+1/d)\lambda_i^2 + (1+2/d)\lambda_i\,\mathrm{trace}(\mathbf{\Sigma}) + \mathrm{trace}(\mathbf{\Sigma})^2/d}{(d+2+1/d)\lambda_i^2 + 2(1+1/d)\lambda_i\,\mathrm{trace}(\mathbf{\Sigma}) + \mathrm{trace}(\mathbf{\Sigma})^2/d}. \tag{1.10}$$

Proof. See Appendix. □

The w_i are shrinkage factors. By defining the proportion of the total variance observed in the direction of the i-th principal component as

$$\alpha_i = \frac{\lambda_i}{\operatorname{trace}(\mathbf{\Sigma})}, \tag{1.11}$$

we can rewrite the shrinkage factors in the foregoing theorem as

$$w_i = \frac{(1+1/d)\alpha_i^2 + (1+2/d)\alpha_i + 1/d}{(d+2+1/d)\alpha_i^2 + 2(1+1/d)\alpha_i + 1/d}. \tag{1.12}$$

Analyzing this term shows that the shrinkage is stronger in directions of high variance compared to directions of low variance. To explain this relation in a bit more detail we compare it to ridge regression. The MSE of ridge regression with penalty term $\lambda\|\boldsymbol{\beta}\|_2^2$ is given by

$$\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\beta}^{Ridge}\|_2^2] = \sigma^2 \sum_{i=1}^{p} \Big(\frac{\lambda_i}{\lambda_i + \lambda}\Big)^2 + \sum_{i=1}^{p} \beta_i^2 \lambda_i \Big(\frac{\lambda}{\lambda + \lambda_i}\Big)^2. \tag{1.13}$$

Imagine that the signal lives on the space spanned by the first q principal directions, that is $\beta_i = 0$ for $i > q$. The best MSE we could then achieve is $\sigma^2 q$ by running a regression on the first q first principal directions. For random projections, we can see that we can indeed reduce the bias term to nearly zero by forcing $w_i \approx 0$ for $i = 1, \dots, q$. This requires $d \gg q$ as the bias factors will then vanish like $1/d$. Ridge regression on the other hand requires that the penalty λ is smaller than the q-th largest eigenvalue λ_q (to reduce the bias on the first q directions) but large enough to render the variance factor $\lambda_i/(\lambda_i+\lambda)$ very small for $i > q$. The tradeoff in choosing the penalty λ in ridge regression and choosing the dimension d for random projections is thus very similar. The number of directions for which the eigenvalue λ_i is larger than the penalty λ in ridge corresponds to the effective dimension and will yield the same variance bound as in random projections. The analogy between the MSE bounds (1.9) for random projections and (1.13) for ridge regression illustrates thus a close relationship between compressed least squares and ridge regression or principal component regression, similar to Dhillon et al. (2013a).

Instead of an upper bound for the MSE of CLSE as in Maillard and Munos (2009) and Kabán (2014), we will in the following try to derive explicit expressions for the MSE, following the ideas in Kabán (2014) and Marzetta et al. (2011) and we give a closed form MSE in the case of orthonormal predictors. The derivation will make use of the following notation:

Definition 1.2.3. *Let* $\mathbf{\Phi} \in \mathbb{R}^{p \times d}$ *be a random projection. We define the following matrices:*

$$\begin{aligned} \mathbf{\Phi}_d^{\mathbf{X}} &= \mathbf{\Phi}(\mathbf{\Phi}^T \mathbf{X}^T \mathbf{X} \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \in \mathbb{R}^{p \times p} \\ \textit{and} \quad T_d^{\mathbf{\Phi}} &= \mathbb{E}_{\mathbf{\Phi}}[\mathbf{\Phi}_d^{\mathbf{X}}] = \mathbb{E}_{\mathbf{\Phi}}[\mathbf{\Phi}(\mathbf{\Phi}^T \mathbf{X}^T \mathbf{X} \mathbf{\Phi})^{-1} \mathbf{\Phi}^T] \in \mathbb{R}^{p \times p}. \end{aligned}$$

The next Lemma (Marzetta et al., 2011) summarizes the main properties of $\mathbf{\Phi}_d^{\mathbf{X}}$ and $T_d^{\mathbf{\Phi}}$.

Lemma 1.2.4. *Let* $\mathbf{\Phi} \in \mathbb{R}^{p \times d}$ *be a random projection. Then*

i) $(\mathbf{\Phi}_d^{\mathbf{X}})^T = \mathbf{\Phi}_d^{\mathbf{X}}$ *(symmetric),*

ii) $\mathbf{\Phi}_d^{\mathbf{X}} \mathbf{X}^T \mathbf{X} \mathbf{\Phi}_d^{\mathbf{X}} = \mathbf{\Phi}_d^{\mathbf{X}}$ *(projection),*

iii) *if* $\mathbf{\Sigma} = \mathbf{X}^T \mathbf{X}$ *is diagonal* $\Rightarrow T_d^{\mathbf{\Phi}}$ *is diagonal.*

Proof. See Marzetta et al. (2011). ☐

The important point of this lemma is that when we assume orthogonal design then $T_d^{\mathbf{\Phi}}$ is diagonal. We will denote this by

$$T_d^{\mathbf{\Phi}} = \operatorname{diag}(1/\eta_1, ..., 1/\eta_p),$$

where the terms η_i are well defined but without an explicit representation.

A quick calculation reveals the following theorem:

Theorem 1.2.5. *Assume* $\mathrm{Rank}(\mathbf{X}) \geq d$*, then the MSE (1.4) equals*

$$\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2]] = \sigma^2 d + \sum_{i=1}^{p} \beta_i^2 \lambda_i \Big(1 - \frac{\lambda_i}{\eta_i}\Big). \tag{1.14}$$

Furthermore we have

$$\sum_{i=1}^{p} \frac{\lambda_i}{\eta_i} = d. \tag{1.15}$$

Proof. See Appendix. ☐

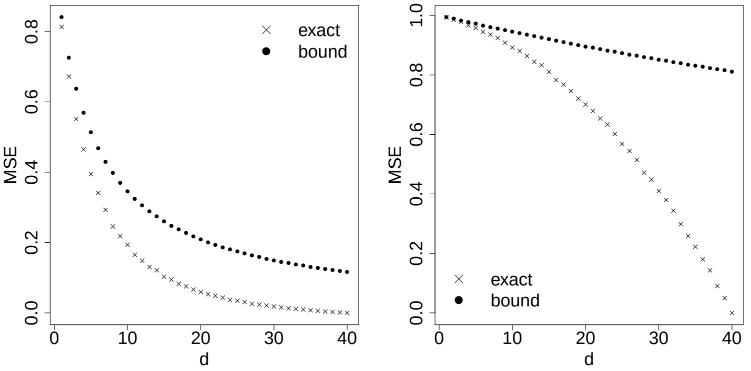


Figure 1.1: Numerical simulations of the bounds in Theorems 1.2.2 and 1.2.5. Left: the exact factor $(1-\lambda_1/\eta_1)$ in the MSE is plotted versus the bound w_1 as a function of the projection dimension d. Right: the exact factor $(1-\lambda_p/\eta_p)$ in the MSE and the upper bound w_p. Note that the upper bound works especially well for small values of d and for the larger eigenvalues and is always below the trivial bound 1.

By comparing coefficients in Theorems 1.2.2 and 1.2.5, we obtain the following corollary.

Corollary 1.2.6. *Assume* Rank($\mathbf{X}$) $\geq d$, *then*

$$\forall i \in \{1, ..., p\}: \quad 1 - \frac{\lambda_i}{\eta_i} \leq w_i \tag{1.16}$$

As already mentioned in general we cannot give a closed-form expression for the terms η_i in general. However, for some special cases (1.26) can help us to get to an exact form of the MSE of CLSE. If we assume orthonormal design ($\mathbf{\Sigma} = CI_{p \times p}$) then we have that λ_i/η_i is a constant for all i and and thus, by (1.26), we have $\eta_i = Cp/d$. This gives

$$\mathbb{E}_{\boldsymbol{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\boldsymbol{\Phi}}\|_2^2]] = \sigma^2 d + C\sum_{i=1}^{p} \beta_i^2 \Big(1 - \frac{d}{p}\Big), \tag{1.17}$$

and thus we end up with a closed form MSE for this special case.

Providing the exact mean-squared errors allows us to quantify the conservativeness of the upper bounds. The upper bound has been shown to give a good approximation for small dimensions d of the projection and for the signal contained in the larger eigenvalues.

1.3 Averaged compressed least squares

We have so far looked only into compressed least squares estimator with one single random projection. An issue in practice of the compressed least squares estimator is its variance due to the random projection as an additional source of randomness. This variance can be reduced by averaging multiple compressed least squares estimates coming from different random projections. In this section we will show some properties of the averaged compressed least squares (ACLSE) estimator and discuss its advantage over the CLSE.

Definition 1.3.1. *(ACLSE) Let* $\{\mathbf{\Phi}_1, ..., \mathbf{\Phi}_K\} \in \mathbb{R}^{p \times d}$ *be independent random projections, and let* $\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}_i}$ *for all* $i \in \{1, ..., K\}$ *be the respective compressed least squares estimators. We define the averaged compressed least squares estimator (ACLSE) as*

$$\hat{\boldsymbol{\beta}}_d^K := \frac{1}{K} \sum_{i=1}^{K} \hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}_i}. \tag{1.18}$$

One major advantage of this estimator is that it can be calculated in parallel with the minimal number of two communications, one to send the data and one to receive the result. This means that the asymptotic computational cost of $\hat{\boldsymbol{\beta}}_d^K$ is equal to the cost of $\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}$ if calculations are done on K different processors. To investigate the MSE of $\hat{\boldsymbol{\beta}}_d^K$, we restrict ourselves for simplicity to the limit case

$$\hat{\boldsymbol{\beta}}_d = \lim_{K \to \infty} \hat{\boldsymbol{\beta}}_d^K \tag{1.19}$$

and instead only investigate $\hat{\boldsymbol{\beta}}_d$. The reasoning being that for large enough values of K (say $K > 100$) the behaviour of $\hat{\boldsymbol{\beta}}_d$ is very similar to $\hat{\boldsymbol{\beta}}_d^K$. The exact form of the MSE in terms of the η_i's is given in Kabán (2014). Here we build on these results and give an explicit upper bound for the MSE.

Theorem 1.3.2. *Assume* $\mathrm{Rank}(\mathbf{X}) \geq d$*. Define*

$$\tau = \sum_{i=1}^{p} \left(\frac{\lambda_i}{\eta_i} \right)^2.$$

The MSE of $\hat{\boldsymbol{\beta}}_d$ *can be bounded from above by*

$$\mathbb{E}_{\boldsymbol{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d\|_2^2]] \leq \sigma^2\tau + \sum_{i=1}^{p} \beta_i^2 \lambda_i w_i^2,$$

where the w_i*'s are given (as in Theorem 1.2.1) by*

$$w_i = \frac{(1+1/d)\lambda_i^2 + (1+2/d)\lambda_i \operatorname{trace}(\boldsymbol{\Sigma}) + \operatorname{trace}(\boldsymbol{\Sigma})^2/d}{(d+2+1/d)\lambda_i^2 + 2(1+1/d)\lambda_i \operatorname{trace}(\boldsymbol{\Sigma}) + \operatorname{trace}(\boldsymbol{\Sigma})^2/d}.$$

and

$$\tau \in [d^2/p, d].$$

Proof. See Appendix. □

Comparing averaging to the case where we only have one single estimator we see that there are two differences: First the variance due to the model noise $\boldsymbol{\varepsilon}$ turns into $\sigma^2\tau$ with $\tau \in [d^2/p, d]$, thus $\tau \leq d$. Secondly the shrinkage factors w_i in the bias are now squared, which in total means that the MSE of $\hat{\boldsymbol{\beta}}_d$ is always smaller or equal to the MSE of a single estimator $\hat{\boldsymbol{\beta}}_d^{\boldsymbol{\Phi}}$.

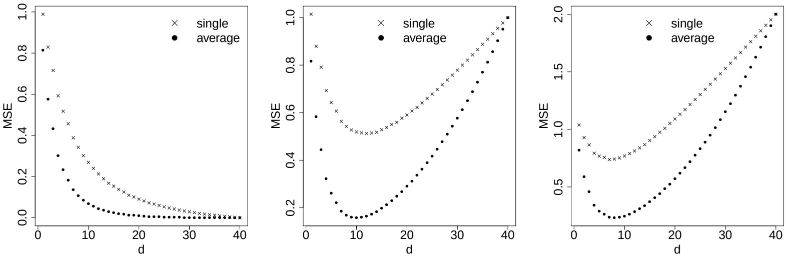


Figure 1.2: MSE of averaged compressed least squares (circle) versus the MSE of the single estimator (cross) with covariance matrix $\boldsymbol{\Sigma}_{i,i} = 1/i$. On the left with $\sigma^2 = 0$ (only bias), in the middle $\sigma^2 = 1/40$ and on the left $\sigma^2 = 1/20$. One can clearly see the quadratic improvement in terms of MSE as predicted by Theorem 1.3.2.

We investigate the behavior of τ as a function of d in three different situations (Figure 1.2). We first look at two extreme cases of covariance matrices for which the respective upper and lower bounds $[d^2/p, d]$ for τ

are achieved. For the lower bound, let $\mathbf{\Sigma} = I_{p\times p}$ be orthonormal. Then $\lambda_i/\eta_i = c$ for all i, as above. From

$$\sum_{i=1}^{p} \lambda_i/\eta_i = d$$

we get $\lambda_i/\eta_i = d/p$. This leads to

$$\tau = \sum_{i=1}^{p} (\lambda_i/\eta_i)^2 = p\frac{d^2}{p^2} = \frac{d^2}{p},$$

which reproduces the lower bound. We will not be able to reproduce the upper bound exactly for all $d \leq p$. But we can show that for any d there exists a covariance matrix $\mathbf{\Sigma}$, such that the upper bound is reached. The idea is to consider a covariance matrix that has equal variance in the first d direction and almost zero in the remaining $p-d$. Define the diagonal covariance matrix

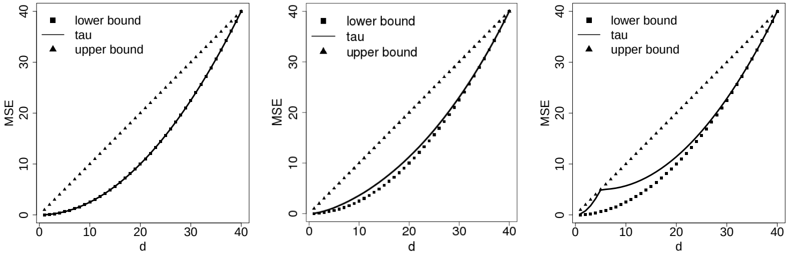


Figure 1.3: Simulations of the variance factor τ (line) as a function of d for three different covariance matrices and in lower bound (d^2/p) and upper bound (d) (square, triangle). On the left ($\mathbf{\Sigma} = I_{p\times p}$) τ as proven reaches the lower bound. In the middle ($\mathbf{\Sigma}_{i,i} = 1/i$) τ reaches almost the lower bound, indicating that in most practical examples τ will be very close to the lower bound and thus averaging improves MSE substantially compared to the single estimator. On the right the extreme case example from (1.20) with $d = 5$, where τ reaches the upper bound for $d = 5$.

$$\mathbf{\Sigma}_{i,j} = \begin{cases} 1, & \text{if } i = j \text{ and } i \leq d \\ \epsilon, & \text{if } i = j \text{ and } i > d \\ 0, & \text{if } i \neq j \end{cases}. \tag{1.20}$$

We show $\lim_{\epsilon\to 0} \tau = d$. For this decompose Φ into two matrices $\Phi_d \in \mathbb{R}^{d\times d}$ and $\Phi_r \in \mathbb{R}^{(p-d)\times d}$:

$$\Phi = \begin{pmatrix} \Phi_d \\ \Phi_r \end{pmatrix}.$$

The same way we define $\boldsymbol{\beta}_d$, $\boldsymbol{\beta}_r$, $\mathbf{X}_d$ and $\mathbf{X}_r$. Now we bound the approximation error of $\hat{\boldsymbol{\beta}}_d^{\Phi}$ to extract information about λ_i/η_i. Assume a squared data matrix $(n = p)$ $\mathbf{X} = \sqrt{\boldsymbol{\Sigma}}$, then

$$\begin{aligned}
\mathbb{E}_{\Phi}[\underset{\boldsymbol{\gamma}\in\mathbb{R}^d}{\operatorname{argmin}}\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\Phi\boldsymbol{\gamma}\|_2^2] &\le \mathbb{E}_{\Phi}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\Phi\Phi_d^{-1}\boldsymbol{\beta}_d\|_2^2] \\
&= \mathbb{E}_{\Phi}[\|\mathbf{X}_r\boldsymbol{\beta}_r - \mathbf{X}_r\Phi_r\Phi_d^{-1}\boldsymbol{\beta}_d\|_2^2] \\
&= \epsilon\mathbb{E}_{\Phi}[\|\boldsymbol{\beta}_r - \Phi_r\Phi_d^{-1}\boldsymbol{\beta}_d\|_2^2] \\
&\le \epsilon(2\|\boldsymbol{\beta}_r\|_2^2 + 2\|\boldsymbol{\beta}_d\|_2^2\mathbb{E}_{\Phi}[\|\Phi_r\|_2^2]\mathbb{E}_{\Phi}[\|\Phi_d^{-1}\|_2^2]) \\
&\le \epsilon C,
\end{aligned}$$

where C is independent of ϵ and bounded since the expectation of the smallest and largest singular values of a random projection is bounded. This means that the approximation error decreases to zero as we let $\epsilon \to 0$. Applying this to the closed form for the MSE of $\hat{\boldsymbol{\beta}}_d^{\Phi}$ we have that

$$\sum_{i=1}^{p} \beta_i^2\lambda_i\Big(1 - \frac{\lambda_i}{\eta_i}\Big) \le \sum_{i=1}^{d} \beta_i^2\Big(1 - \frac{\lambda_i}{\eta_i}\Big) + \epsilon \sum_{i=d+1}^{p} \beta_i^2\Big(1 - \frac{\lambda_i}{\eta_i}\Big)$$

has to go to zero as $\epsilon \to 0$, which in turn implies

$$\lim_{\epsilon\to 0} \sum_{i=1}^{d} \beta_i^2\Big(1 - \frac{\lambda_i}{\eta_i}\Big) = 0,$$

and thus $\lim_{\epsilon\to 0}\ \lambda_i/\eta_i = 1$ for all $i \in \{1, ..., d\}$. This finally yields a limit

$$\lim_{\epsilon\to 0} \sum_{i=1}^{p} \frac{\lambda_i^2}{\eta_i^2} = d.$$

This illustrates that the lower bound d^2/p and upper bound d for the variance factor τ can both be attained. Simulations suggest that τ is usually close to the lower bound, where the variance of the estimator is reduced by a factor d/p compared to a single iteration of a compressed least-squares estimator, which is on top of the reduction in the bias error term. This shows, perhaps unsurprisingly, that averaging over random projection estimators improves the mean-squared error in a Rao-Blackwellization sense. We have quantified the improvement. In practice, one would have to decide whether to run multiple versions of a compressed least-squares regression in parallel or run a single random projection with a perhaps larger embedding dimension. The computational effort and statistical error tradeoffs will depend on the implementation but the bounds above will give a good basis for a decision.

1.4 Discussion

We discussed some known results about the properties of compressed least-squares estimation and proposed possible tighter bounds and exact results for the mean-squared error. While the exact results do not have an explicit representation, they allow nevertheless to quantify the conservative nature of the upper bounds on the error. Moreover, the shown results allow to show a strong similarity of the error of compressed least squares, ridge and principal component regression. We also discussed the advantages of a form of Rao-Blackwellization, where multiple compressed least-square estimators are averaged over multiple random projections. The latter averaging procedure also allows to compute the estimator trivially in a distributed way and is thus often better suited for large-scale regression analysis. The averaging methodology also motivates the use of compressed least squares in the high dimensional setting where it performs similar to ridge regression and the use of multiple random projection will reduce the variance and result in a non random estimator in the limit, which presents a computationally attractive alternative to ridge regression.

1.5 Appendix

In this section we give proofs of the statements from the section theoretical results.

Theorem 1.2.1. (Kabán, 2014) Assume fixed design and $\mathrm{Rank}(\mathbf{X}) \geq d$, then the AMSE 1.4 can be bounded above by

$$\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2]] \leq \sigma^2 d + \frac{\|\mathbf{X}\boldsymbol{\beta}\|_2^2}{d} + \mathrm{trace}(\mathbf{X}^T\mathbf{X})\frac{\|\boldsymbol{\beta}\|_2^2}{d}. \quad (1.21)$$

Proof. (Sketch)

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\boldsymbol{\Phi}}\|_2^2]] &= \mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}(\boldsymbol{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}\|_2^2] + \sigma^2 d\\ &\leq \mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}(\boldsymbol{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\beta}\|_2^2] + \sigma^2 d\\ &= \mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\beta}\|_2^2] + \sigma^2 d.\end{aligned}$$

Finally a rather lengthy but straightforward calculation leads to

$$\mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T\boldsymbol{\beta}\|_2^2] = \frac{\|\mathbf{X}\boldsymbol{\beta}\|_2^2}{d} + \operatorname{trace}(\mathbf{X}^T\mathbf{X})\frac{\|\boldsymbol{\beta}\|_2^2}{d} \tag{1.22}$$

and thus proving the statement above. □

Theorem 1.2.2. Assume Rank($\mathbf{X}$) $\geq d$, then the AMSE (1.4) can be bounded above by

$$\mathbb{E}_{\boldsymbol{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\boldsymbol{\Phi}}\|_2^2]] \leq \sigma^2 d + \sum_{i=1}^{p}\boldsymbol{\beta}_i^2\lambda_i w_i \tag{1.23}$$

where

$$w_i = \frac{(1+1/d)\lambda_i^2 + (1+2/d)\lambda_i\operatorname{trace}(\boldsymbol{\Sigma}) + \operatorname{trace}(\boldsymbol{\Sigma})^2/d}{(d+2+1/d)\lambda_i^2 + 2(1+1/d)\lambda_i\operatorname{trace}(\boldsymbol{\Sigma}) + \operatorname{trace}(\boldsymbol{\Sigma})^2/d}. \tag{1.24}$$

Proof. We have for all $v \in \mathbb{R}^p$

$$\mathbb{E}_{\boldsymbol{\Phi}}[\min_{\hat{\boldsymbol{\gamma}}\in\mathbb{R}^d}\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\hat{\boldsymbol{\gamma}}\|_2^2] \leq \mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T v\|_2^2].$$

Which we can minimize over the whole set $\mathbb{R}^p$:

$$\mathbb{E}_{\boldsymbol{\Phi}}[\min_{\hat{\boldsymbol{\gamma}}\in\mathbb{R}^d}\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\hat{\boldsymbol{\gamma}}\|_2^2] \leq \min_{v\in\mathbb{R}^p}\mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T v\|_2^2].$$

This last expression we can calculate following the same path as in Theorem 1.2.1:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T v\|_2^2] =& \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\mathbb{E}_{\boldsymbol{\Phi}}[\boldsymbol{\Phi}\boldsymbol{\Phi}^T]v\\ &+ v^T\mathbb{E}_{\boldsymbol{\Phi}}[\boldsymbol{\Phi}\boldsymbol{\Phi}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\Phi}\boldsymbol{\Phi}^T]v\\ =& \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}v\\ &+ (1+1/d)v^T\mathbf{X}^T\mathbf{X}v + \frac{\operatorname{trace}(\boldsymbol{\Sigma})}{d}\|v\|_2^2,\end{aligned}$$

where $\mathbf{\Sigma} = \mathbf{X}^T\mathbf{X}$. Next we minimize the above expression w.r.t v. For this we take the derivative w.r.t. v and then we zero the whole expression. This yields

$$2\Big(1+\frac{1}{d}\Big)\mathbf{\Sigma}v + 2\frac{\mathrm{trace}(\mathbf{\Sigma})}{d}I_{p\times p}v - 2\mathbf{\Sigma}\boldsymbol{\beta} = 0.$$

Hence we have

$$v = \bigg(\Big(1+\frac{1}{d}\Big)\mathbf{\Sigma} + \frac{\mathrm{trace}(\mathbf{\Sigma})}{d}I_{p\times p}\bigg)^{-1}\mathbf{\Sigma}\boldsymbol{\beta},$$

which is element wise equal to

$$v_i = \frac{\beta_i\lambda_i}{(1+1/d)\lambda_i + \mathrm{trace}(\mathbf{\Sigma})/d}.$$

Define the notation $s = \mathrm{trace}(\mathbf{\Sigma})$. We now plug this back into the original expression and get

$$\begin{aligned}\min_{v\in\mathbb{R}^p}\mathbb{E}_{\mathbf{\Phi}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\mathbf{\Phi}\mathbf{\Phi}^T v\|_2^2] =&\boldsymbol{\beta}^T\mathbf{\Sigma}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\mathbf{\Sigma}v \\ &+ (1+1/d)v^T\mathbf{\Sigma}v + \frac{s}{d}\|v\|_2^2 \\ =&\sum_{i=1}^p \beta_i^2\lambda_i - 2\beta_i v_i\lambda_i + (1+1/d)v_i^2\lambda_i + s/dv_i^2 \\ =&\sum_{i=1}^p\Big(\beta_i^2\lambda_i - 2\beta_i^2\lambda_i\frac{\lambda_i}{(1+1/d)\lambda_i + s/d} \\ &+ \beta_i^2\lambda_i(1+1/d)\frac{\lambda_i^2}{((1+1/d)\lambda_i+s/d)^2} \\ &+ \beta_i^2\lambda_i\frac{s}{d}\frac{\lambda_i}{((1+1/d)\lambda_i+s/d)^2}\Big) \\ =&\sum_{i=1}^p\beta_i^2\lambda_i w_i,\end{aligned}$$

by combining the summands we get for w_i the expression mentioned in the theorem. □

Theorem 1.2.5. Assume $\mathrm{Rank}(\mathbf{X}) \geq d$, then the MSE (1.4) equals

$$\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d^{\mathbf{\Phi}}\|_2^2]] = \sigma^2 d + \sum_{i=1}^p\beta_i^2\lambda_i\Big(1-\frac{\lambda_i}{\eta_i}\Big). \qquad (1.25)$$

Furthermore we have

$$\sum_{i=1}^{p} \frac{\lambda_i}{\eta_i} = d. \tag{1.26}$$

Proof. Calculating the expectation yields

$$\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\beta - \mathbf{X}\hat{\boldsymbol{\beta}}_d\|_2^2]] = \boldsymbol{\beta}^T\mathbf{\Sigma}\boldsymbol{\beta} - 2\boldsymbol{\beta}^T\mathbf{\Sigma}T_d^{\mathbf{\Phi}}\mathbf{\Sigma}\boldsymbol{\beta} + \mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\mathbf{Y}^T\mathbf{X}\mathbf{\Phi}_d^{\mathbf{X}}\mathbf{X}^T\mathbf{Y}]].$$

Going through these terms we get:

$$\begin{aligned}
\boldsymbol{\beta}^T\mathbf{\Sigma}\boldsymbol{\beta} &= \sum_{i=1}^{p} \beta_i^2\lambda_i \\
\boldsymbol{\beta}^T\mathbf{\Sigma}T_d^{\mathbf{\Phi}}\mathbf{\Sigma}\boldsymbol{\beta} &= \sum_{i=1}^{p} \beta_i^2\frac{\lambda_i^2}{\eta_i} \\
\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\mathbf{Y}^T\mathbf{X}\mathbf{\Phi}_d^{\mathbf{X}}\mathbf{X}^T\mathbf{Y}]] &= \boldsymbol{\beta}^T\mathbf{\Sigma}\mathbb{E}_{\mathbf{\Phi}}[\mathbf{\Phi}_d^{\mathbf{X}}]\mathbf{\Sigma}\boldsymbol{\beta} + \mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\boldsymbol{\varepsilon}^T\mathbf{X}\mathbf{\Phi}_d^{\mathbf{X}}\mathbf{X}^T\boldsymbol{\varepsilon}]].
\end{aligned}$$

The first term in the last line equals $\sum_{i=1}^{p}\beta_i^2\lambda_i^2/\eta_i$. The second can be calculated in two ways, both relying on the shuffling property of the trace operator:

$$\begin{aligned}
\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\boldsymbol{\varepsilon}^T\mathbf{X}\mathbf{\Phi}_d^{\mathbf{X}}\mathbf{X}^T\boldsymbol{\varepsilon}]] &= \mathbb{E}_{\boldsymbol{\varepsilon}}[\boldsymbol{\varepsilon}^T\mathbf{X}T_d^{\mathbf{X}}\mathbf{X}^T\boldsymbol{\varepsilon}]] \\
&= \sigma^2\operatorname{trace}(\mathbf{X}T_d^{\mathbf{X}}\mathbf{X}^T) \\
&= \sigma^2\operatorname{trace}(\mathbf{\Sigma}T_d^{\mathbf{X}}) = \sum_{i=1}^{p}\frac{\lambda_i}{\eta_i}. \\
\mathbb{E}_{\mathbf{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\boldsymbol{\varepsilon}^T\mathbf{X}\mathbf{\Phi}_d^{\mathbf{X}}\mathbf{X}^T\boldsymbol{\varepsilon}]] &= \sigma^2\mathbb{E}_{\mathbf{\Phi}}[\operatorname{trace}(\mathbf{X}\mathbf{\Phi}_d^{\mathbf{X}}\mathbf{X}^T)] \\
&= \sigma^2\mathbb{E}_{\mathbf{\Phi}}[\operatorname{trace}(\mathbf{\Sigma}\mathbf{\Phi}_d^{\mathbf{X}})] \\
&= \sigma^2\mathbb{E}_{\mathbf{\Phi}}[\operatorname{trace}(I_{d\times d})] = \sigma^2 d.
\end{aligned}$$

Adding the first version to the expectation from above we get the exact expected mean squared error. Setting both versions equal we get the equation

$$d = \sum_{i=1}^{p}\frac{\lambda_i}{\eta_i}$$

□

Theorem 1.3.2. Assume Rank($\mathbf{X}$) $\geq d$, then there exists a real number $\tau \in [d^2/p, d]$ such that the AMSE of $\hat{\boldsymbol{\beta}}_d$ can be bounded from above by

$$\mathbb{E}_{\boldsymbol{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d\|_2^2]] \leq \sigma^2\tau + \sum_{i=1}^{p} \beta_i^2 \lambda_i w_i^2,$$

where the w_i's are given as

$$w_i = \frac{(1+1/d)\lambda_i^2 + (1+2/d)\lambda_i \operatorname{trace}(\boldsymbol{\Sigma}) + \operatorname{trace}(\boldsymbol{\Sigma})^2/d}{(d+2+1/d)\lambda_i^2 + 2(1+1/d)\lambda_i \operatorname{trace}(\boldsymbol{\Sigma}) + \operatorname{trace}(\boldsymbol{\Sigma})^2/d}$$

and

$$\tau \in [d^2/p, d].$$

Proof. First a simple calculation (Kabán, 2014) using the closed form solution gives the following equation:

$$\mathbb{E}_{\boldsymbol{\Phi}}[\mathbb{E}_{\boldsymbol{\varepsilon}}[\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\hat{\boldsymbol{\beta}}_d\|_2^2]] = \sigma^2 \sum_{i=1}^{p} \left(\frac{\lambda_i}{\eta_i}\right)^2 + \sum_{i=1}^{p} \beta_i^2 \lambda_i \left(1 - \frac{\lambda_i}{\eta_i}\right)^2. \quad (1.27)$$

Now using the corollary from the last section we can bound the second term the following way:

$$\left(1 - \frac{\lambda_i}{\eta_i}\right)^2 \leq w_i^2. \quad (1.28)$$

For the first term we write

$$\tau = \sum_{i=1}^{p} \left(\frac{\lambda_i}{\eta_i}\right)^2. \quad (1.29)$$

Now note that since $\lambda_i/\eta_i \leq 1$ we have

$$\left(\frac{\lambda_i}{\eta_i}\right)^2 \leq \frac{\lambda_i}{\eta_i} \quad (1.30)$$

and thus we get the upper bound by

$$\sum_{i=1}^{p} \left(\frac{\lambda_i}{\eta_i}\right)^2 \leq \sum_{i=1}^{p} \frac{\lambda_i}{\eta_i} = d. \quad (1.31)$$

For the lower bound of τ we consider an optimisation problem. Denote $t_i = \frac{\lambda_i}{\eta_i}$, then we want to find $t \in \mathbb{R}^p$ such that

$$\sum_{i=1}^{p} t_i^2 \text{ is minimal}$$

under the restrictions that

$$\sum_{i=1}^{p} t_i = d \text{ and } 0 \leq t_i \leq 1.$$

The problem is symmetric in each coordinate and thus $t_i = c$. Plugging this into the linear sum gives $c = d/p$ and we calculate the quadratic term to give the result claimed in the theorem. □

Chapter 2

The *xyz* algorithm for fast interaction search in high-dimensional data

When performing regression on a data set with p variables, it is often of interest to go beyond using main linear effects and include interactions as products between individual variables. For small-scale problems, these interactions can be computed explicitly but this leads to a computational complexity of at least $\mathcal{O}(p^2)$ if done naively. This cost can be prohibitive if p is very large.

We introduce a new randomised algorithm that is able to discover interactions with high probability and under mild conditions has a runtime that is subquadratic in p. We show that strong interactions can be discovered in almost linear time, whilst finding weaker interactions requires $\mathcal{O}(p^\alpha)$ operations for $1 < \alpha < 2$ depending on their strength. The underlying idea is to transform interaction search into a closest pair problem which can be solved efficiently in subquadratic time. The algorithm is called *xyz* and is implemented in the language `R`. We demonstrate its efficiency for application to genome-wide association studies, where more than 10^{11} interactions can be screened in under 280 seconds with a single-core 1.2 GHz CPU[1].

---

[1] This chapter has been published in (Thanei et al., 2016b)

2.1 Introduction

Given a response vector $\mathbf{Y} \in \mathbb{R}^n$ and matrix of associated predictors $\mathbf{X} = (\mathbf{X}_1, \dots, \mathbf{X}_p) \in \mathbb{R}^{n \times p}$, finding interactions is often of great interest as they may reveal important relationships and improve predictive power. When the number of variables p is large, fitting a model involving interactions can involve serious computational challenges. The simplest form of interaction search consists of screening for pairs (j, k) with high inner product between the outcome of interest $\mathbf{Y}$ and the point-wise product $\mathbf{X}_j \circ \mathbf{X}_k$:

$$\text{Keep all pairs } (j,k) \text{ for which } \mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)/n > \gamma. \tag{2.1}$$

This search is of complexity $\mathcal{O}(np^2)$ in a naive implementation and quickly becomes infeasible for large p. Of course one would typically be interested in maximising (absolute values of) correlations rather than dot products in (2.1), an optimisation problem that would be at least as computationally intensive.

Even more challenging is the task of fitting a linear regression model involving pairwise interactions:

$$Y_i = \mu + \sum_{j=1}^{p} X_{ij}\beta_j + \sum_{k=1}^{p}\sum_{j=1}^{k-1} X_{ij}X_{ik}\theta_{jk} + \varepsilon_i. \tag{2.2}$$

Here $\mu \in \mathbb{R}$ is the intercept and β_j and θ_{jk} contain coefficients for main effects and interactions respectively, and ε_i is random noise.

In this paper, we make several contributions to the problem of searching for interactions in high-dimensional settings.

(a) We first establish a form of equivalence between (2.1) and closest-pair problems (Agarwal et al., 1991; Shamos and Hoey, 1975). Assume for now that all predictors and outcomes are binary, so $X_{ij}, Y_i \in \{-1, 1\}$ (we will later relax this assumption) and define $\mathbf{Z} \in \{-1, 1\}^{n \times p}$ as $Z_{ij} = Y_i X_{ij}$. Then it is straightforward to show that (2.1) is equivalent to

$$\text{Keep all pairs } (j,k) \text{ for which } \|\mathbf{X}_j - \mathbf{Z}_k\|_2 < \gamma' \tag{2.3}$$

for some γ'. This connects the search for interactions to literature in computational geometry on problems of finding closest pairs of points.

(b) We introduce the *xyz* algorithm to solve (2.3) based on randomly projecting each of the columns in $\mathbf{X}$ and $\mathbf{Z}$ to a one-dimensional space. By exploiting the ability to sort the resulting $2p$ points with $\mathcal{O}(p\log(p))$ computational cost, we achieve a run time that is always subquadratic in p and can even reach a linear complexity $\mathcal{O}(np)$ when γ is much larger than the interaction strengths $|\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)|/n$ of the bulk of the pairs (j,k). We show that our approach can be viewed as an example of locality sensitive hashing (Leskovec et al., 2014) optimised for our specific problem.

(c) We show how any method for solving (2.1) can be used to fit regression models with interactions (2.15) by building it into an algorithm for the Lasso (Tibshirani, 1996). The use of *xyz* thus leads to a procedure for applying the Lasso to all main effects and interactions with computational cost that scales subquadratically in p.

(d) We provide implementations of both the core *xyz* algorithm and its extension to the Lasso in the R package `xyz`, which is available on github (Thanei, 2016) and CRAN.

Our work here is thus related to "closest pairs of points" algorithms in computational geometry as well as an extensive literature on modelling interactions in statistics, both of which we now review.

2.1.1 Related work

A common approach to avoid the quadratic cost in p of searching over all pairs of variables (2.1) is to restrict the search space: one can first seek a small number of important main effects, and then only consider interactions involving these discovered main effects. More specifically, one could fit a main effects Lasso (Tibshirani, 1996) to the data first, add interactions between selected main effects to the matrix of predictors, and then run the Lasso once more on the augmented design matrix in order to produce the final model (see Wu et al. (2010) for example). Tree-based methods such as CART (Breiman et al., 1984) work in a similar fashion by aiming to identify an important main effect and then only considering interactions involving this discovered effect.

However it is quite possible for the signal to be such that main effects corresponding to important interactions are hard to detect. As a concrete example of this phenomenon, consider the setting where $\mathbf{X}$ is generated randomly with all entries independent and having the uniform distribution on $\{-1, 1\}$. Suppose the response is given by $Y_i = X_{i1}X_{i2}$, so there is no noise. Since the distribution $Y_i|X_{ij}$ is the same for all k, main effects regressions would find it challenging to select variables 1 and 2. Note that by reparametrising the model by adding one to each entry of $\mathbf{X}$ for example, we obtain $Y_i = (X_{i1}-1)(X_{i2}-1) = 1-X_{i1}-X_{i2}+X_{i1}X_{i2}$. The model now respects the so-called strong hierarchical principle (Bien et al., 2013) that interactions are only present when their main effects are. The hierarchical principle is useful to impose on any fitted model. However, imposing the principle on the model does not imply that the interactions will easily be found by searching for main effects first. The difficulty of the example problem is due to interaction effects masking main effects: this is a property of the signal $\mathcal{E}(Y_i)$ and of course no reparametrisation can make the main effects any easier to find. Approaches that increase the set of interactions to be considered iteratively can help to tackle this sort of issue in practice (Bickel et al., 2010; Friedman, 1991; Hao and Zhang, 2014; Shah, 2016) as can those that randomise the search procedure (Breiman, 2001). However they cannot eliminate the problem of missing interactions, nor do these approaches offer guarantees of how likely it is that they discover an interaction.

As alluded to earlier, the pure interaction search problem (2.3) is related to close pairs of points problems, and more specifically the close bichromatic pairs problem in computational geometry (Agarwal et al., 1991). Most research in this area has focused on algorithms that lead to computationally optimal results in the number of points p whilst considering the dimension n to be constant. This has resulted in algorithms where the scaling of the computational complexity with n is at least of order 2^n (Shamos and Hoey, 1975). Since for meaningful statistical results one would typically require $n \gg \log(p)$, these approaches would not lead to subquadratic complexity. In the special case where $n = p$ and $Z_{ij}, X_{ij} \in \{-1, 1\}$, (2.3) may be seen to be equivalent to searching for large magnitude entries in the product of square matrices $\mathbf{X}$ and $\mathbf{Z}^T$. This latter problem is amenable to fast matrix multiplication algorithms, which in theory can deliver a subquadratic complexity of roughly $\mathcal{O}(p^{2.4}) = \mathcal{O}(np^{1.4})$ (Davie and Stothers, 2013; Le Gall, 2012; Williams, 2012). However the constants hidden in the order notation are typically very large, and practical implementations are unavailable. The Strassen algorithm (Strassen, 1969) is the only fast matrix multiplication algorithm used regularly in practice to the best of our knowledge. With a complexity of roughly $\mathcal{O}(p^{2.8}) = \mathcal{O}(np^{1.8})$, the improvement over a brute force close pairs search is only slight.

The strategy we use is most closely related to locality sensitive hashing (LSH) (Indyk and Motwani, 1998) which encompasses a family of hashing procedures such that similar points are mapped to the same bucket with high probability. A close pair search can then be conducted by searching among pairs mapped to the same bucket. In fact, our proposed algorithm for solving (2.3) can be thought of as an example of LSH optimised for our particular problem setting.

A seemingly attractive alternative to the subsampling-based LSH-strategy we employ is the method of Random Projections which is motivated by the theoretical guarantees offered by the Johnson–Lindenstrauss Lemma (Achlioptas, 2003). Perhaps surprisingly, we can show that using Random Projections instead of our subsampling-based scheme leads to a quadratic run time for interaction search (see Theorem 2.2.1 and section 2.5.1).

An approach that bears some similarity with our procedure is *epiq* (Arkin et al., 2014). This works by projecting the data and then searches through a lower dimensional representation for close pairs. This appears to improve upon a naive brute force empirically but there are no proven guarantees that the run time improves on the $\mathcal{O}(np^2)$ complexity of a naive search.

The *Random Intersection Trees* algorithm of Shah and Meinshausen (2014) searches for potentially deeper interactions in data with both $\mathbf{X}$ and $\mathbf{Y}$ binary. In certain cases with strong interactions a complexity close to linear in p is achieved; however it is not clear how to generalise the approach to continuous data or embed it within a regression procedure.

The idea of Kong et al. (2016) is to first transform the data by forming $\tilde{\mathbf{Y}} = \mathbf{Y} \circ \mathbf{Y}$ and $\tilde{\mathbf{X}}_j = \mathbf{X}_j \circ \mathbf{X}_j$ for each predictor. Next $\tilde{\mathbf{X}}_j$ and $\tilde{\mathbf{Y}}$ are tested for independence using the distance correlation test. In certain settings, this can reveal important interactions with a computational cost linear in p. However, the powers of these tests depend on the distributions of the transformed variables $\tilde{\mathbf{X}}_j$. For example in the binary case when $\mathbf{X} \in \{-1, 1\}^{n \times p}$, each transformed variable will be a vector of 1's and the independence tests will be unhelpful. We will see that our proposed approach works particularly well in this setting.

2.1.2 Organisation of the paper

In Section 2.2 we consider the case where both the response **Y** and the predictors **X** are binary. We first demonstrate how (2.15) may be converted to a form of closest pair of points problem. We then introduce a general version of the *xyz* algorithm which solves this based on random projections. As we show in Section 2.2.1 there is a particular random projection distribution that is optimal for our purposes. This leads to our final version of the *xyz* algorithm which we present in Section 2.2.3 along with an analysis of its run time and probabilistic guarantees that it recovers important interactions. In Section 2.3 we extend the *xyz* algorithm to continuous data. These ideas are then used in Section 2.4 to demonstrate how the *xyz* algorithm can be embedded within common algorithms for high-dimensional regression (Friedman et al., 2010) allowing high-dimensional regression models with interactions to be fitted with subquadratic complexity in p. Section 2.5 contains a variety of numerical experiments on real and simulated data that complement our theoretical results and demonstrate the effectiveness of our proposal in practice. We conclude with a brief discussion in Section 2.6 and all proofs are collected in the Appendix.

2.2 The *xyz* algorithm for binary data

In this section, we present a version of the *xyz* algorithm applicable in the special case where both **X** and **Y** are binary, so $X_{ij} \in \{-1, 1\}$ and $Y_i \in \{-1, 1\}$. We build up to the algorithm in stages, giving the final version in Section 2.2.2.

Define $\mathbf{Z} \in \{-1, 1\}^{n \times p}$ by $Z_{ij} = Y_i X_{ij}$ and

$$\gamma_{jk} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{Y_i = X_{ij} X_{ik}\}}. \tag{2.4}$$

We call γ_{jk} the interaction strength of the pair (j, k). It is easy to see that the interaction search problem (2.1) can be expressed in terms of either the γ_{jk} or the normalised squared distances. Indeed

$$2\gamma_{jk} - 1 = \mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)/n = \mathbf{Z}_j^T \mathbf{X}_k / n = 1 - \|\mathbf{Z}_j - \mathbf{X}_k\|_2^2/(2n). \tag{2.5}$$

Thus those pairs (j, k) with $Y^T(\mathbf{X}_j \circ \mathbf{X}_k)/n$ large will have γ_{jk} large, and $\|\mathbf{Z}_j - \mathbf{X}_k\|_2^2$ small.

This equivalence suggests that to solve (2.1), we can search for pairs (j, k) of columns $\mathbf{Z}_j, \mathbf{X}_k$ that are close in ℓ_2 distance. At first sight, this new problem would also appear to involve a search across all pairs, and would thus incur an $\mathcal{O}(np^2)$ cost. As mentioned in the introduction, close pair searches that avoid a quadratic cost in p incur typically an exponential cost in n. Since n would typically be much larger than $\log(p)$, such searches would be computationally infeasible.

We can however project each of the n-dimensional columns of $\mathbf{X}$ and $\mathbf{Z}$ to a lower dimensional space and then perform a close pairs search. The Johnson–Lindenstrauss Lemma, which states roughly that one can project p points into a space of dimension $\mathcal{O}(\log(p))$ and faithfully preserve distances, may appear particularly relevant here. The issue is that the projected dimension suggested by the Johnson–Lindenstrauss Lemma is still too large to allow for an efficient close pairs search. The following observation however gives some encouragement: if we had $Y = \mathbf{X}_j \circ \mathbf{X}_k$ so $\mathbf{X}_j = \mathbf{Z}_k$, even a one-dimensional projection $\mathbf{R} \in \mathbb{R}^n$ will have $|\mathbf{R}^T(\mathbf{X}_j - \mathbf{Z}_k)| = 0 = \|\mathbf{X}_j - \mathbf{Z}_k\|_2$, which implies that a perfect interaction will have zero distance in the projected space. We will later see that our approach leads to a linear run time in such a case. Importantly, we are only interested in using a projection that preserves the distances between the close pairs rather than all pairs, which makes our problem very different to the setting considered in the Johnson–Lindenstrauss Lemma.

With this in mind, consider the following general strategy. First project the columns of $\mathbf{X}$ and $\mathbf{Z}$ to one-dimensional vectors x and z using a random projection $\mathbf{R}$: $\mathbf{x} = \mathbf{X}^T\mathbf{R}$, $\mathbf{z} = \mathbf{Z}^T\mathbf{R}$. Next for some threshold τ, collect all pairs (j, k) such that $|x_j - z_k| \leq \tau$ in the set C. By first sorting $\mathbf{x}$ and $\mathbf{z}$, a step requiring only $\mathcal{O}(p\log(p))$ computations (see for example Sedgewick (1998)), this close pairs search can be shown to be very efficient. Given this set of candidate interactions, we can check for each $(j, k) \in C$ whether we have $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)/n > \gamma$. The process can be repeated L times with different random projections, and one would hope that given enough repetitions, any given strong interaction would be present in one of the candidate sets $E_1, \ldots, E_L$ with high probability. This approach is summarised in Algorithm 1 which we term the general form of the *xyz* algorithm. A schematic overview is given in Figure 2.1.

There are several parameters that must be selected, and a key choice to be made is the form of the random projection $\mathbf{R}$. For the joint distribution G of $\mathbf{R}$ we consider the following general class of distributions, which

Algorithm 1 A general form of the *xyz* algorithm.

```
Input: $\mathbf{X} \in \{-1,1\}^{n \times p}$, $\mathbf{Y} \in \{-1,1\}^n$
Parameters: $\xi = (G, L, \tau, \gamma)$. Here $G$ is the joint distribution for the projection vector $\mathbf{R}$, $L$ is the number of projections, and $\tau$ and $\gamma$ are the thresholds for close pairs and interactions strength respectively.
Output: $I$ set of strong interactions.
Form $\mathbf{Z}$ via $Z_{ij} = Y_i X_{ij}$ and set $I := \emptyset$.
for $l \in \{1, \dots, L\}$ do
    Draw random vector $\mathbf{R} \in \mathbb{R}^n$ with distribution $G$ and project the data using $\mathbf{R}$, to form
        $$\mathbf{x} = \mathbf{X}^T \mathbf{R} \text{ and } \mathbf{z} = \mathbf{Z}^T \mathbf{R}.$$
    Find all pairs $E_l$ for which $(j,k)$ such that $|x_j - z_k| \leq \tau$
    Add to $I$ those pairs in $E_l$ for which $|\mathbf{X}_j^T \mathbf{Z}_k|/n > \gamma$.
end for
```

includes both dense and sparse projections. We sample a random or deterministic number M of indices from the set $\{1, \dots, n\}$, $i_1, \dots, i_M$, either with or without replacement. Then, given a distribution $F \in \mathcal{F}$ where $\mathcal{F}$ is a class of distributions to be specified later, we form a vector $\mathbf{D} \in \mathbb{R}^M$ with independent components each distributed according to F. We then define the random projection vector $\mathbf{R}$ by

$$R_i = \sum_{m=1}^{M} D_m \mathbb{1}_{\{i_m = i\}}, \qquad i = 1, \dots, n. \tag{2.6}$$

Each configuration of the *xyz* algorithm is characterised by fixing the following parameters:

(i) G, a distribution for the projection vector R which is determined through (2.6) by $F \in \mathcal{F}$, a distribution for the subsample size M and whether sampling is with replacement or not;

(ii) $L \in \mathbb{N}$, the number of projection steps;

(iii) $\tau \geq 0$, the close pairs threshold;

(iv) $\gamma \in (0, 1)$, the interaction strength threshold.

We will denote the collection of all possible parameter levels by Ξ. This includes the following subclasses of interest. Fix $F \in \mathcal{F}$.

(a) **Dense projections**. Let $\mathbf{R} \in \mathbb{R}^n$ have independent components distributed according to F and denote the distribution of $\mathbf{R}$ by G. This falls within our general framework above with M set to n and sampling without replacement. Let

$$\Xi_{\text{dense}} := \{\xi \in \Xi \text{ with joint distribution equal to G}\}.$$

(b) **Subsampling**. Let $\mathcal{G}_{\text{subsample}}$ be the set of distributions for R obtained through (2.6) when subsampling with replacement. Let +

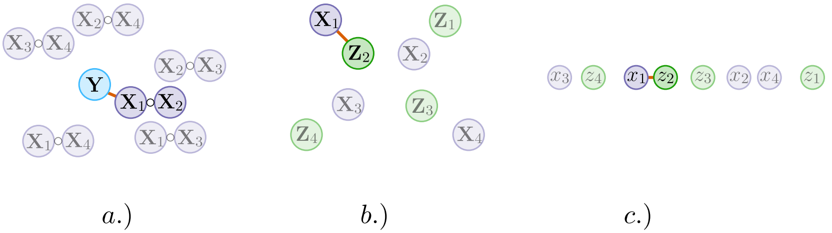
$$\Xi_{\text{subsample}} := \{\xi \in \Xi : \text{joint distribution } G \in \mathcal{G}_{\text{subsample}}\}.$$

(c) **Minimal subsampling**. Let Ξ_{minimal} be the set of all parameters in $\Xi_{\text{subsample}}$ such that the close pairs threshold is $\tau = 0$ and M takes randomly values in the set $\{m, m+1\}$ for some positive integer m.

$$\Xi_{\text{minimal}} := \{\xi \in \Xi_{\text{subsample}} \text{ with } \tau = 0 \text{ and } M \in \{m, m+1\} \text{ for some } m \in \mathbb{N}\}.$$

Note that we have suppressed the dependence of the classes above on the fixed distribution $F \in \mathcal{F}$ for notational simplicity. We define $\mathcal{F}$ to be the set of all univariate absolutely continuous and symmetric distributions with bounded density and finite third moment. The restriction to continuous distributions in $\mathcal{F}$ ensures that Ξ_{minimal} is invariant to the choice of F: when $\tau \equiv 0$, every $F \in \mathcal{F}$ with $L \in \mathbb{N}$ and the distribution for M fixed yields the same algorithm. Moreover the set of close pairs in C_l is simply the set of pairs (j, k) that have $X_{i_m j} = Z_{i_m k}$ for all $m = 1, \ldots, M$, that is the set of pairs that are equal on the subsampled rows. We note that the symmetry and boundedness of the densities in $\mathcal{F}$ and finiteness of the third moment are mainly technical conditions necessary for the theoretical developments in the following section. We will assume without loss of generality that the second moment is equal to 1. This condition places no additional restriction on Ξ since a different second moment may be absorbed into the choice of τ.

Minimal subsampling represents a very small subset of the much larger class of randomised algorithms outlined above. However, Theorem 2.2.1 below shows that minimal subsampling is essentially always at least as good as any algorithm from the wider class, which is perhaps surprising.



a.) *b.)* *c.)*

Figure 2.1: Illustration of the general *xyz* algorithm. The strongest interaction is the pair $(1, 2)$ and $p = 4$. Panel a.) illustrates the interaction search among $\mathbf{Y}$ and $\mathbf{X}_j \circ \mathbf{X}_k$, panel b.) shows the closest pair problem after the transformation $Z_{ij} = X_{ij}Y_i$ and panel c.) depicts the closest pair problem after the data has been projected. These are the three main steps in the *xyz* algorithm.

A beneficial consequence of this result is that we only need to search for the optimal ways of selecting M and L; the threshold τ is fixed at $\tau = 0$ and the choice of the continuous distribution F is inconsequential for minimal subsampling. The choices we give in Section 2.2.2 yield a subquadratic run time that approaches linear in p when the interactions to be discovered are much stronger than the bulk of the remaining interactions.

2.2.1 Optimality of minimal subsampling

In this section, we compare the run time of the algorithms in $\xi \in \Xi_{\text{dense}}$, $\Xi_{\text{subsample}}$ and Ξ_{minimal} that return strong interactions with high probability. Let (j^*, k^*) be the indices of a strongest interaction pair, that is $\gamma_{j^*k^*} = \max_{j,k \in \{1,\ldots,p\}} \gamma_{jk}$. We will consider algorithms ξ with γ set to $\gamma_{j^*k^*}$. Define the power of $\xi \in \Xi$ as

$$\text{Power}(\xi) := \mathbb{P}_\xi((j^*, k^*) \in I).$$

For $\eta \in (0, 1)$, let

$$\Xi_{\text{dense}}(\eta) = \{\xi \in \Xi_{\text{dense}} : \text{Power}(\xi) \geq \eta\},$$

and define $\Xi_{\text{subsample}}(\eta)$ and $\Xi_{\text{minimal}}(\eta)$ analogously. Note that these classes depend on the underlying $F \in \mathcal{F}$, which is considered to be fixed, and moreover that we are fixing $\gamma = \gamma_{j^*k^*}$. We consider an asymptotic regime where we have a sequence of response–predictor matrix pairs

$(\mathbf{Y}^{(n)}, \mathbf{X}^{(n)}) \in \mathbb{R}^n \times \mathbb{R}^{n \times p_n}$. Write $\gamma_{jk}^{(n)}$ for the corresponding interaction strengths, and let $\gamma_1^{(n)} = \max_{j,k} \gamma_{jk}^{(n)}$. Let $f_{\boldsymbol{\gamma}^{(n)}}$ be the probability mass function corresponding to drawing an element of $\boldsymbol{\gamma}^{(n)}$ uniformly at random. Note that $f_{\boldsymbol{\gamma}^{(n)}}$ has domain $\{0, 1/n, 2/n, \ldots, 1\}$. We make the following assumptions about the sequence of interaction strength matrices $\boldsymbol{\gamma}^{(n)}$.

(A1) There exists c_0 such that $|\{(j,k) : \gamma_{jk}^{(n)} = \gamma_1^{(n)}\}| \leq c_0 p_n$.

(A2) There exists $\gamma_l > 0$, $\gamma_u < 1$ such that $\gamma_u \geq \gamma_1^{(n)} \geq \gamma_l$ for all n.

(A3) There exists $\rho < 1$ such that $f_{\boldsymbol{\gamma}^{(n)}}$ is non-increasing on $[\rho\gamma_1^{(n)}, \gamma_1^{(n)}) \cap \{0, 1/n, \ldots, 1\}$.

Assumption (A1) is rather weak: typically one would expect the maximal strength interaction to be essentially unique, while (A1) requires that at most of order p_n interactions have maximal strength. (A2) requires the maximal interaction strength to be bounded away from 0 and 1, which is the region where complexity results for the search of interactions are of interest. As mentioned earlier, if the maximal interaction strength is 1, it will always be retained in the close-pair sets C_l, whilst if its strength is too close to 0, then it is near impossible to distinguish it from the remaining interactions. (A3) ensures a certain form of separation between maximal strength interactions and the bulk of the interactions.

To aid readability, in the following we suppress the dependence of quantities on n in the notation. Given $\mathbf{X}$ and $\mathbf{Y}$, we may define $T(\xi)$ as the expected number of computational operations performed by the algorithm corresponding to ξ. We have the following result.

Theorem 2.2.1. *Given $F \in \mathcal{F}$ and $\eta \in (0,1)$, there exists n_0 such that for all $n \geq n_0$ we have*

$$\inf_{\xi \in \Xi_{minimal}(\eta)} T(\xi) = \inf_{\xi \in \Xi_{subsample}(\eta)} T(\xi), \tag{2.7}$$

$$\inf_{\xi \in \Xi_{minimal}(\eta)} \frac{T(\xi)}{np^2} \to 0, \tag{2.8}$$

and there exists $c > 0$ such that

$$\inf_{\xi \in \Xi_{dense}(\eta)} \frac{T(\xi)}{np^2} > c. \tag{2.9}$$

The theorem shows that the optimal run time is achieved when using minimal subsampling. The last point is surprising: setting $\mathbf{R} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ will not improve the computational complexity over the brute-force approach and dense Gaussian projections hence do not reduce the complexity of the search.

2.2.2 The final version of xyz

The optimality properties of minimal subsampling presented in the previous section suggest the approach set out in Algorithm 2, which we will refer to as the *xyz* algorithm. Here we are using a simplified version of the minimal subsampling proposal given in the previous section where we keep M fixed rather than allowing it to be random. The reason is that the potential additional gain from allowing M to be any one of two consecutive numbers with certain probabilities is minimal but necessary for Theorem 2.2.1 and so the simpler approach is preferable. We note that the uniform distribution in line 3 may be replaced with any continuous distribution to yield identical results.

Algorithm 2 Final version of the *xyz* algorithm.

```
   Input: $\mathbf{X} \in \{-1,1\}^{n \times p}$, $\mathbf{Y} \in \{-1,1\}^{n}$, subsample size $M$, number of projections $L$, threshold for interaction strength $\gamma$.
   Output: $I$ set of strong interactions.
1: Form $\mathbf{Z}$ via $Z_{ij} = Y_i X_{ij}$.
2: for $l \in \{1, \dots, L\}$ do
3:     Form $\mathbf{R} \in \mathbb{R}^n$ as in (2.6) with distribution $F = U[0,1]$ and set $\mathbf{x} = \mathbf{X}^T\mathbf{R}$, $\mathbf{z} = \mathbf{Z}^T\mathbf{R}$.
4:     Find all pairs $(j,k)$ such that $x_j = z_k$ and store these in $E_l$.
5:     Add to $I$ those pairs in $E_l$ for which $|\mathbf{X}_j^T\mathbf{Z}_k|/n > \gamma$.
6: end for
```

To perform the equal pairs search in line 4, we sort the concatenation $(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^{2p}$ to determine the unique elements of $\{x_1, \dots, x_p, z_1, \dots, z_p\}$. At each of these locations, we can check if there are components from both x and z lying there, and if so record their indices. This procedure, which is illustrated in Figure 2.2, gives us the set of equal pairs E in the form of a union of Cartesian products. The computational cost is $\mathcal{O}(p\log(p))$. This complexity is driven by the cost of sorting whilst the recording of indices is linear in p. We note, however, that looping through

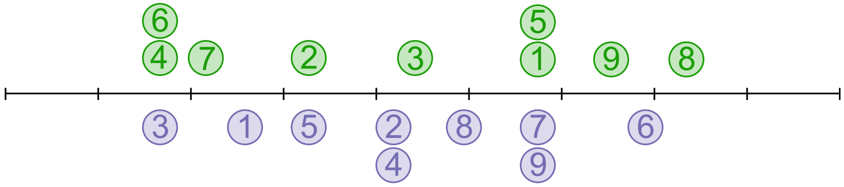


Figure 2.2: Illustration of an equal pairs search among components of $\mathbf{x}, \mathbf{z} \in \mathbb{R}^p$ when $p = 9$. The horizontal locations of blue and green circles numbered j give x_j and z_j respectively. Sorting of $(\mathbf{x}, \mathbf{z})$ allows traversal of the unique locations. At each of these it is checked whether points of both colours are present, and if so, the indices are recorded. Here the set of equal pairs $(\{3\} \times \{4, 6\}) \cup (\{5\} \times \{2\}) \cup (\{7, 9\} \times \{1, 5\})$ would be returned.

the set of equal pairs in order to output a list of close pairs of the form $(j_1, k_1), \ldots, (j_{|E|}, k_{|E|})$ would incur an additional cost of the size of E, though in typical usage we would have $|E| = o(p)$. Readers familiar with locality sensitive hashing (LSH) can find a short interpretation of equal pairs search as an LSH-family in the appendix. In the next section, we discuss in detail the impact of minimal subsampling on the complexity of the *xyz* algorithm and the discovery probability it attains.

2.2.3 Computational and statistical properties of xyz

We have the following upper bound on the expected number of computational operations performed by *xyz* (Algorithm 2) when the subsample size and number of repetitions are M and L:

$$C(M, L) := \underset{\text{(i)}}{np} + L\{\underset{\text{(ii)}}{Mp} + \underset{\text{(iii)}}{p\log(p)} + \underset{\text{(iv)}}{n\,\mathcal{E}_\xi(|E_1|)}\}. \tag{2.10}$$

The terms may be explained as follows: (i) construction of $\mathbf{Z}$; (ii) multiplying M subsampled rows of $\mathbf{X}$ and $\mathbf{Z}$ by $\mathbf{R} \in \mathbb{R}^n$; (iii) finding the equal pairs; (iv) checking whether the interactions exceed the interaction strength threshold γ. Note we have omitted a constant factor from the upper bound $C(M, L)$. There is a lower bound only differing from (2.10) in the equal pairs search term (iii), which is p instead of $p\log(p)$. It will be shown that (iv) is the dominating term and therefore the upper and lower bound are asymptotically equivalent, implying the bounds are tight.

An interaction with strength γ is retained in E_1 with probability γ^M.

Hence it is present in the final set of interactions I with probability

$$\eta(M, L) = 1 - (1 - \gamma^M)^L. \tag{2.11}$$

The following result demonstrates how the *xyz* algorithm can be used to find interactions whilst incurring only a subquadratic computational cost.

Theorem 2.2.2. *Let F_Γ be the distribution function corresponding to a random draw from the set of interaction strengths $\{\gamma_{jk}\}_{j,k\in\{1,\ldots,p\}}$. Given an interaction strength threshold γ, let $1 - F_\Gamma(\gamma) = c_1/p$. Define $\gamma_0 = p^{-1/M}$ and let c_2 be defined by $1 - F_\Gamma(\gamma_0) = c_2 p^{\log(\gamma)/\log(\gamma_0)-1}$. We assume that $\gamma_0 < \gamma$. Finally given a discovery threshold $\eta' \in [1/2, 1)$ let L be the minimal L' such that $\eta(M, L') \geq \eta'$. Ignoring constant factors we have*

$$C(M, L) \leq \log\{1/(1-\eta')\}(1 + c_1 + c_2)[\{1 + 1/\log(\gamma_0^{-1})\}\log(p) + n]p^{1+\log(\gamma)/\log(\gamma_0)}$$

If $n \gg \log(p)$ and γ_0 is bounded away from 1 we see that the dominant term in the above is

$$cnp^{1+\log(\gamma)/\log(\gamma_0)}, \tag{2.12}$$

where $c = \log\{1/(1-\eta')\}(1 + c_1 + c_2)$. Typically we would expect γ to be such that $|\{\gamma_{jk} : \gamma_{jk} > \gamma\}| \sim p$ as only the largest interactions would be of interest: thus we may think of c_1 as relatively small. If M is such that γ_0 is also larger than the bulk of the interactions, we would also expect c_2 to be small. Indeed, suppose that the proportion of interactions whose strengths are larger than γ_0 is $1 - F_\Gamma(\gamma_0) = c_1'/p$. Then $c_2 = c_1'/p^{\log(\gamma)/\log(\gamma_0)} < c_1'$. As a concrete example, if $\gamma = 0.9$ and M is such that $\gamma_0 = 0.55$, the exponent in (2.12) is around 1.17, which is significantly smaller than the exponent of 2 that a brute-force approach would incur; see also the examples in Section 2.5. Note also that when $\gamma = 1$, the exponent is 1 for all $\gamma_0 < 1$: if we are only interested in interactions whose strength is as large as possible, we have a run time that is linear in p.

It is interesting to compare our results here with the run times of approaches based on fast matrix multiplication. By computing $\mathbf{X}^T\mathbf{Z}$ we may solve the interaction search problem (2.1). Naive matrix multiplication would require $\mathcal{O}(np^2)$ operations, but there are faster alternatives when $n = p$. The fastest known algorithm (Williams, 2012) gives a theoretical run time of $\mathcal{O}(np^{1.37})$ when $n = p$. For *xyz* to achieve such a run time when $\gamma_0 = 0.55$ for example, the target interaction strength would have to be $\gamma \geq 0.81$: a somewhat moderate interaction strength. For $\gamma > 0.81$, *xyz* is strictly better; we also note that fast matrix multiplication algorithms tend to be unstable or lack a known implementation and are therefore rarely used in practice. A further advantage is that the *xyz* algorithm has an optimal memory usage of $\mathcal{O}(np)$.

We also note that whilst Theorem 2.2.2 concerns the the discovery of any single interaction with strength at least γ, the run time required to discover a fixed number interactions with strength at least γ would only differ by a multiplicative constant. If we however want a guarantee of discovering the p strongest pairs the bound in Theorem 2.2.2 would no longer hold.

To minimise the run time in (2.12), we would like γ_0 to be larger than most of the interactions in order that c_2 and hence c be small, yet a smaller γ_0 yields a more favourable exponent. Thus a careful choice of M, on which γ_0 depends, is required for *xyz* to enjoy good performance. In the following we show that an optimal choice of M exists, and we discuss how this M may be estimated based on the data.

Clearly if for some pair (M, L), we find another pair (M', L') with $\eta(M', L') > \eta(M, L)$ but $C(M', L') \leq C(M, L)$, we should always use (M', L') rather than (M, L). It turns out that there is in fact an optimal choice of M such that the parameter choice is not dominated by any others in this fashion. Define

$$M^* = \underset{M \in \mathcal{N}}{\arg\min} \left\{ -\frac{1}{\log(1-\gamma^M)} \left(Mp + p\log(p) + n \sum_{j,k} \gamma_{jk}^M \right) \right\}, \quad (2.13)$$

where it is implicitly assumed that the minimiser is unique. This will always be the case except for peculiar values of γ and γ_{jk}.

Proposition 2.2.3. *Let $L \in \mathcal{N}$. If $(M', L') \in \mathcal{N}^2$ has $\eta(M', L') \geq \eta(M^*, L)$, then also $C(M', L') \geq C(M^*, L)$ with the final inequality being strict if $M' \neq M^*$ and M^* is a unique minimiser.*

Thus there is a unique Pareto optimal M. Although the definition of M^* involves the moments of F_Γ, this can be estimated by sampling from $\{\gamma_{jk}\}$. We can then numerically optimise a plugin version of the objective to arrive at an approximately optimal M.

2.3 Interaction search on continuous data

In the previous section we demonstrated how the *xyz* algorithm can be used to efficiently solve the simplest form of interaction search (2.1) when both $\mathbf{X}$ and $\mathbf{Y}$ are binary. In this section we show how small modifications to the basic algorithm can allow it to do the same when $\mathbf{Y}$ is continuous, and also when $\mathbf{X}$ is continuous. We discuss the regression setting in Section 2.4.

2.3.1 Continuous Y and binary X

We begin by considering the setting where $\mathbf{X} \in \{-1, 1\}^{n\times p}$, but where we now allow real-valued $\mathbf{Y} \in \mathbb{R}^n$. Without loss of generality, we will assume $\|\mathbf{Y}\|_1 = 1$. The approach we take is motivated by the observation that the inner product $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$ can be interpreted as a weighted inner product of $\mathbf{X}_j \circ \mathbf{X}_k$ with the sign pattern of $\mathbf{Y}$, using weights $w_i = |Y_i|$.

With this in mind, we modify *xyz* in the following way. We set $\mathbf{Z}$ to be $Z_{ij} = \mathrm{sign}(Y_i)X_{ij}$. Let $i_1, \dots, i_M \in \{1, \dots, n\}$ be i.i.d. such that $\mathbb{P}(i_s = i) = w_i$. Forming the projection vector $\mathbf{R}$ using (2.6), we then find the probability of (j, k) being in the equal pairs set may be computed as follows.

$$
\begin{aligned}
\{\mathbb{P}(\mathbf{R}^T\mathbf{X}_j = \mathbf{R}^T\mathbf{Z}_k)\}^{1/M} &= \mathbb{P}(X_{i_1 j} = \mathrm{sign}(Y_{i_1})X_{i_1 k}) \\
&= \sum_{i=1}^{n} \mathbb{P}(X_{i_1 j} = \mathrm{sign}(Y_{i_1})X_{i_1 k} | i_1 = i)\mathbb{P}(i_1 = i) \\
&= \sum_{i=1}^{n} |Y_i| \mathbb{1}_{\{X_{ij} = \mathrm{sign}(Y_i)X_{ik}\}} \\
&= \sum_{i:\mathrm{sign}(Y_i) = X_{ij}X_{ik}} Y_i X_{ij} X_{ik} =: \tilde{\gamma}_{jk},
\end{aligned}
$$

where $\mathbb{P}$ here is with respect to the randomness of $\mathbf{R}$ (and, equivalently, the random indices $i_1, \dots, i_M$) with $\mathbf{Y}$ and $\mathbf{X}$ considered fixed. The calculation above shows that the run time bound of Theorem 2.2.2 continues to hold in the setting with continuous $\mathbf{Y}$ provided we replace the interaction strengths γ_{jk} with their continuous analogues $\tilde{\gamma}_{jk}$.

As a simple example, consider the model

$$
Y_i = X_{i1}X_{i2} + \varepsilon_i,
$$

with $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ and $\mathbf{X}$ generated randomly having each entry drawn independently from $\{-1, 1\}$ each with probability 1/2. Then for a non-interacting pair $j \neq 1,2$ or $k \neq 1,2$, we have $\tilde{\gamma}_{jk} \approx 0.5$. For the pair $(1,2)$ we calculate an interaction strength of

$$\begin{aligned}\tilde{\gamma}_{12} &= \mathbb{P}(\text{sign}(Y_{i_1}) = X_{i_1 1}X_{i_1 2}) = \mathbb{P}(\text{sign}(X_{i_1 1}X_{i_1 2} + \varepsilon_i) = X_{i_1 1}X_{i_1 2}) \\ &= \mathbb{P}(|\varepsilon_i| < 1) + \frac{1}{2}\mathbb{P}(|\varepsilon_i| > 1) = \frac{1}{2}(1 + \mathbb{P}(|\varepsilon_i| < 1)).\end{aligned}$$

A quick simulation gives the following table:

σ^2	0.1	0.25	0.5	1	2	5
$\tilde{\gamma}_{12}$	0.99	0.98	0.92	0.84	0.76	0.67

Using Theorem 2.2.2 and the above table we can estimate the computational complexity needed to discover the pair $(1,2)$ given a value of σ^2.

2.3.2 Continuous Y and continuous X

The previous section demonstrated how resampling with non-uniform weights transforms a setup with continuous $\mathbf{Y}$ into one with binary response. If both $\mathbf{X}$ and $\mathbf{Y}$ are continuous, we continue to use the previous strategy to deal with the continuous response. For the matrix $\mathbf{X}$ with continuous predictor values we cannot use weighted resampling as the weights would depend on the interaction pair of interest. In the following we examine the effects of transformations of $\mathbf{X}$ to a binary data matrix $\tilde{\mathbf{X}}$. To allow for randomized mappings, we define the transformations via a function $g : \mathbb{R} \mapsto [0,1]$ as

$$\mathbb{P}(\tilde{X}_{ij} = 1) = g(X_{ij}) \text{ and } 1 - \mathbb{P}(\tilde{X}_{ij} = -1) = 1 - g(X_{ij}),$$

where the transformation is always applied independently for each entry of the predictor matrix.

The following gives the probability of Y_i agreeing in sign with $\tilde{X}_{ij}\tilde{X}_{ik}$ when i is sampled with probability proportional to $|Y_i|$.

Proposition 2.3.1. *Given the transform* $\mathbb{P}(\tilde{X}_{ij} = 1) = g(X_{ij})$ *and sampling an index* i_s *according to* $\mathbb{P}(i_s = i) = Y_i/\|\mathbf{Y}\|_1$, *then the probability* *of a match is*

$$\mathbb{P}(\text{sign}(Y_{i_s}) = \tilde{X}_{i_s j}\tilde{X}_{i_s k}) = \frac{1}{2} + \frac{1}{2\|\mathbf{Y}\|_1}\sum_{i=1}^{n} Y_i(1-2g(X_{ij}))(1-2g(X_{ik})). \tag{2.14}$$

Thus we may define a continuous analogue of the interaction strength γ_{jk} based on the transform given by g as

$$\gamma_{jk}^{g} = \frac{1}{2} + \frac{1}{2\|\mathbf{Y}\|_1}\sum_{i=1}^{n} Y_i(1-2g(X_{ij}))(1-2g(X_{ik})).$$

These quantities may be substituted into Theorem 2.2.2 to yield the following upper bound on expected run time when using *xyz* on transformed data.

Corollary 2.3.2. *Let* F_{Γ^g} *be the distribution function corresponding to a random draw from the set of interaction strengths* $\{\gamma_{jk}^{g}\}_{j,k\in\{1,\ldots,p\}}$. *Given an interaction strength threshold* γ, *let* $1 - F_{\Gamma^g}(\gamma) = c_1/p$. *Define* $\gamma_0 = p^{-1/M}$ *and let* c_2 *be defined by* $1 - F_{\Gamma}(\gamma_0) = c_2 p^{\log(\gamma)/\log(\gamma_0)-1}$. *We assume that* $\gamma_0 < \gamma$. *Finally given a discovery threshold* $\eta' \in [1/2, 1)$ *let* L *be the minimal* L' *such that* $\eta(M, L') \geq \eta'$. *Ignoring constant factors we have*

$$\begin{aligned}&C(M,L)\\ &\leq \log\{1/(1-\eta')\}(1+c_1+c_2)[\{1+1/\log(\gamma_0^{-1})\}\log(p)+n]p^{1+\log(\gamma)/\log(\gamma_0)}\end{aligned}$$

The expected computational costs depends critically on the distribution of the interaction strengths F_{Γ^g}. To gain a better understanding of what impact different transformations have on this distribution and subsequently on run time we will study the following simple model for $(\mathbf{Y}, \mathbf{X}) \in \mathbb{R}^n \times \mathbb{R}^{n\times p}$:

$$Y_i = X_{ij^*}X_{ik^*} + \varepsilon_i, \qquad i = 1, \ldots, n, \tag{2.15}$$

where the ε_i are independent and have identical sub-exponential distributions symmetric about 0 and the rows of $\mathbf{X}$ are i.i.d. We now introduce two practically useful choices of g and study their properties in the context of model (2.15).

The unbiased transform

A natural choice for the transform g is one that satisfies the unbiasedness requirement:

$$\mathbb{E}(\tilde{X}_{ij}) = X_{ij}. \tag{2.16}$$

It turns out that this requirement uniquely defines the transform, which we refer to as the *unbiased transform.*

Proposition 2.3.3. *Let* $X_{ij} \in [-1, 1]$*. If its transformed version* $\tilde{X}_{ij}$ *satisfies* (2.16)*, then* g *takes the form*

$$\mathbb{P}(\tilde{X}_{ij} = 1) = g(X_{ij}) = \frac{X_{ij} + 1}{2}.$$

Furthermore the interaction strength in (2.14) *is given by*

$$\mathbb{P}(\text{sign}(Y_{i_s}) = \tilde{X}_{i_s j}\tilde{X}_{i_s k}) = \gamma^g_{jk} = \frac{1}{2} + \frac{1}{2\|\mathbf{Y}\|_1}\sum_{i=1}^{n} Y_i X_{ij} X_{ik}.$$

Proposition 2.3.3 shows that γ^g_{jk} is a monotone function of the inner product

$$\sum_{i=1}^{n} Y_i X_{ij} X_{ik}$$

We remark that if the entries of $\mathbf{X}$ do not lie in $[-1, 1]$, we may divide each entry in the ith row by $\nu_i := \max_j |X_{ij}|$, and multiply Y_i by ν_i^2, for each i. Proposition 2.3.3 will then hold for the scaled versions of $\mathbf{Y}$ and $\mathbf{X}$. In order to describe the performance of the unbiased transform when applied to data generated by the model (2.15), we define the following quantities:

$$\mathbb{E}(|X_{ij^*} X_{ik^*}|) = m_1, \;\; \mathbb{E}(X^2_{ij^*} X^2_{ik^*}) = m_2 \;\text{ and } \mathbb{E}(|\varepsilon_i|) = m_\varepsilon.$$

We consider an asymptotic regime where $p = p_n$ may diverge as n tends to infinity, though we suppress this in the notation. We introduce the following assumptions.

(B1) $m_2(r_u - 1) \leq \mathbb{E}(X_{ij^*} X_{ik*} X_{ij} X_{ik}) \leq m_2(1 - r_u)$, for $r_u \in (0, 1)$ and $\forall\, j, k \in \{1, \ldots, p\}^2$.

(B2) The noise level satisfies the bound

$$\frac{1}{1 - r_u} > 1 + \frac{m_\epsilon}{m_1}.$$

(B3) Let p be such that be such that

$$\frac{\log(n)\log(p)}{n} \overset{n\to\infty}{\to} 0.$$

(B1) ensures non-interactions are not too strongly correlated to the actual interaction pair (j^*, k^*). Note that (B3) allows for high-dimensional settings with $p \gg n$.

Theorem 2.3.4. *Assume all entries of* $\mathbf{X}$ *have mean zero and lie in* $[-1, 1]$ *almost surely. Further assume (B1)–(B3) hold. When* M *and* L *are as in Corollary 2.3.2 and the unbiased transform is used, we have*

$$C(M, L) = o_{\mathbb{P}}\left(np^{1+\delta+\frac{\log(1/2+m_2/2(m_1+m_\varepsilon))}{\log(1/2+m_2(1-r_u)/2m_1)}}\right)$$

for any $\delta > 0$*. Here* $\mathbb{P}$ *is with respect to the randomness in* $\mathbf{X}$ *and* $\boldsymbol{\varepsilon}$*.*

Though the run time above can often improve significantly on the worst-case quadratic run time, observe that unlike in the binary case, if there is no noise and $Y_i = X_{ij^*}X_{ik^*}$, we do not necessarily have a run time close to linear in p. For example, when $X_{ij} \overset{iid}{\sim} \text{Uniform}(-1, 1)$, the interaction strength of the true interaction can be shown to equal to

$$\gamma^g_{j^*k^*} = \frac{1}{2} + \frac{\sum_{i=1}^n Y_i X_{ij^*} X_{ik^*}}{2\|\mathbf{Y}\|_1} = \frac{1}{2} + \frac{\|\mathbf{Y}\|_2^2}{2\|\mathbf{Y}\|_1} \overset{n\to\infty}{=} \frac{13}{18}.$$

Substituting this into the run time given by Theorem 2.2.2, this would result in an expected complexity of roughly $\mathcal{O}(np^{1.47})$; this is still substantially smaller than a quadratic run time, but raises the question as to whether such a loss in speed is avoidable.

Additionally, if $\mathbf{X}$ has several outlying entries, normalising the design matrix by scaling by the row-wise maximums can shrink $\gamma^g_{j^*k^*}$ towards $1/2$. To limit the impact of this normalisation, we can first cap the entries of $\mathbf{X}$ so their absolute value is bounded by some $c > 0$. Though the resulting interaction strength will not have the form given in Proposition 2.3.3, it may better discriminate between interactions of interest and noise.

Capping with $c = 1$ is closely related to applying the sign transform, which we study next.

The sign transform

We now consider the *sign transform* given by $\tilde{X}_{ij} = \text{sign}(X_{ij})$; if there are zero cases we use a coin toss to map them to $\{-1, 1\}$. For the sign transform we have $g(X_{ij}) = 2\text{sign}(X_{ij}) - 1$ and so the interaction strength is given as:

$$\mathbb{P}(\text{sign}(Y_{i_s}) = \tilde{X}_{i_s j}\tilde{X}_{i_s k}) = \gamma^g_{jk} = \frac{1}{2} + \frac{1}{2\|\mathbf{Y}\|_1}\sum_{i=1}^{n} Y_i \text{sign}(X_{ij})\text{sign}(X_{ik}).$$

The sign transform recovers the close to linear run time achieved in the binary case when a interaction is perfect as now if $Y_i = X_{ij^*}X_{ik^*}$, we have $\gamma^g_{j^*k^*} = 1$. Also the sign transform is not adversely affected by the presence of outlying entries in $\mathbf{X}$, and for our theory we can relax the assumption that the entries of $\mathbf{X}$ are in $[-1, 1]$ to here only requiring that they have a subexponential distribution. To facilitate comparison with the unbiased transform, we impose assumptions analogous to (B1)–(B3):

(C1) $r_s/2 \leq \mathbb{P}(X_{ij} < 0 | X_{ik}, X_{ij^*}, X_{ik^*}) \leq 1 - r_s/2$, for $r_s \in (0, 1)$ and $\forall\, j, k \in \{1, ..., p\}^2$.

(C2) The noise level satisfies

$$\frac{1}{1 - r_s} > 1 + \frac{m_\epsilon}{m_1}.$$

(C3) Let p be such that

$$\frac{\log(p)^5}{n} \overset{n\to\infty}{\to} 0.$$

Theorem 2.3.5. *Suppose that each entry of* $\mathbf{X}$ *has a mean-zero subexponential distribution. Further assume (C1)–(C3). When* M *and* L *are as in Corollary 2.3.2 and the sign transform is used, we have*

$$C(M, L) = o_{\mathbb{P}}\left(np^{1+\delta+\frac{\log(1/2 + m_1/2(m_1+m_\varepsilon))}{\log(1-r_s)}}\right)$$

for any $\delta > 0$. *Here* $\mathbb{P}$ *is with respect to the randomness in* $\mathbf{X}$ *and* ε.

Both transforms yield a run time of the form $o_{\mathbb{P}}(np^\alpha)$. Comparing the exponents α we have:

unbiased transform:

$$\alpha_u = 1 + \frac{\log(1/2 + m_2/2(m_1 + m_\varepsilon))}{\log(1/2 + m_2(1 - r_u)/2m_1)}$$

sign transform:

$$\alpha_s = 1 + \frac{\log(1/2 + m_1/2(m_1 + m_\varepsilon))}{\log(1/2 + (1 - r_s)/2)}.$$

For bounded data $\mathbf{X} \in [-1, 1]^{n \times p}$ and when $m_\varepsilon \ll m_1$, we have $m_1/2(m_1 + m_\varepsilon) \approx 1/2$ so that $\alpha_s = 1$ whereas $\alpha_u > 1$. Hence in case of a strong signal the sign transform can give a smaller run time than the unbiased transform.

2.4 Application to Lasso regression

Thus far we have only considered the simple version of the interaction search problem (2.1) involving finding pairs of variables whose interaction has a large dot product with $\mathbf{Y}$. In this section we show how any solution to this, and in particular the *xyz* algorithm, may be used to fit the Lasso (Tibshirani, 1996) to all main effects and pairwise interactions in an efficient fashion.

Given a response $\mathbf{Y} \in \mathbb{R}^n$ and a matrix of predictors $\mathbf{X} \in \mathbb{R}^{n \times p}$, let $\mathbf{W} \in \mathbb{R}^{n \times p(p+1)/2}$ be the matrix of interactions defined by

$$\mathbf{W} = (\mathbf{X}_1 \circ \mathbf{X}_1, \mathbf{X}_1 \circ \mathbf{X}_2, \cdots, \mathbf{X}_1 \circ \mathbf{X}_p, \mathbf{X}_2 \circ \mathbf{X}_2, \mathbf{X}_2 \circ \mathbf{X}_3, \cdots, \mathbf{X}_p \circ \mathbf{X}_p).$$

We will assume that $\mathbf{Y}$ and the columns of $\mathbf{X}$ have been centred. Note that the centring of $\mathbf{X}$ means the $\mathbf{W}$ implicitly contains main effects terms. Let $\tilde{\mathbf{W}}$ be a version of $\mathbf{W}$ with centred columns. Consider the Lasso objective function

$$(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}) = \underset{\boldsymbol{\beta} \in \mathbb{R}^p, \boldsymbol{\theta} \in \mathbb{R}^{p(p+1)/2}}{\operatorname{argmin}} \left\{ \frac{1}{2n} \|\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \tilde{\mathbf{W}}\boldsymbol{\theta}\|_2^2 + \lambda(\|\boldsymbol{\beta}\|_1 + \|\boldsymbol{\theta}\|_1) \right\}. \tag{2.17}$$

Note that since the entire design matrix in the above is column-centred, any intercept term would always be zero.

In order to avoid a cost of $\mathcal{O}(np^2)$ it is necessary to avoid explicitly computing $\mathbf{W}$. To describe our approach, we first review in Algorithm 3 the active set strategy employed by several of the fastest Lasso solvers such as `glmnet` (Friedman et al., 2010). We use the notation that for a matrix $\mathbf{M}$ and a set of column indices H, $\mathbf{M}_H$ is the submatrix of $\mathbf{M}$ formed from those columns indexed by H. Similarly for a vector $\mathbf{v}$ and component indices H, $\mathbf{v}_H$ is the subvector of $\mathbf{v}$ formed from the components of $\mathbf{v}$ indexed by H.

Algorithm 3 Active set strategy for Lasso computation

```
   Input: $\mathbf{X}$, $\mathbf{Y}$ and grid of $\lambda$ values $\lambda_1 > \cdots > \lambda_L$.
   Output: Lasso solutions $\hat{\boldsymbol{\beta}}_{\lambda_l}$ and $\hat{\boldsymbol{\theta}}_{\lambda_l}$ at each $\lambda$ on the grid.
1: for $l \in \{1, \ldots, L\}$ do
2:     If $l = 1$ set $A, B = \emptyset$; otherwise set $A = \{k : \hat{\beta}_{\lambda_{l-1},k} \neq 0\}$ and $B = \{k : \hat{\theta}_{\lambda_{l-1},k} \neq 0\}$.
3:     Compute the Lasso solution $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}})$ when $\lambda = \lambda_l$ under the additional constraint that $\hat{\boldsymbol{\beta}}_{A^c} = 0$ and $\hat{\boldsymbol{\theta}}_{B^c} = 0$.
4:     Let $U = \{k : |\mathbf{X}_k^T(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)|/n > \lambda_l\}$ and $V = \{k : |\tilde{\mathbf{W}}_k^T(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)|/n > \lambda_l\}$ be the set of coordinates that violate the KKT conditions when $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}})$ is taken as a candidate solution.
5:     If $U$ and $V$ are empty, we set $\hat{\boldsymbol{\beta}}_{\lambda_l} = \hat{\boldsymbol{\beta}}$, $\hat{\boldsymbol{\theta}}_{\lambda_l} = \hat{\boldsymbol{\theta}}$. Else we update $A = A \cup U$ and $B = B \cup V$ and return to line 3.
6: end for
```

As the sets A and B would be small, computation of the Lasso solution in line 3 is not too expensive. Instead line 4, which performs a check of the Karush–Kuhn–Tucker (KKT) conditions involving dot products of all interaction terms and the residuals, is the computational bottleneck: a naive approach would incur a cost of $\mathcal{O}(np^2)$ at this stage.

There is however a clear similarity between the KKT conditions check for the interactions and the simple interaction search problem (2.1). Indeed the computation of V, the set containing all interactions that violate the KKT conditions, may be expressed in the following way:

$$\text{Keep all pairs } (j,k) \text{ for which } |(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)^T(\mathbf{X}_j \circ \mathbf{X}_k)/n| > \lambda_l. \tag{2.18}$$

Note that since $\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B$ is necessarily centered, there is no need to center the interactions in (2.18). In order to solve (2.18) we can use the *xyz* algorithm, setting γ in Algorithm 2 to λ_l and $\mathbf{Y}$ to each of $\pm(\mathbf{Y} - \mathbf{X}_A\hat{\boldsymbol{\beta}}_A - \tilde{\mathbf{W}}_B\hat{\boldsymbol{\theta}}_B)$ in turn.

Precisely the same strategy of performing KKT condition checks using *xyz* can be used to accelerate computation for interaction modeling for a variety of variants of the Lasso such as the elastic net (Zou and Hastie, 2005) and ℓ_1-penalised generalised linear models. Note also that it is straightforward to use a different scaling for the penalty on the interaction coefficients in (2.17), which may be helpful in practice.

2.5 Experiments

To test the algorithm and theory developed in the previous sections, we run a sequence of experiments on real and simulated data.

2.5.1 Comparison of minimal subsampling and dense projections

One of the surprising outcomes of our theoretical analysis is extent of the suboptimality of Gaussian random projections, which whilst they suffice for the conclusion of the Johnson–Lindenstrauss Lemma, are not well-suited for our purposes here (see Theorem 2.2.1). We can explicitly compute the probability of retaining an interaction of strength γ in E_1 for both dense Gaussian projections ξ_{Gauss} and minimal subsampling $\xi_{minimal}$ given an equal computational budget. We consider various values of p ranging from 10 up to 10^6 and we fix $n = 1000$. We set $L = 1$ and select other parameters of the algorithms to ensure the average size of E_1 is equal to p in the setting when all interaction strengths are equal to 0.5. Specifically we make the following choices.

- ξ_{Gauss}: the close pairs threshold $\tau \geq 0$ is the $1/p$–quantile of the distribution of $|W|$ when $W \sim N(0, 0.5n)$.
- $\xi_{minimal}$: the subsample size $M = \lceil \log(1/p)/\log(0.5) \rceil$.

We then plot the probability η of discovering an interaction of strength γ, as a function of γ for different values of p (Figure 2.3). For $\xi_{minimal}$, η is given in equation (2.11). For ξ_{Gauss}, η is the $1/p$–quantile of the distribution of $|W|$ when $W \sim N(0, n(1-\gamma))$.

2.5.2 Scaling

In this experiment we test how the *xyz* algorithm scales on a simple test example as we increase the dimension p. We generate data $\mathbf{X} \in \mathbb{R}^{n \times p}$ with each entry sampled independently uniformly from $\{-1, 1\}$. We do this for different values of p, ranging from 1000 to 30 000: this way for the largest p considered there are more than 400 million possible interactions. Then for each $\mathbf{X}$ we construct response vectors $\mathbf{Y}$ such that only the pair $(1, 2)$ is a strong interaction with an interaction strength taking values in $\{0.7, 0.8, 0.9\}$. Through this construction, if n is large enough, all the pairs except $(1, 2)$ will have an interaction strength around 0.5, and very few will have one above 0.55. We thus set M so that $\gamma_0 = p^{-1/M} \approx 0.55$. Since the only strong interaction is $(1, 2)$, we set $\gamma = \gamma_{12}$ Each data set configuration determined by p and γ_{12} is simulated 300 times and we measure the time it takes *xyz* to find the pair $(1, 2)$. In Figure 2.3 we plot the average run time against the dimension p with the different choices for γ_{12} highlighted in different colours.

Theorem 2.2.2 indicates that the run time should be of the order $np^{1+\log(\gamma)/\log(\gamma_0)}$. We see that the experimental results here are in close agreement with this prediction.

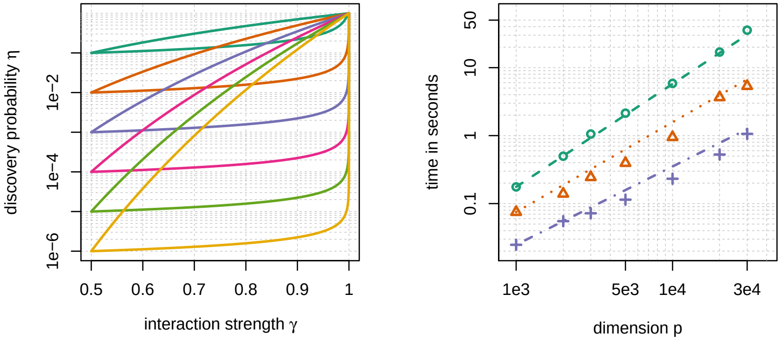


Figure 2.3: Left panel: Discovery probability as a function of γ for different values of $p \in \{10^1, \ldots, 10^6\}$ (colours decreasing in p from yellow $p = 10^6$ to green $p = 10$). The lower lines correspond to the dense Gaussian projections, the upper lines to minimal subsampling. It can be seen that the discovery probability for minimal subsampling is much higher (up to factor 10^4) than for Gaussian projections. Right panel: Time to discover the interaction pair as a function of the data set dimension p. Lines correspond to the theoretical prediction and symbols give the actual measured run time. Colour coding: green $\gamma = 0.7$, orange $\gamma = 0.8$ and purple $\gamma = 0.9$.

2.5.3 Run on SNP data

In the next experiment we compare the performance of *xyz* to its closest competitors on a real data set. For each method we measure the time it takes to discover strong interactions. We consider the LURIC data set (Winkelmann et al., 2001), which contains data of patients that were hospitalised for coronary angiography. We use a preprocessed version of the data set that is made up of $n = 859$ observations and 687 253 predictors. The data set is binary. The response $\mathbf{Y}$ indicates coronary disease (1 corresponding to affected and -1 healthy) and $\mathbf{X}$ contains Single Nucleotide Polymorphisms (SNPs) which are variations of base-pairs on DNA. The response vector $\mathbf{Y}$ is strongly unbalanced: there are 681 affected cases ($Y_i = 1$) and 178 unaffected ($Y_i = -1$). For a fair comparison among the classes -1 and 1 we generate a subsample of the data set with equal class distributions. We repeat the subsampling a few times and pick the interactions that consistently appear over many subsamples.

To get a contrast of the performance of *xyz* we compare it to *epiq* (Arkin et al., 2014), another method for fast high-dimensional interaction search. In order for *epiq* to detect interactions it needs to assume the model

$$Y_i = \alpha_{j^*k^*} X_{ij^*} X_{ik^*} + \varepsilon_i, \tag{2.19}$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. It then searches for interactions by considering the test statistics

$$T_{jk} = (\mathbf{R}^T(\mathbf{Y} \circ \mathbf{X}_j))(\mathbf{R}^T \mathbf{X}_k)$$

where $\mathbf{R} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. These are used to try to find the pair (j^*, k^*), which is assumed to be the pair for which the inner product $\mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$ is maximal. It is an easy calculation to show that $\mathbb{E}(T_{jk}) = \mathbf{Y}^T(\mathbf{X}_j \circ \mathbf{X}_k)$. To maximise the inner product on the right, *epiq* considers pairs where T_{jk}^2 is large by looking at pairs where both $(\mathbf{R}^T(\mathbf{Y} \circ \mathbf{X}_j))^2$ and $(\mathbf{R}^T \mathbf{X}_k)^2$ are large. While the approach of *epiq* is somewhat related to *xyz*, there are no bounds available for the time it takes to find strong interactions.

We also compare both methods to a naive approach where we subsample a fixed number of interactions uniformly at random, and retain the strongest one. We refer to this as *naive search.*

At fixed time intervals we check for the strongest interaction found so far with all three methods. We plot the interaction strength as a function of the computational time (Figure 2.4). All three methods eventually discover interactions of very similar strength and it would be a hasty judgement to say whether one significantly outperforms the others. *xyz* nevertheless discovers the strongest interactions on average for a fixed run time compared to the other two approaches. To get a clearer picture we run two additional experiments on a slight modification of the LURIC data set. We implant artificial interactions where we set the strength to $\gamma_{12} = 0.8$ and another example with $\gamma_{12} = 0.9$. In these two experiments *xyz* clearly outperforms all other methods considered (Figure 2.4; panels 3 and 4). Besides *xyz* being the fastest at interaction search, it also offers a probabilistic guarantee that there are no strong interactions left in the data. This guarantee comes out of Theorem 2.2.2. To run *xyz* we have to calculate the optimal subsample size (2.13) for use of minimal subsampling:

$$M^* = \operatorname*{arg\,min}_{M \in \mathcal{N}} \left\{ -\frac{1}{\log(1-\gamma^M)} \left(Mp + p\log(p) + n\sum_{j,k} \gamma_{jk}^M \right) \right\} = 21.$$

The sum in this optimisation can be approximated by uniformly sampling over pairs. Assume we have an interaction pair (j^*, k^*) with interaction strength $\gamma_{j^*k^*} = 0.85$ and say the rest of the pairs (j,k) have an interaction strength of no more than $\gamma_{jk} \leq 0.55$. The probability that we discover this pair in one run ($L = 1$) of the *xyz* algorithm is $\gamma_{j^*k^*}^{21}$. Therefore the probability of missing this pair after $L = 100$ runs is given by

$$(1 - \gamma_{j^*k^*}^{21})^L \approx 0.03.$$

Note that the number of possible interactions is $p(p-1)/2 \approx 10^{11}$. The whole search took 280 seconds. Naive search offers a similar guarantee, however it is extremely weak. The probability of not discovering the pair after drawing pL samples (with $L = 100$) is bounded by $[1 - 2/\{p(p-1)\}]^{Lp} \approx 0.999$. If we consider the run time guarantee from Theorem 2.2.2, the dominating term in the complexity of *xyz* in terms of p is

$$p^{1+\frac{\log(0.85)}{\log(0.55)}} \approx p^{1.27}.$$

This may be compared to the expected run time of order p^2 for naive search, which means that *xyz* is about 30 000 times faster than naive search (when $p = 687\,253$). In the empirical comparison this factor is around 20 000.

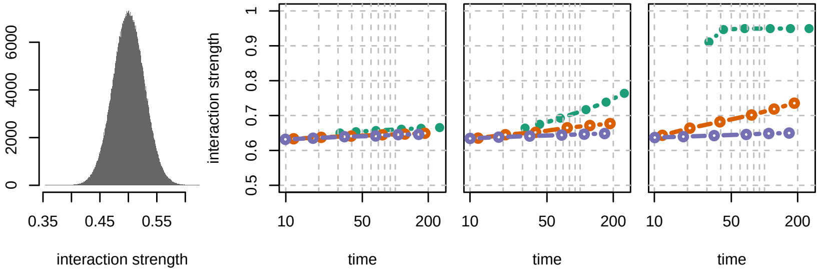


Figure 2.4: Left: Histogram of interaction strength of 10^6 interaction pairs, sampled at random from the more than 10^{11} existing pairs from the LURIC data set. The right three panels show the interaction strength of the discovered pairs as a function of the computation time for *xyz* (green), *epiq* (orange) and naive search (purple). The first panel gives results on the the original LURIC data set, and the second and third (rightmost) panels show results with an implanted interaction with strengths $\gamma_{12} = 0.8$ and $\gamma_{12} = 0.95$ respectively. It can be clearly seen that *xyz* outperforms its competitors by a large margin.

2.5.4 Regression on artificial data

In this section we demonstrate the capabilities of *xyz* in interaction search for continuous data as explained in Section 2.3. We simulate two different models of the form (2.15):

$$Y_i = \mu + \sum_{j=1}^{p} X_{ij}\beta_j + \sum_{k=1}^{p}\sum_{j=1}^{k-1} X_{ij}X_{ik}\theta_{jk} + \varepsilon_i.$$

We consider three settings. For all three settings we have $n = 1000$. We let $p \in \{250, 500, 750,$
$1000\}$. Each row of $\mathbf{X}$ is generated i.i.d. as $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. The magnitudes of both the main and interaction effects are chosen uniformly from the interval $[2, 6]$ (20 main effects and 10 interaction effects) and we set $\varepsilon_i \sim \mathcal{N}(0, 1)$. The three settings we consider are as follows.

1. $\mathbf{\Sigma} = \mathbf{I} \in \mathbb{R}^{p \times p}$, we generate a hierarchical model: $\theta_{jk} \neq 0 \Rightarrow \beta_j \neq 0$ and $\beta_k \neq 0$. We first sample the main effects and then pick interaction effects uniformly from the pairs of main effects.

2. $\mathbf{\Sigma} = \mathbf{I} \in \mathbb{R}^{p \times p}$, we generate a strictly non-hierarchical model: $\theta_{jk} \neq 0 \Rightarrow \beta_j = 0$ and $\beta_k = 0$. We first sample the main effects and

then pick interaction effects uniformly from all pairs excluding main effects as coordinates.

3. We repeat the setting 2 with a data set that contains strong correlations. We create a dependence structure in $\mathbf{X}$, by first generating a DAG with on average 10 edges per node. Each node is sampled so that it is a linear function of its parents plus some independent centred Gaussian noise, with a variance of 10% the variance coming from the direct parents. The resulting correlation matrix then unveils for each variable $\mathbf{X}_j$ a substantial number of variables strongly correlated to $\mathbf{X}_j$ (There is usually around 10 variables with a correlation of above 0.9). Such a correlation structure will make it easier to detect pairs of variables whose product can serve as strong predictor of $\mathbf{Y}$, even though it has not been included in the construction of $\mathbf{Y}$.

We run three different procedures to estimate the main and interaction effects.

- **Two-stage Lasso:** We fit the Lasso to the data, and then run the Lasso once more on an augmented design matrix containing interactions between all selected main effects. Complexity analysis of the Least Angle Regression (LARS) algorithm (Efron et al., 2004) suggests the computational cost would be $\mathcal{O}(np\min(n,p))$, making the procedure very efficient. However, as the results show, it struggles in situations such as that given by model 2, where a main effects regression will fail to select variables involved in strong interactions.

- **Lasso with all interactions:** Building the full interaction matrix and computing the standard Lasso on this augmented data matrix. Analysis of the LARS algorithm would suggest the computational complexity would be in the order $\mathcal{O}(np^2\min(n,p^2))$. Nevertheless, for small p, this approach is feasible.

- **xyz:** This is Algorithm 3; we set the parameter L to be $\sqrt{p}$ in order to target the strong interactions.

The experiment (seen in Figure 2.5) shows that *xyz* enjoys the favourable properties of both its competitors: it is as fast as the two-stage Lasso that gives an almost linear run time in p, and it is about as accurate as the estimator calculated from screening all pairs (brute-force).

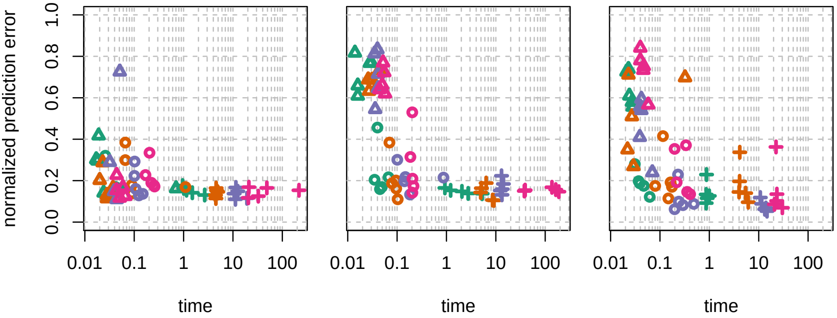


Figure 2.5: Normalised ℓ_2^2 prediction error as a function of time in seconds. Triangle: Two-stage Lasso. Circle: *xyz*-regression. Cross: Brute-force. The different colours correspond to different values of p: green $p = 250$, orange $p = 500$, purple $p = 750$ and pink $p = 1000$. The left panel shows the results on setting 1, center panel shows setting 2 and right panel setting 3.

2.5.5 Regression on real data

Here we run *xyz* regression on continuous real data sets where the ground truth is unknown. On each data set we pick at random $p = 2000$ variables and run *xyz* and the Lasso implemented in `glmnet` with all interactions included. We subsample an increasing number of variables to vary the difficulty of the regression problem. For each sample we measure the run time and the normalized out of sample squared ℓ_2^2 error:

$$\frac{\|\mathbf{Y}_{\text{test}} - \mathbf{X}_{\text{test}}\hat{\boldsymbol{\beta}} - \tilde{\mathbf{W}}_{\text{test}}\hat{\boldsymbol{\theta}}\|_2^2}{\|\mathbf{Y}_{\text{test}}\|_2^2}.$$

Experiments are run on the following three different data sets:

- **Riboflavin:** The Riboflavin production data set (Bühlmann et al., 2014) contains $n = 71$ samples and $p = 4088$ predictors (gene-expressions). The response $\mathbf{Y}$ and the design $\mathbf{X}$ are both continuous.

- **Kemmeren:** The Kemmeren (Kemmeren and al., 2014) data set records knockouts of $p = 6170$ genes. The data $\mathbf{X}$ is continuous. We sample $\mathbf{Y}$ randomly from the genes not present in the subsample taken from $\mathbf{X}$.

- **Climate:** The climate data set from the CNRM model from the CMIP5 model ensemble (Knutti et al., 2013) simulates the temperature of points on the northern hemisphere which is recorded in $\mathbf{X}$. The response $\mathbf{Y}$ simulates the temperature on a random position on the southern hemisphere. The data contains $n = 231$ observations.

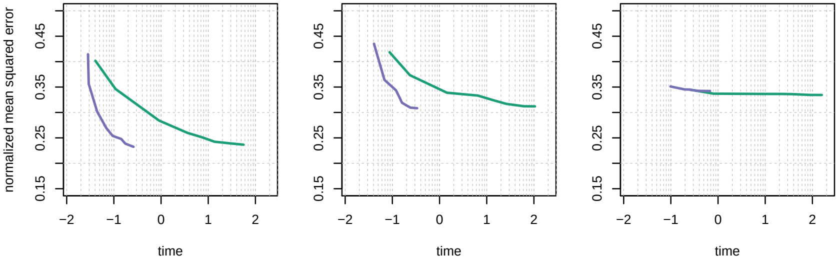


Figure 2.6: From left to right column the experiments correspond to Riboflavin, Kemmeren and Climate. The y-axis depicts the normalized squared error and the x-axis records the run time in seconds on the $\log_{10}$ scale. It can be seen that *xyz* (purple) offers clear computational advantages while giving similar level of prediction error to the Lasso fitted to all interactions as implemented in `glmnet` (green).

For each experiment we fix the number of runs L to $\sqrt{p}$ so the run time of *xyz* is $\mathcal{O}(np^{1.5})$. The experiments show that the *xyz* algorithm has a similar prediction performance to the Lasso applied to all interactions as implemented in `glmnet`. However *xyz* is around 100 times faster for $p = 2000$. The results of all 6 experiments can be seen in Figure 2.6.

2.6 Discussion

In this work we exploited a relationship between closest pairs of point problems and interaction search. By solving the former problem using random projections to project points down to a one-dimensional space and then sorting the resulting projected points, we were able to produce an algorithm for interaction search that enjoys a run time that is sub quadratic under mild assumptions and when used to search for very strong interactions can be almost linear. Though we have looked at interaction search in this paper, the basic engine for computing the large inner products between collections of vectors may have other interesting applications, for example in large-scale clustering problems. We hope to study such applications in future work.

2.7 Appendix

Table 2.1: Table of frequently used notation

n, p	number of samples and number of variables
$\mathbf{X}, \mathbf{Y}$	predictor matrix and response vector
$\mathbf{X}_j$	j-th variable/columns of X
$\boldsymbol{\beta}, \boldsymbol{\theta}$	coefficients of main effects and interaction effects
γ_{jk}	interaction strength of the pair (j, k)
G	distribution of projection
M	subsample size
$\mathbf{R}$	projection vector
L	number of projections, repetitions of the *xyz* algorithm
τ, γ	close pairs threshold and interaction strength threshold
Ξ	set of all configurations of the *xyz* algorithm, the elements of this set are denoted by ξ
η	probability that interaction is present in the output of the *xyz* algorithm
$\tilde{\mathbf{X}}$	binarized version of $\mathbf{X}$
$\mathbf{W}$	predictor matrix containing all possible interaction pairs

Appendix A.

Here we include proofs that were omitted earlier.

Proof of Theorem 2.2.1

In the following, we fix the following notation for convenience:

$$\Psi = \Xi_{\text{minimal}}, \qquad \Psi(\eta) = \Xi_{\text{minimal}}(\eta),$$
$$\Xi = \Xi_{\text{subsample}}, \qquad \Xi(\eta) = \Xi_{\text{subsample}}(\eta).$$

Note that both $\Psi(\eta)$ and $\Xi(\eta)$ depend on F though this is suppressed in the notation. Also define $\Xi_{\text{all}} = \Xi \cup \Xi_{\text{dense}}$ and $\Xi_{\text{all}}(\eta) = \Xi(\eta) \cup \Xi_{\text{dense}}(\eta)$.

We will reference the parameters levels contained in $\xi \in \Xi_{\text{all}}$ as ξ_L and ξ_τ. If $\xi \in \Xi$ then we will write ξ_M for the distribution of the subsample size M.

If we let V denote the complexity of the search for τ-close pairs, similarly to (2.10) we have that

$$T(\xi) = c_1 np + L(c_2\, \mathcal{E}_\xi\, Mp + \mathcal{E}_\xi\, V + c_3 n\, \mathcal{E}_\xi\, |E_1|), \tag{2.20}$$

where c_1, c_2, c_3 are constants. Suppose $\psi \in \Psi$ and $\xi \in \Xi$ have $\mathcal{E}_\xi\, |E_1| = \mathcal{E}_\psi\, |E_1|$. Then since searching for τ-close pairs is at least as computationally difficult as finding equal pairs we know that $\mathcal{E}_\xi\, V \geq \mathcal{E}_\psi\, V$.

Similarly for $\xi \in \Xi_{\text{dense}}$ we have

$$T(\xi) = c_1 np + L(c_2 np + \mathcal{E}_\xi\, V + c_3 n\, \mathcal{E}_\xi\, |E_1|). \tag{2.21}$$

For $\xi \in \Xi_{\text{all}}$, define

$$\alpha(\xi) = \mathcal{E}_\xi\, |E_1|/p^2, \qquad \beta(\xi) = \mathbb{P}_\xi((j^*, k^*) \in I_1)$$

where I_1 is the set of candidate interactions I when $L = 1$. Note that

$$\mathbb{P}_\xi((j^*, k^*) \in I) = 1 - \{1 - \beta(\xi)\}^{\xi_L}.$$

Thus any $\xi \in \Xi_{\text{all}}(\eta)$ with $T(\xi)$ minimal must have ξ_L as the smallest L such that $1 - \{1 - \beta(\xi)\}^{\xi_L} \geq \eta$, whence

$$\xi_L = \lceil \log(1 - \eta) / \log\{1 - \beta(\xi)\} \rceil. \tag{2.22}$$

Note that $\beta(\xi)$ does not depend on ξ_L, so the above equation completely determines the optimal choice of L once other parameters have been fixed. We will therefore henceforth assume that L has been chosen this way so that the discovery probability of all the algorithms is at least η.

The proofs of (2.8) and (2.9) are contained in Lemmas 2.7.4 and 2.7.5 respectively. The proof of (2.7) is more involved and proceeds by establishing a Neyman–Pearson type lemma (Lemmas 2.7.2 and 2.7.3) showing that given a constraint on the 'size' α that is sufficiently small, minimal subsampling enjoys maximal 'power' β. To complete the argument, we show that any sequence of algorithms with size α remaining constant as $p \to \infty$ cannot have a subquadratic complexity, whilst Lemma 2.7.4 attests that in contrast minimal subsampling does have subquadratic complexity under the assumptions of the theorem. Several auxiliary technical lemmas are collected in Section 2.7

Our proofs Lemmas 2.7.2 and 2.7.3 make use of the following bound on a quantity related to the ratio of the size to the power of minimal subsampling.

Lemma 2.7.1. *Suppose $\psi \in \Psi$ has distribution for M placing mass on M and $M+1$. Under the assumptions of Theorem 2.2.1,*

$$\frac{\alpha(\psi)}{\gamma_1^M} \leq \frac{2}{1-\rho}\frac{1}{M+1}.$$

Proof. We have

$$\frac{\alpha(\psi)}{\gamma_1^M} \leq \frac{1}{p^2}\sum_{j,k}(\gamma_{jk}/\gamma_1)^M \leq \frac{c_0}{p} + \sum_{i=0}^{n\gamma_1 - 1}\left(\frac{i}{n\gamma_1}\right)^M f_n(i/n).$$

Now the sum on the RHS is maximised over f_n obeying constraints (A1) and (A2) in the following way. If $\rho\gamma_1 n > \gamma_1 n - 1$ then f_n places all available mass on $\gamma_1 - 1/n$. Otherwise f_n should be as close to constant as possible on $\lceil \rho\gamma_1 n \rceil /n, \ldots, (\gamma_1 n - 1)/n$, and zero below $\lceil \rho\gamma_1 n \rceil /n$. In both cases it can be seen that

$$\sum_{i=0}^{n\gamma_1 - 1}\left(\frac{i}{n\gamma_1}\right)^M f_n(i/n) \leq \frac{2}{1-\rho}\int_{(1+\rho)/2}^{1} x^M dx \leq \frac{2}{1-\rho}\frac{1}{M+1}.$$

□

The following Neyman–Pearson-type lemma considers only non-randomised algorithms in Ξ. In Lemma 2.7.3 we extend this result to randomised algorithms.

Lemma 2.7.2. *Let Ξ_0 be the set of $\xi \in \Xi$ such that ξ_M places mass only on a single M, so the subsample size is not randomised. There exists an α_0 independent of n such that for all $\alpha' \leq \alpha_0$, we have*

$$\sup_{\psi \in \Psi : \alpha(\psi) \leq \alpha'} \beta(\psi) = \sup_{\xi \in \Xi_0 : \alpha(\xi) \leq \alpha'} \beta(\xi).$$

Moreover the suprema are achieved.

Proof. Each $\xi \in \Xi_0$ is parametrised by its close pairs threshold τ and subsample size M. Given a $\xi \in \Xi_0$ with parameter values τ and M we compute $\alpha(\xi)$ as follows. Note that by replacing the threshold τ by

$\tau/2$, we may assume that $\mathbf{X}$ and $\mathbf{Z}$ have entries in $\{-1/2, 1/2\}$. Thus $\mathbf{X}_j - \mathbf{Z}_k$ has components in $\{-1, 0, 1\}$. Let J_{jk} be the number of non-zero components of $(X_{i_m j} - Z_{i_m k})_{m=1}^M$. Then $J_{jk} \sim \text{Binom}(M, 1-\gamma_{jk})$. Thus

$$\begin{aligned}\mathbb{P}\Big(\Big|\sum_{m=1}^M D_m(X_{i_m j} - Z_{i_m k})\Big| \leq \tau\Big) &= \mathbb{P}(J_{jk} = 0) \\ &\quad + \sum_{r=1}^M \mathbb{P}\Big(\Big|\sum_{m=1}^r D_m\Big| \leq \tau\Big)\mathbb{P}(J_{jk} = r),\end{aligned}$$

noting that $D_m \overset{d}{=} -D_m$. By Lemma 2.7.6 we know there exists an $a > 0$ such that for all $\tau \leq a\sqrt{M}$ the RHS is bounded below by

$$\gamma_{jk}^M + \sum_{r=r_0}^M \frac{c_1\tau}{\sqrt{r}}\binom{M}{r}\gamma_{jk}^{M-r}(1-\gamma_{jk})^r \tag{2.23}$$

for M sufficiently large. Here the constants $a, c_1 > 0$ and $r_0 \in \mathcal{N}$ depend only on F.

Consider $\tau > a\sqrt{M}$. In this case, for $r \leq M$ sufficiently large we have by Lemma 2.7.6

$$\mathbb{P}\Big(\Big|\sum_{m=1}^r D_m\Big| \leq \tau\Big) \geq \mathbb{P}\Big(\Big|\sum_{m=1}^r D_m\Big| \leq a\sqrt{r}\Big) \geq c_1 a.$$

However then for M sufficiently large,

$$\mathbb{P}(J_{jk} = 0) + \sum_{r=1}^M \mathbb{P}\Big(\Big|\sum_{m=1}^r D_m\Big| \leq \tau\Big)\mathbb{P}(J_{jk} = r) \geq c_1 a/2,$$

so $\alpha(\xi) \geq c_1 a/2$. Note also that we must have $\alpha_0 \geq \alpha(\xi) \geq \gamma_l^M$, so $M \geq \log(\alpha_0)/\log(\gamma_l)$. Thus by choosing $0 < \alpha_0 < c_1 a/2$ sufficiently small, we can rule out $\tau > a\sqrt{M}$ and so we henceforth assume that $\tau \leq a\sqrt{M}$, and that M is sufficiently large such that (2.23) holds for all (j, k).

We have

$$\alpha(\xi) \geq \frac{1}{p^2}\sum_{j,k}\Big\{\gamma_{jk}^M + \tau\sum_{r=r_0}^M \frac{c_1}{\sqrt{r}}\binom{M}{r}\gamma_{jk}^{M-r}(1-\gamma_{jk})^r\Big\}. \tag{2.24}$$

Similarly we have

$$\beta(\xi) \leq \gamma_1^M + \tau \sum_{r=1}^{M} \frac{c_2}{\sqrt{r}} \binom{M}{r} \gamma_1^{M-r}(1-\gamma_1)^r. \tag{2.25}$$

Now substituting the upper bound on τ implied by (2.24) into (2.25), we get

$$\beta(\xi) \leq \gamma_1^M + Q_M\biggl(\alpha(\xi) - \frac{1}{p^2}\sum_{j,k}\gamma_{jk}^M\biggr)$$

where

$$Q_M = \frac{c_2 \sum_{r=1}^M r^{-1/2}\binom{M}{r}\gamma_1^{M-r}(1-\gamma_1)^r}{c_1 p^{-2}\sum_{j,k}\sum_{r=r_0} r^{-1/2}\binom{M}{r}\gamma_{jk}^{M-r}(1-\gamma_{jk})^r}.$$

Now by Lemma 2.7.7, for M sufficiently large and some constant Q we have

$$Q_M \leq Q\frac{\sqrt{1-\gamma_1}}{\sum_{j,k}\sqrt{1-\gamma_{jk}}/p^2} \leq Q.$$

Thus

$$\beta(\xi) \leq \gamma_1^M + Q\biggl(\alpha(\xi) - \frac{1}{p^2}\sum_{j,k}\gamma_{jk}^M\biggr) \tag{2.26}$$

for all M sufficiently large. Now given α_0, let M_0 be such that

$$\frac{1}{p^2}\sum_{j,k}\gamma_{jk}^{M_0} \geq \alpha_0 \geq \frac{1}{p^2}\sum_{j,k}\gamma_{jk}^{M_0+1}.$$

Consider the minimal subsampling algorithm ψ that chooses subsample size as either M_0 or M_0+1 with probabilities b and $1-b$ such that

$$\alpha(\psi) = \frac{1}{p^2}\sum_{j,k}\{b\gamma_{jk}^{M_0} + (1-b)\gamma_{jk}^{M_0+1}\} = \alpha_0.$$

Then we have $\beta(\psi) = b\gamma_1^{M_0} + (1-b)\gamma_1^{M_0+1}$. Now suppose $\xi \in \Xi_0$ has $\alpha(\xi) \leq \alpha_0$. Then in particular $M \geq M_0 + 1$. We first examine the case where $M = M_0 + 1$. Then

$$\begin{aligned}\frac{1}{\gamma_1^{M_0}}\{\beta(\psi) - \beta(\xi)\} &\geq b + (1-b)\gamma_1 - \gamma_1 - \frac{Q}{\gamma_1^{M_0}}\left(\alpha_0 - \frac{1}{p^2}\sum_{j,k}\gamma_{j,k}^{M_0+1}\right) \\ &= b + (1-b)\gamma_1 - \gamma_1 - \frac{aQ}{\gamma_1^{M_0}}\frac{1}{p^2}\sum_{j,k}(\gamma_{j,k}^{M_0} - \gamma_{j,k}^{M_0+1}) \\ &\geq b\biggl((1-\gamma_u) - \frac{2Q}{1-\rho}\frac{1}{M_0+1}\biggr),\end{aligned}$$

using Lemma 2.7.1 in the final line. Note this is non-negative for M_0 sufficiently large. When $M \geq M_0 + 2$ we instead have

$$\frac{\beta(\xi)}{\beta(\psi)} \leq \frac{\beta(\xi)}{\gamma_1^{M_0+1}} \leq \gamma_1 + \frac{2Q}{\gamma_1(1-\rho)}\frac{1}{M_0+1} \leq \gamma_u + \frac{2Q}{\gamma_l(1-\rho)}\frac{1}{M_0+1} < 1$$

for M_0 sufficiently large. Recall that by making α_0 sufficiently small, we can force M_0 to be arbitrarily large. Thus the result is proved. □

Lemma 2.7.3. *There exists an α_0 independent of n such that for all $\alpha' \leq \alpha_0$, we have*

$$\sup_{\psi\in\Psi:\alpha(\psi)\leq\alpha'} \beta(\psi) = \sup_{\xi\in\Xi:\alpha(\xi)\leq\alpha'} \beta(\xi).$$

Moreover the suprema are achieved.

Proof. With a slight abuse of notation, write $\xi(M', \tau')$ for the element of $\xi \in \Xi$ that fixes $M = M'$ and $\tau = \tau'$. Using the notation of Lemma 2.7.2, define function $f : [0, 1] \to [0, 1]$ by

$$f(\alpha') = \sup_{\xi\in\Xi_0:\,\alpha(\xi)\leq\alpha'} \beta(\xi).$$

Note that for $\xi \in \Xi$ we have

$$\beta(\xi) \leq \mathcal{E}_{M\sim\xi_M}\, f[\alpha\{\xi(M, \xi_\tau)\}]. \tag{2.27}$$

Now by Lemma 2.7.2 we know there exists α_0 (depending on F) such that on $[0, \alpha_0]$, f is the linear interpolation of points

$$\left(\frac{1}{p^2}\sum_{j,k}\gamma_{j,k}^M,\, \gamma_1^M\right)_{M=1}^{\infty}.$$

We claim that f is concave on $[0, \alpha_0]$. Indeed, it suffices to show that the slopes of the successive linear interpolants are decreasing in this region, or equivalently that their reciprocals are increasing. We have

$$\frac{1}{p^2} \sum_{j,k} \frac{\gamma_{jk}^{M+1} - \gamma_{jk}^{M}}{\gamma_1^{M+1} - \gamma_1^{M}} = \frac{1}{p^2} \sum_{j,k} \left(\frac{\gamma_{j,k}}{\gamma_1} \right)^M \frac{\gamma_{jk} - 1}{\gamma_1 - 1} \tag{2.28}$$

which increases as M decreases, thus proving the claim.

Note also that the RHS of (2.28) is at most $\alpha(\psi)/\{(1-\gamma_u)\gamma_1^M\}$ when ψ has subsample size fixed at M. Thus by Lemma 2.7.1 we see the derivatives of the linear interpolants approach infinity as they get closer to the origin. This implies the existence of an $0 < \alpha_1 < \alpha_0$ such that $-\sup\big(\partial(-f)(\alpha_1)\big) \geq \{1 - f(\alpha_1)\}/(\alpha_0 - \alpha_1)$, where $\partial(-f)(\alpha_1)$ denotes the subdifferential of the function $-f$ at α_1. We may therefore invoke Lemma 2.7.8 to conclude that for ξ with $\alpha(\xi) \leq \alpha_1$

$$\begin{aligned} \mathcal{E}_{M \sim \xi_M} f[\alpha\{\xi(M, \xi_\tau)\}] &\leq f[\mathcal{E}_{M \sim \xi_M} \alpha\{\xi(M, \xi_\tau)\}] \\ &= f(\alpha(\xi)) \leq f(\alpha_1) = \max_{\psi \in \Psi : \alpha(\psi) \leq \alpha_1} \beta(\psi). \end{aligned}$$

Combining with (2.27) gives the result. □

The next lemma establishes subquadratic complexity of minimal subsampling.

Lemma 2.7.4. *Under the assumptions of Theorem 2.2.1, we have*

$$\inf_{\psi \in \Psi(\eta)} T(\psi)/(np^2) \to 0$$

Proof. Let $\psi \in \Psi$ be such that ψ_M places all mass on M. We have that $\beta(\psi) = \gamma_1^M$. Thus using the inequality $-x \leq \log(1-x)$ for $x \in (0,1)$, we have

$$\psi_L \leq -\gamma_1^{-M} \log(1-\eta).$$

Lemma 2.7.1 gives an upper bound on $\psi_L \, \mathcal{E}_\psi \, E_1$. Note that $\mathcal{E}_\psi \, V = \mathcal{O}(p \log(p))$. Thus ignoring constant factors, we have

$$T(\psi)/(np^2) \leq \frac{M + \log(p)}{\gamma_1^M np} + \frac{1}{M+1}.$$

Taking $M = \lfloor \log(1/\sqrt{p})/\log(\gamma_1) \rfloor$ then ensures $T(\psi)/(np^2) \to 0$. □

Lemma 2.7.5. *Let* $\xi \in \Xi_{dense}$*. There exists* $c > 0$ *and* $n_0 \in \mathcal{N}$ *such that for all* $n \geq n_0$*,*

$$\inf_{\xi \in \Xi_{dense}} T(\xi)/(np^2) > c.$$

Proof. Each $\xi \in \Xi_{\mathrm{dense}}$ is parametrised by its close pairs threshold τ. Given a $\xi \in \Xi_{\mathrm{dense}}(F)$ with close pairs threshold τ we compute $\alpha(\xi)$ as follows. Similarly to Lemma 2.7.2 we may assume without loss of generality that $\mathbf{X}$ and $\mathbf{Z}$ have entries in $\{-1/2, 1/2\}$ so $\mathbf{X}_j - \mathbf{Z}_k$ has components in $\{-1, 0, 1\}$. Since $R_i \overset{d}{=} -R_i$ as $F \in \mathcal{F}$, we have

$$\mathbb{P}\left(\left| \sum_{i=1}^{n} R_i(X_{ij} - Z_{ik}) \right| \leq \tau \right) = \mathbb{P}\left(\left| \sum_{i=1}^{n(1-\gamma_{jk})} R_i \right| \leq \tau \right).$$

We now use Lemma 2.7.6. For $n(1-\gamma_u)$ sufficiently large, when $\tau \leq a\sqrt{n}$ the RHS is bounded below by

$$\frac{c_1 \tau}{\sqrt{n(1-\gamma_{jk})}}.$$

Here constant $a, c_1 > 0$ also depend only on F. Thus

$$\alpha(\xi) \geq \frac{1}{p^2} \sum_{j,k} \frac{c_1 \tau}{\sqrt{n(1-\gamma_{jk})}} \geq c_1 \tau / \sqrt{n}. \tag{2.29}$$

Similarly we have

$$\beta(\xi) \leq \frac{c_2 \tau}{\sqrt{n(1-\gamma_1)}}. \tag{2.30}$$

Note that from (2.29), when $\tau > a\sqrt{n}$ we have $\alpha(\xi) \geq c_1 a$. Thus from (2.21) we know there exists n_0 such that for all $n \geq n_0$, we have

$$\inf_{\xi \in \Xi_{\mathrm{dense}}(\eta): \xi_\tau > a\sqrt{n}} T(\xi)/(np^2) \geq \inf_{\xi \in \Xi_{\mathrm{dense}}(\eta): \xi_\tau > a\sqrt{n}} \xi_L \alpha(\xi) \geq \xi_L c_1 a > 0. \tag{2.31}$$

We therefore need only consider the case where $\tau \leq a\sqrt{n}$ and where $\alpha(\xi) \to 0$.

Substituting the upper bound on τ implied by (2.29) into (2.30), we get

$$\beta(\xi) \leq \alpha(\xi) \frac{c_2}{c_1 \sqrt{1-\gamma_u}}.$$

Note that then

$$\xi_L \geq \frac{\log(1-\eta)}{\log\{1-\alpha(\xi)c_2/(c_1\sqrt{1-\gamma_u})\}} \geq c_3 \frac{\log\big(1/1-\eta\big)}{\alpha(\xi)}$$

for some $c_3 > 0$ provided $\alpha(\xi) < 1/2$ say. However this gives us

$$\begin{aligned}\inf_{\xi\in\Xi_{\text{dense}}(\eta):\xi_\tau\leq a\sqrt{n}} T(\xi)/(np^2) &\geq \inf_{\xi\in\Xi_{\text{dense}}(\eta):\xi_\tau\leq a\sqrt{n}} \xi_L\alpha(\xi) \\ &\geq \min\{1/2, c_3\log\big(1/1-\eta\big)\} > 0.\end{aligned}$$

Combined with (2.31) this give the result. □

With the previous lemmas in place, we are in a position to prove (2.7) of Theorem 2.2.1.

Proof of Theorem 2.2.1

The proofs of (2.8) and (2.9) are contained in Lemmas 2.7.4 and 2.7.5 respectively. To show (2.7) we argue as follows. Given F and η, suppose for contradiction that there exists a sequence $\xi^{(1)}, \xi^{(2)}, \ldots$ and $n_1 < n_2 < \cdots$ such that (making the dependence on n of the computational time explicit)

$$\inf_{\psi\in\Psi(\eta)} T^{(n_k)}(\psi) > T^{(n_k)}(\xi^{(k)})$$

for all k. By Lemma 2.7.4, we must have $T^{(n_k)}(\xi^{(k)})/(np^2) \to 0$. This implies that $\alpha(\xi^{(k)}) \to 0$. By Lemma 2.7.3, we know that for k sufficiently large

$$\sup_{\psi\in\Psi:\alpha(\psi)=\alpha(\xi^{(k)})} \beta(\psi) \geq \beta(\xi^{(k)}).$$

Let $\psi^{(k)}$ be the maximiser of the LHS. In order for $T^{(n_k)}(\psi^{(k)}) > T^{(n_k)}(\xi^{(k)})$, it must be the case that $\mathcal{E}_{M\sim\psi_M^{(k)}} M > \mathcal{E}_{M\sim\xi_M^{(k)}} M$. However we claim that $\xi = \psi^{(k)}$ minimises $\mathcal{E}_{M\sim\xi_M} M$ among all $\xi \in \Xi$ with $\alpha(\xi) \leq \alpha(\xi^{(k)}) =: \alpha_0$, which gives a contradiction and completes the proof. Let f be the function that linearly interpolates the points

$$\left(\frac{1}{p^2}\sum_{j,k}\gamma_{j,k}^M,\, M\right)_{M=1}^{\infty}.$$

Note that f is decreasing. By considering the inverse of f it is clear that f is convex. With a slight abuse of notation, write $\xi(M,\tau)$ for the element of $\xi \in \Xi$ such that ξ_M places all mass on M and $\xi_\tau = \tau$. Note that

$$\mathcal{E}_{M \sim \xi_M} M = \mathcal{E}_{M \sim \xi_M} f[\alpha\{\xi(M,0)\}] \geq \mathcal{E}_{M \sim \xi_M} f[\alpha\{\xi(M,\xi_\tau)\}].$$

Now suppose ξ has $\alpha(\xi) \leq \alpha_0$. Then from the above and Jensen's inequality,

$$\mathcal{E}_{M \sim \xi_M} M \geq f\big(\mathcal{E}_{M \sim \xi_M} \alpha(\xi(M,\xi_\tau))\big) \geq f(\alpha_0) = \mathcal{E}_{M \sim \psi_M^{(k)}} M.$$

Proof of Theorem 2.2.2

First note that from (2.11) we have $L \leq \log(1-\eta')/\log(1-\gamma^M) + 1$. Then using the inequality $\log(1-x) \leq -x$ for $x \in (0,1)$, we have

$$L \leq \frac{\log\{1/(1-\eta')\} + 1}{\gamma^M}.$$

Note that from the definition of γ_0 we have $\gamma^{-M} = p^{\log(\gamma)/\log(\gamma_0)}$. We then see that

$$\begin{aligned}
\gamma^{-M}\,\mathcal{E}(E_1) &= \gamma^{-M} \sum_{j,k} \gamma_{jk}^M \\
&\leq \gamma^{-M} \Bigg(\sum_{j,k:\gamma_{jk} > \gamma} \gamma_{jk}^M + \sum_{j,k:\gamma_0 < \gamma_{jk} \leq \gamma} \gamma_{jk}^M + \sum_{j,k:\gamma_{jk} \leq \gamma_0} \gamma_{jk}^M \Bigg) \\
&\leq c_1 p \gamma^{-M} + c_2 p^{1+\log(\gamma)/\log(\gamma_0)} + p^2 \gamma_0^M \gamma^{-M} \\
&\leq (c_1 + c_2 + 1) p^{1+\log(\gamma)/\log(\gamma_0)}.
\end{aligned}$$

Collecting together the terms in (2.10) we have

$$\begin{aligned}
C(M,L) \leq& np \\
&+ [\log\{1/(1-\eta')\} + 1][\log(p)\{1 + 1/\log(\gamma_0^{-1})\} + n(c_1 + c_2 + 1)]p^{1+\frac{log(\gamma)}{\log(\gamma_0)}}
\end{aligned}$$

from which the result easily follows.

Proof of Proposition 2.2.3

Let $\eta^* = \eta(M^*, L)$. Note that in order for $\eta(M', L') \geq \eta^*$ it must be the case that $L' \geq \log(1-\eta^*)/\log(1-\gamma^{M'})$. Therefore

$$\begin{aligned}
C(M', L') - np &\geq \frac{\log(1-\eta^*)}{\log(1-\gamma^{M'})}\Big(M'p + p\log(p) + n\sum_{j,k}\gamma_{jk}^{M'}\Big) \\
&\geq \min_{M\in\mathcal{N}} \frac{\log(1-\eta^*)}{\log(1-\gamma^{M})}\Big(Mp + p\log(p) + n\sum_{j,k}\gamma_{jk}^{M}\Big) \qquad (2.32) \\
&= \frac{\log(1-\eta^*)}{\log(1-\gamma^{M^*})}\Big(M^*p + p\log(p) + n\sum_{j,k}\gamma_{jk}^{M^*}\Big) = C(M^*, L).
\end{aligned}$$

Moreover, the inequality leading to (2.32) is strict if M^* is the unique minimiser and $M' \neq M^*$.

Technical lemmas

Lemma 2.7.6. *Let $F \in \mathcal{F}$ and suppose $(R_i)_{i=1}^\infty$ is an i.i.d. sequence with $R_i \sim F$.*

Then for all $a > 0$, there exists $c_1, c_2 > 0$ and $l_0 \in \mathbb{N}$ such that for all $l \geq l_0$ and $0 \leq \tau \leq a\sqrt{l}$ we have

$$\frac{c_1\tau}{\sqrt{l}} \leq \mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big| \leq \tau\Big) \leq \frac{c_2\tau}{\sqrt{l}}.$$

Proof. Let f_l be the density of $\sum_{i=1}^{l} R_i/\sqrt{l}$. Note that as $\mathcal{E}(|R_1|^3) < \infty$, we must have $\mathcal{E}(R_1^2) < \infty$, so we may assume without loss of generality that $\mathcal{E}(R_1^2) = 1$. Then by Theorem 3 of Petrov (1964) we have that for sufficiently large l,

$$|f_l(t) - \phi(t)| \leq \frac{c}{\sqrt{l}(1+|t|^3)}. \qquad (2.33)$$

Here c is a constant and $\phi(t) = e^{-t^2/2}/\sqrt{2\pi}$ is the standard normal density. Now by the mean value theorem, we have

$$2\inf_{0\leq t\leq \tau/\sqrt{l}}\{f_l(t)\}\frac{\tau}{\sqrt{l}} \leq \mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big|/\sqrt{l} \leq \tau/\sqrt{l}\Big) \leq 2\sup_{0\leq t\leq \tau/\sqrt{l}}\{f_l(t)\}\frac{\tau}{\sqrt{l}}.$$

Thus from (2.33), for l sufficiently large we have

$$\mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big| \le \tau\Big) \ge \frac{\tau}{\sqrt{l}}\Big(\frac{\sqrt{2}}{\sqrt{\pi}}\exp\{-\tau^2/(2l)\} - \frac{2c}{\sqrt{l}}\Big).$$

Note that for $a > 0$ and l sufficiently large we have $\sqrt{2/\pi}e^{-a^2/2} > 2c/\sqrt{l}$, whence

$$\mathbb{P}\Big(\big|\sum_{i=1}^{l} R_i\big| \le \tau\Big) \ge \frac{c_1\tau}{\sqrt{l}}$$

for $0 \le \tau \le a\sqrt{l}$, some $c_1 > 0$. A similar argument yields the upper bound in the final result. □

Lemma 2.7.7. *Suppose* $\gamma \in [0,1)$. *For all* $M \in \mathcal{N}$ *we have*

$$\sum_{r=1}^{M} \frac{1}{\sqrt{r}}\binom{M}{r}(1-\gamma)^r\gamma^{M-r} \le \frac{\sqrt{2}}{\sqrt{(1-\gamma)M}}. \tag{2.34}$$

Given $r_0 \in \mathcal{N}$ *and* $\gamma \in [0,1)$, *there exists* $c > 0$ *and* $M_0 \in \mathcal{N}$ *such that for all* $M \ge M_0$ *we have*

$$\sum_{r=r_0}^{M} \frac{1}{\sqrt{r}}\binom{M}{r}(1-\gamma)^r\gamma^{M-r} \ge \frac{c}{\sqrt{(1-\gamma)M}}. \tag{2.35}$$

Proof. First we show the upper bound (2.34). Let $J \sim \text{Binomial}(M, 1-\gamma)$.

$$\begin{aligned}\sum_{r=1}^{M} \frac{1}{\sqrt{r}}\binom{M}{r}(1-\gamma)^r\gamma^{M-r} &\le \sqrt{2}\sum_{r=1}^{M} \frac{1}{\sqrt{r+1}}\binom{M}{r}(1-\gamma)^r\gamma^{M-r} \\ &\le \sqrt{2}\,\mathcal{E}(1/\sqrt{J+1}).\end{aligned}$$

Next, by Jensen's inequality we have $\mathcal{E}(1/\sqrt{J+1}) \le \sqrt{\mathcal{E}\{1/(J+1)\}}$.

We now compute $\mathcal{E}\{1/(J+1)\}$ as follows.

$$\begin{aligned}
\mathcal{E}\left(\frac{1}{J+1}\right) &= \sum_{r=0}^{M} \frac{1}{r+1}\binom{M}{r}(1-\gamma)^r\gamma^{M-r} \\
&= \frac{1}{M+1}\sum_{r=0}^{M}\binom{M+1}{r+1}(1-\gamma)^r\gamma^{M-r} \\
&= \frac{1}{(1-\gamma)(M+1)}\sum_{r=0}^{M}\binom{M+1}{r+1}(1-\gamma)^{r+1}\gamma^{M-r} \\
&= \frac{1-\gamma^{M+1}}{(1-\gamma)(M+1)} \leq \frac{1}{(1-\gamma)(M+1)}.
\end{aligned}$$

Putting things together gives (2.34).

Turning now to (2.35), we see that the LHS equals

$$\mathcal{E}(1/\sqrt{J}\mathbb{1}_{\{J\geq r_0\}}) = \mathcal{E}(1/\sqrt{J}|J \geq r_0)\mathbb{P}(J \geq r_0).$$

By Jensen's inequality we have

$$\mathcal{E}(1/\sqrt{J}|J \geq r_0) \geq \frac{1}{\sqrt{\mathcal{E}(J|J \geq r_0)}} = \frac{\sqrt{\mathbb{P}(J \geq r_0)}}{\sqrt{\mathcal{E}(J\mathbb{1}_{\{J\geq r_0\}})}} \geq \frac{\sqrt{\mathbb{P}(J \geq r_0)}}{\sqrt{(1-\gamma)M}}.$$

But as $M \to \infty$, $\mathbb{P}(J \geq r_0) \to 1$, which easily gives the result. □

Lemma 2.7.8. *Let $f : [0,\infty) \to [0,1]$ be non-decreasing. Suppose there exists $0 < \alpha_1 < \alpha_0$ such that:*

- *(i) f is concave on $[0, \alpha_0]$;*
- *(ii) $-\sup\big(\partial(-f)(\alpha_1)\big) \geq \{1 - f(\alpha_1)\}/(\alpha_0 - \alpha_1)$, where $\partial(-f)(\alpha_1)$ denotes the subdifferential of the function $-f$ at α_1.*

Then if random variable X has $\mathcal{E}(X) \leq \alpha_0$, then $f(\mathcal{E}\,X) \geq \mathcal{E}\,f(X)$.

Proof. Write $m = -\sup\big(\partial(-f)(\alpha_1)\big)$ Let function $g : [0,\infty) \to [0,\infty)$ be defined as follows.

$$g(x) = \begin{cases} f(x) & \text{if } 0 \leq x \leq \alpha_1 \\ f(\alpha_1) + m(x - \alpha_1) & \text{if } x > \alpha_1. \end{cases}$$

A Note that g thus defined has $g(\alpha_0) \geq 1$. We see that g is convex and $g \geq f$. Thus if $\mathcal{E}(X) \leq \alpha_1$, by Jensen's inequality we have

$$f(\mathcal{E}\, X) = g(\mathcal{E}\, X) \geq \mathcal{E}\, g(X) \geq \mathcal{E}\, f(X).$$

□

Appendix B

Connection to LSH

Minimal subsampling as considered in Algorithm 2 is closely related to the locality-sensitive hashing (LSH) framework: Define $h(j) = \mathbf{R}^T \mathbf{X}_j$ (R corresponds to the minimal subsampling projection) to be the hashing function and $\mathcal{H}$ to be the family of such functions, from which we sample uniformly. Then $\mathcal{H}$ is $(\gamma, c\gamma, p_1, p_2)$-sensitive, that is:

- if $\gamma_{jk} \geq \gamma$ then $\mathbb{P}(h(j) = h(k)) \geq p_1$
- if $\gamma_{jk} \leq c\gamma$ then $\mathbb{P}(h(j) = h(k)) \leq p_2$,

where $0 < c < 1$. In the case of the minimal subsampling we have $p_1 = \gamma^M$ and $p_2 = \gamma^M c^M$. However, the typical LSH machinery cannot be applied directly to the equal pairs problem above. In our setting, we are not interested in preserving close pairs but rather the closest pairs. Theorem 2.2.1 establishes that the family $\mathcal{H}$ leads to the maximal ratio p_1/p_2 among all linear hashing families.

Appendix C

Proof of Proposition 2.3.1

Proof.

$$\begin{aligned}
\mathbb{P}(\text{sign}(Y_i) = \tilde{X}_{ij}\tilde{X}_{ik}) &= \frac{\text{sign}(Y_i)+1}{2}(g(X_{ij})g(X_{ik}) + (1-g(X_{ij}))(1-g(X_{ik}))) \\
&+ \frac{1-\text{sign}(Y_i)}{2}(g(X_{ij})(1-g(X_{ik})) + (1-g(X_{ij}))g(X_{ik})) \\
&= \frac{1}{2} + \frac{\text{sign}(Y_i)}{2}(1-2g(X_{ij}))(1-2g(X_{ik})).
\end{aligned}$$

□

Appendix D

The unbiased transform and the sign transform

Proposition 2.3.3

Proof. The equation

$$\mathcal{E}[\tilde{X}_{ij}] = \mathbb{P}(\tilde{X}_{ij} = 1) - \mathbb{P}(\tilde{X}_{ij} = -1) = X_{ij},$$

implies

$$\mathbb{P}(\tilde{X}_{ij} = 1) = \frac{X_{ij} + 1}{2}$$

This uniquely determines the unbiased transform. □

Next we show two Lemmas that will be useful when proving Theorems 2.3.4 and 2.3.5.

Lemma 2.7.9. *Consider the setup of Theorem 2.3.4. Then there exists constants* $C_1^\varepsilon, C_2^\varepsilon > 0$ *such that defining*

$$\alpha_{n,p}^u = \alpha_{n,p}^u(t) = \left(1 + \frac{t + \log(nC_1^\varepsilon)}{C_2^\varepsilon}\right)\sqrt{2\{t + \log(4p)\}/n},$$

with probability at least $1 - 2\exp(-t)$ *we have:*

$$\frac{\sum_i Y_i X_{ij^*} X_{ik^*}}{\|\mathbf{Y}\|_1} \notin \left[-\frac{m_2 - \alpha_{n,p}^u}{m_1 + m_\varepsilon + \alpha_{n,p}^u}, \frac{m_2 - \alpha_{n,p}^u}{m_1 + m_\varepsilon + \alpha_{n,p}^u}\right]$$
$$\frac{\sum_{i=1}^n Y_i X_{ij} X_{ik}}{\|\mathbf{Y}\|_1} \in \left[-\frac{m_2(1 - r_u) + \alpha_{n,p}^u}{m_1 - \alpha_{n,p}^u}, \frac{m_2(1 - r_u) + \alpha_{n,p}^u}{m_1 - \alpha_{n,p}^u}\right] \forall (j,k) \neq (j^*, k^*).$$

Proof. First we consider a capped version of ε:

$$\varepsilon_i' = \begin{cases} \varepsilon_i \text{ if } |\varepsilon_i| \leq \sigma \\ \sigma \text{sign}(\varepsilon_i) \text{ otherwise,} \end{cases}$$

where σ is to be chosen later. We may apply Hoeffding's inequality to these bounded variables. We have to bound two terms:

$$\frac{\sum_{i=1}^{n} Y_i X_{ij^*} X_{ik^*}}{\|\mathbf{Y}\|_1}$$

from below and $\dfrac{\sum_{i=1}^{n} Y_i X_{ij} X_{ik}}{\|\mathbf{Y}\|_1}$ from above, for $(j,k) \neq (j^*,k^*)$.

Schematically the first term can be dealt with in the following way:

$$\mathbb{P}\Big(\frac{A+B}{C+D} \geq \frac{a+b}{c+d}\Big) \geq 1 - \mathbb{P}(A \leq a) - \mathbb{P}(B \leq b) - \mathbb{P}(C \geq c) - \mathbb{P}(D \geq d) \tag{2.36}$$

where

$$A+B = \sum_{i=1}^{n} (X_{ij^*} X_{ik^*})^2 + \varepsilon_i' X_{ij^*} X_{ik^*} \quad \text{and} \quad C+D = \sum_{i=1}^{n} |X_{ij^*} X_{ik^*} + \varepsilon_i'|.$$

We deal with each term individually. Using Hoeffding's inequality we get:

$$A:\ \mathbb{P}\Big(\textstyle\sum_{i=1}^{p} (X_{ij^*} X_{ik^*})^2 \leq nm_2 - \delta\Big) \leq \exp(-\delta^2/2n))$$

$$B:\ \mathbb{P}\Big(\textstyle\sum_{i=1}^{n} \varepsilon_i' X_{ij^*} X_{ik^*} \leq -\kappa\Big) \leq \exp(-\kappa^2/2n\sigma^2)$$

$$C:\ \mathbb{P}\Big(\textstyle\sum_{i=1}^{n} |X_{ij^*} X_{ik^*}| \geq nm_1 + \delta\Big) \leq \exp(-\delta^2/2n)$$

$$D:\ \mathbb{P}\Big(\textstyle\sum_{i=1}^{n} |\varepsilon_i'| \geq nm_\varepsilon + \kappa\Big) \leq \exp(-2\kappa^2/n\sigma^2).$$

This gives us a bound of the interaction strength of the true interaction pair:

$$\begin{aligned}\mathbb{P}\Big(\frac{\sum_i Y_i X_{ij^*} X_{ik^*}}{\|\mathbf{Y}\|_1} &\geq \frac{nm_2 - \delta - \kappa}{nm_1 + nm_\varepsilon + \delta + \kappa}\Big) \\ &\geq 1 - \exp(-\delta^2/2n) - \exp(-\delta^2/2n) \\ &- \exp(-\kappa^2/2n\sigma^2) - \exp(-\kappa^2/2n\sigma^2)\end{aligned}$$

Similarly we can treat the interaction strength of the non interacting pairs:

$A:$ Here we use assumption $(B1)$:

$$m_2(r_u-1) \leq \mathcal{E}[X_{ij^*}X_{ik^*}X_{im}X_{io}] \leq m_2(1-r_u).$$

Hence, $\mathbb{P}\Big(\sum_{i=1}^n X_{ij^*}X_{ik^*}X_{ij}X_{ik} \geq nm_2(1-r_u)+\delta\Big) \geq \exp(-\delta/2n).$

For the rest we run the same bounds as before (using $|X_{ij^*}X_{ik^*}+\varepsilon_i'| \geq |X_{ij^*}X_{ik^*}|+\varepsilon_i'$). This yields the bound

$$\mathbb{P}\Big(\frac{\sum_{i=1}^n Y_iX_{ij}X_{ik}}{\|\mathbf{Y}\|_1} \leq \frac{nm_2(1-r_u)+\delta+\kappa}{nm_1-\delta-\kappa}\Big)$$
$$\geq 1-\exp(-\delta^2/2n)-\exp(-\delta^2/2n)$$
$$-\exp(-\kappa^2/2n\sigma^2)-\exp(-\kappa^2/2n\sigma^2)$$

The above inequality needs to hold for all at most p^2 pairs that are not interactions, so that we effectively multiply the exponential terms with p^2. Another factor of 2 is multiplied in for the negative sign, as the fraction also has to be bounded away from -1. In total we thus have:

$$\frac{\sum_{i=1}^n Y_iX_{ij^*}X_{ik^*}}{\|\mathbf{Y}\|_1} \notin \Big[-\frac{nm_2-\delta-\kappa}{nm_1+nm_\varepsilon+\delta+\kappa}, \frac{nm_2-\delta-\kappa}{nm_1+nm_\varepsilon+\delta+\kappa}\Big]$$
$$\frac{\sum_{i=1}^n Y_iX_{ij}X_{ik}}{\|\mathbf{Y}\|_1} \in \Big[-\frac{nm_2(1-r_u)+\delta+\kappa}{nm_1-\delta-\kappa}, \frac{nm_2(1-r_u)+\delta+\kappa}{nm_1-\delta-\kappa}\Big] \ \forall(m,o)\neq(j,l)$$

with probability at least

$$1-\exp(-\delta^2/2n)-\exp(-\delta^2/2n)-\exp(-\kappa^2/2n\sigma^2)-\exp(-\kappa^2/2n\sigma^2).$$

Finally, let $\sigma \geq 1$, then we have to set δ and κ so that the probability is bigger than $1-\exp(-t)$. This gives:

$$\exp(-t) = 4p\exp(-\delta^2/2n) \text{ and } \exp(-t) = 4p\exp(-\kappa^2/2n\sigma^2).$$

This gives

$$\delta = \sqrt{2n(t+\log(4p))} \text{ and } \kappa = \sqrt{2n\sigma^2(t+\log(4p))}.$$

Thus for $\alpha_{n,p}^u = \frac{\sqrt{2(t+\log(4p))(1+\sigma^2)}}{\sqrt{n}}$,

$$\frac{\sum_i Y_iX_{ij^*}X_{ik^*}}{\|\mathbf{Y}\|_1} \notin \Big[-\frac{m_2-\alpha_{n,p}^u}{m_1+m_\varepsilon+\alpha_{n,p}^u}, \frac{m_2-\alpha_{n,p}^u}{m_1+m_\varepsilon+\alpha_{n,p}^u}\Big]$$
$$\frac{\sum_{i=1}^n Y_iX_{ij}X_{ik}}{\|\mathbf{Y}\|_1} \in \Big[-\frac{m_2(1-r_u)+\alpha_{n,p}^u}{m_1-\alpha_{n,p}^u}, \frac{m_2(1-r_u)+\alpha_{n,p}^u}{m_1-\alpha_{n,p}^u}\Big]$$
$$\forall(j,k)\neq(j^*,k^*) \text{ with probability at least } 1-\exp(-t).$$

Now we extend this result to the case of unbounded errors, that is we now assume that with high probability ε_i are bounded:

$$\mathbb{P}(\varepsilon_i = \varepsilon_i',\ \forall\, i) = 1 - \exp(-t).$$

Here we used the sub-exponential tail behavior of ε. We have $\mathbb{P}(|\varepsilon_i| \geq t) \leq C_1^{\varepsilon} \exp(-C_2^{\varepsilon} t)$. Hence we set

$$t = C_2^{\varepsilon}\sigma - \log(nC_1^{\varepsilon}) \ \Rightarrow \sigma = \frac{t + \log(nC_1^{\varepsilon})}{C_2^{\varepsilon}}$$

Thus,

$$\alpha_{n,p}^u = \frac{\sqrt{2\{t + \log(4p)\}\{1 + (\frac{t+\log(nC_1^{\varepsilon})}{C_2^{\varepsilon}})\}^2\}}}{\sqrt{n}}$$

with probability at least $1 - 2\exp(-t)$ we have:

$$\begin{aligned}
\frac{\sum_i Y_i X_{ij^*} X_{ik^*}}{\|\mathbf{Y}\|_1} &\notin \Big[- \frac{m_2 - \alpha_{n,p}^u}{m_1 + m_\varepsilon + \alpha_{n,p}^u}, \frac{m_2 - \alpha_{n,p}^u}{m_1 + m_\varepsilon + \alpha_{n,p}^u} \Big] \\
\frac{\sum_{i=1}^n Y_i X_{ij} X_{ik}}{\|\mathbf{Y}\|_1} &\in \Big[- \frac{m_2(1 - r_u) + \alpha_{n,p}^u}{m_1 - \alpha_{n,p}^u}, \frac{m_2(1 - r_u) + \alpha_{n,p}^u}{m_1 - \alpha_{n,p}^u} \Big] \\
\forall (j,k) \neq (j^*, k^*).
\end{aligned}$$

□

Next we prove the equivalent result for the sign transform. The proof is very similar to the unbiased case:

Lemma 2.7.10. *Consider the setup of Theorem 2.3.5. Then there exists constants* $C_1^X, C_2^X, C_1^{\varepsilon}, C_2^{\varepsilon} > 0$ *such that defining*

$$\alpha_{n,p}^s = \alpha_{n,p}^s(t) = \frac{\sqrt{2(t + \log(4p))\Big(\Big(\frac{t+\log(pnC_1^X)}{C_2^X}\Big)^4 + \Big(\frac{t+\log(nC_1^{\varepsilon})}{C_2^{\varepsilon}}\Big)^2\Big)}}{\sqrt{n}},$$

with probability at least $1 - 3\exp(-t)$ *we have:*

$$\begin{aligned}
\frac{\sum_{i=1}^n Y_i \mathrm{sign}(X_{ij^*} X_{ik^*})}{\|\mathbf{Y}\|_1} &\notin \Big[- \frac{m_1 - \alpha_{n,p}^s}{m_1 + m_\varepsilon + \alpha_{n,p}^s}, \frac{m_1 - \alpha_{n,p}^s}{m_1 + m_\varepsilon + \alpha_{n,p}^s} \Big] \\
\frac{\sum_{i=1}^n Y_i \mathrm{sign}(X_{ij} X_{ik})}{\|\mathbf{Y}\|_1} &\in \Big[- \frac{m_1(1 - r_s) + \alpha_{n,p}^s}{m_1 - \alpha_{n,p}^s}, \frac{m_1(1 - r_s) + \alpha_{n,p}^s}{m_1 - \alpha_{n,p}^s} \Big] \\
\forall\, (m, o) \neq (j^*, k^*).
\end{aligned}$$

Proof. First consider capped versions of the random variables of interest:

$$X'_{ij} = \begin{cases} X_{ij} \text{ if } |X_{ij}| \leq M \\ M\text{sign}(X_{ij}) \text{ otherwise} \end{cases} \qquad \text{and} \quad \varepsilon'_i = \begin{cases} \varepsilon_i \text{ if } |\varepsilon_i| \leq \sigma \\ \sigma\text{sign}(\varepsilon_i) \text{ otherwise} \end{cases}$$

where M and σ are to be chosen later. Given these capped variables we can use Hoeffding's inequality as we now deal with bounded variables. We have to bound two terms:

$$\frac{\sum_{i=1}^n Y_i\text{sign}(X'_{ij^*}X'_{ik^*})}{\|\mathbf{Y}\|_1}$$

$$\text{from below and } \frac{\sum_{i=1}^n Y_i\text{sign}(X'_{ij}X'_{ik})}{\|\mathbf{Y}\|_1} \text{ from above, for } (j,k) \neq (j^*,k^*)$$

As in Lemma 2.7.9 equation (2.36):

$$A+B = \sum_{i=1}^n |X'_{ij^*}X'_{ik^*}| + \varepsilon'_i\text{sign}(X'_{ij^*}X'_{ik^*}) \quad \text{and} \quad C+D = \sum_{i=1}^n |X'_{ij^*}X'_{ik^*} + \varepsilon'_i|.$$

We deal with each term individually. Using Hoeffding's inequality we get:

$$A: \mathbb{P}\Big(\sum_{i=1}^p |X'_{ij^*}X'_{ik^*}| \leq nm_1 - \delta\Big) \leq \exp(-\delta^2/2nM^4))$$

$$B: \mathbb{P}\Big(\sum_{i=1}^n \varepsilon'_i \leq -\kappa\Big) \leq \exp(-\kappa^2/2n\sigma^2)$$

$$C: \mathbb{P}\Big(\sum_{i=1}^n |X'_{ij^*}X'_{ik^*}| \geq nm_1 + \delta\Big) \leq \exp(-\delta^2/2nM^4)$$

$$D: \mathbb{P}\Big(\sum_{i=1}^n |\varepsilon'| \geq nm'_\varepsilon + \kappa\Big) \leq \exp(-2\kappa^2/n\sigma^2)$$

This gives us a bound of the interaction strength of the true interaction pair:

$$\begin{aligned} \mathbb{P}\Big(\frac{\sum_i Y_i\text{sign}(X'_{ij^*}X'_{ik^*})}{\|\mathbf{Y}\|_1} &\geq \frac{nm_1 - \delta - \kappa}{nm_1 + nm_\varepsilon + \delta + \kappa}\Big) \\ &\geq 1 - 2\exp(-\delta^2/2nM^4) - 2\exp(-\kappa^2/2n\sigma^2) \end{aligned}$$

Similarly we can treat the interaction strength of the non interacting pairs:

$A:$ Here we use assumption $(C1)$. It implies

$$r_s/2 \leq \mathbb{P}(\text{sign}(X'_{ij^*}X'_{ik^*}) = \text{sign}(X'_{ij}X'_{ik})|\mathbf{X}) \leq 1 - r_s/2.$$

This we use for computing the expectation:

$$\begin{aligned}\mathbb{E}[X'_{ij^*}X'_{ik^*}\text{sign}(X'_{ij}X'_{ik})] &= \mathbb{E}[\mathbb{E}[|X'_{ij^*}X'_{ik^*}|\text{sign}(X'_{ij}X'_{ik}X'_{ij^*}X'_{ik^*})] \\ &= \mathbb{E}[\mathbb{E}[2|X'_{ij^*}X'_{ik^*}|\mathbf{1}_{\{\text{sign}(X'_{ij}X'_{ik}X'_{ij^*}X'_{ik^*})=1\}}|\mathbf{X}]] - \mathbb{E}[|X'_{ij^*}X'_{ik^*}|] \\ &= \mathbb{E}[\mathbb{E}[2|X'_{ij^*}X'_{ik^*}||\mathbf{X}]]\mathbb{P}(\text{sign}(X'_{ij}X'_{ik}X'_{ij^*}X'_{ik^*}) = 1|\mathbf{X}) - \mathbb{E}[|X'_{ij^*}X'_{ik^*}|] \\ &= \mathbb{E}[|X'_{ij^*}X'_{ik^*}|](2\mathbb{P}(\text{sign}(X'_{ij}X'_{ik}X'_{ij^*}X'_{ik^*}) = 1|\mathbf{X}) - 1).\end{aligned}$$

Thus the expectation is given as:

$$m_1(r_s - 1) \leq E[X'_{ij^*}X'_{ik^*}\text{sign}(X'_{ij}X'_{ik})] \leq m_1(1 - r_s).$$

Hence, $\mathbb{P}\Big(\sum_{i=1}^n X'_{ij^*}X'_{ik^*}\text{sign}(X'_{ij}X'_{ik}) \geq nm_1(1-r_s)+\delta\Big) \geq \exp(-2\delta/nM^4)$.

For the rest we use the same bounds as before (using $|X'_{ij^*}X'_{ik^*} + \varepsilon'_i| \geq |X'_{ij^*}X'_{ik^*}| + \varepsilon'_i$). This yields the bound

$$\begin{aligned}\mathbb{P}\Big(\frac{\sum_{i=1}^n Y_i\text{sign}(X'_{ij}X'_{ik})}{\|\mathbf{Y}\|_1} &\leq \frac{nm_1(1 - r_s) + \delta + \kappa}{nm_1 - \delta - \kappa}\Big) \\ &\geq 1 - \exp(-2\delta^2/nM^4) - \exp(-2\kappa^2/n\sigma^2).\end{aligned}$$

The above inequality needs to hold for the at most p^2 pairs that are not interactions, so that we effectively multiply the exponential terms with p^2. Another factor of 2 is multiplied in for the negative sign, as the fraction also has to be bounded away from -1. In total we thus have:

$$\begin{aligned}\frac{\sum_i Y_i\text{sign}(X'_{ij^*}X'_{ik^*})}{\|\mathbf{Y}\|_1} &\notin \Big[-\frac{nm_1 - \delta - \kappa}{nm_1 + nm_\varepsilon + \delta + \kappa}, \frac{nm_1 - \delta - \kappa}{nm_1 + nm_\varepsilon + \delta + \kappa}\Big] \\ \frac{\sum_{i=1}^n Y_i\text{sign}(X'_{ij}X'_{ik})}{\|\mathbf{Y}\|_1} &\in \Big[-\frac{nm_1(1 - r_s) + \delta + \kappa}{nm_1 - \delta - \kappa}, \frac{nm_1(1 - r_s) + \delta + \kappa}{nm_1 - \delta - \kappa}\Big] \\ \forall (j, k) \neq (j^*, k^*)\end{aligned}$$

with probability at least $1 - 2p\exp(-\delta^2/2nM^4) - 2p\exp(-\kappa^2/2n\sigma^2)$.

Finally we have to set δ and κ so that the probability is bigger than $1 - \exp(-t)$. This gives:

$$\exp(-t) = 4p\exp(-\delta^2/2nM^4) \text{ and } \exp(-t) = 4p\exp(-\kappa^2/2n\sigma^2)$$

This gives

$$\delta = \sqrt{2nM^4(t + \log(4p))} \text{ and } \kappa = \sqrt{2n\sigma^2(t + \log(4p))}$$

Thus for $\alpha^s_{n,p} = \frac{\sqrt{2(t+\log(4p))(M^4+\sigma^2)}}{\sqrt{n}}$

$$\begin{aligned}
&\frac{\sum_i Y_i \text{sign}(X'_{ij^*}X'_{ik^*})}{\|\mathbf{Y}\|_1} \notin \Big[-\frac{m_1 - \alpha^s_{n,p}}{m_1 + m_\varepsilon + \alpha^s_{n,p}}, \frac{m_1 - \alpha^s_{n,p}}{m_1 + m_\varepsilon + \alpha^s_{n,p}}\Big] \\
&\frac{\sum_{i=1}^n Y_i \text{sign}(X'_{ij}X'_{ik})}{\|\mathbf{Y}\|_1} \in \Big[-\frac{m_1(1-r_s) + \alpha^s_{n,p}}{m_1 - \alpha^s_{n,p}}, \frac{m_1(1-r_s) + \alpha^s_{n,p}}{m_1 - \alpha^s_{n,p}}\Big] \\
&\qquad \forall (j,k) \neq (j^*,k^*)
\end{aligned}$$

with probability at least $1 - \exp(-t)$.

We now extend this result to the case of unbounded variables, that is we now assume that with high probability the variables X_{ij} and ε_i are bounded:

$$\mathbb{P}(X_{ij} = X'_{ij},\ \forall\, i,j) = 1 - \exp(-t) \text{ and } \mathbb{P}(\varepsilon_i = \varepsilon'_{ij},\ \forall\, i) = 1 - \exp(-t).$$

Here we used the sub-exponential tail behaviour of the X_{ij} and ε_i. There exists constants C^X_1, C^X_2 such that $\mathbb{P}(|X_{ij}| \geq t) \leq C^X_1 \exp(-C^X_2 t)$ and similarly for ε. Hence we set

$$\begin{aligned}
t = C^X_2 M - \log(pnC^X_1) \ &\Rightarrow M = \frac{t + \log(pnC^X_1)}{C^X_2} \\
t = C^\varepsilon_2 \sigma - \log(nC^\varepsilon_1) \ &\Rightarrow \sigma = \frac{t + \log(nC^\varepsilon_1)}{C^\varepsilon_2}
\end{aligned}$$

Thus we have

$$\alpha^s_{n,p} = \frac{\sqrt{2(t + \log(4p))\Big(\Big(\frac{t+\log(pnC^X_1)}{C^X_2}\Big)^4 + \Big(\frac{t+\log(nC^\varepsilon_1)}{C^\varepsilon_2}\Big)^2\Big)}}{\sqrt{n}}$$

□

Next we prove **Theorem** 2.3.4:

Proof. Given $\delta, \epsilon > 0$, choose t such that $3\exp(-t) < \epsilon$. From (B3) we have that $\alpha^u_{n,p}(t)$ defined in Lemma 2.7.9 satisfies $\alpha^u_{n,p}(t) \to 0$ as $n \to \infty$.

Thus from Lemma 2.7.9 we know that there exists N such that for all $n \geq N$, with probability $1 - \epsilon$ we have

$$\frac{\log(\gamma^g_{j^*k^*})}{\log(\gamma^g_{jk})} < \frac{\log\{(1 + \frac{m_2}{m_1+m_\varepsilon})/2\}}{\log\{(1 + \frac{m_1}{m_2(1-r_s)})/2\}} + \delta/2.$$

Thus for $n \geq N$, applying Corollary 2.3.2 we have that with probability $1 - \epsilon$,

$$C(M, L) \leq cnp^{1+\delta/2+\frac{\log(1/2+m_2/2((m_1+m_\varepsilon)))}{\log(1/2+m_2(1-r_u)/(2m_1))}},$$

for some constant c. □

The proof of **Theorem** 2.3.5 is very similar and is thus omitted.

Chapter 3

Rowspace projections for fast graphical model selection under latent confounding[1]

3.1 Introduction

Conditional independence graphs are well-established as a formalism for identifying related variables in high-dimensional settings. They consist of a node for each variable and have an edge between two nodes if and only if the corresponding variables are conditionally dependent given all other variables. The most popular model for continuous data assumes that the observations have a multivariate Gaussian distribution. In this context, lack of edges in the conditional independence graph correspond to zeroes in the inverse covariance – or *precision* – matrix Lauritzen (1996). To facilitate estimation of the location of the zeroes when the number of variables p exceeds the number of observations n, it is typically assumed that the precision matrix is sparse. Two basic approaches to the estimation problem have emerged in the literature. One can aim

---

[1] This chapter is not yet publicly available.

for a sparse estimate of the precision matrix, for example by adding a ℓ_1 penalty to the entries in the usual Gaussian log-likelihood; this is known as the graphical Lasso (Friedman et al., 2008; Yuan and Lin, 2007a). An alternative is to form sparse estimates of the means of each variable conditional on every other variable, for example through nodewise Lasso regressions Tibshirani (1996): this is the so-called neighbourhood selection approach (Meinshausen and Bühlmann, 2006).

Although the simplifying assumption of sparsity has motivated the development of several popular methods for conditional independence graph estimation, there are a number of settings of interest where the presence of unobserved variables acting as confounders is known to be an issue and prevents the marginal inverse covariance of the observed data from being sparse. In genomics, for example, confounding occurs due to technical factors or unobserved environmental variables (Gagnon-Bartsch et al., 2013; Leek and Storey, 2007; Stegle et al., 2012). In other settings, confounding may be present but its origin unknown. Driven by the needs of practitioners, there has been growing interest in extensions that allow for the presence of latent variables: conditional on these unobserved latent variables, one can assume that the observations have a sparse inverse covariance matrix. This model was pioneered by Chandrasekaran et al. (2012) in the context Gaussian graphical model (GGM) estimation, building on previous work for the noiseless setting by Chandrasekaran et al. (2011) and Candès et al. (2011). In order to guarantee identifiability, it is usually assumed that there are only a few latent variables, and that each of them acts on a large fraction of the p observed variables.

Whilst much progress has been made in developing estimation procedures for both the sparse Gaussian graphical model and its extension with latent variables, there remain several important challenges.

Methods for both models struggle computationally when p is large. The problem is most severe for methods capable of handling latent variables as their scaling with p can prohibit their application on datasets where p is moderate in size (*i.e.* of the order of a thousand). For example, whole genome-scale data remains a challenge even for some of the fastest methods for sparse GGM estimation, such as neighbourhood selection, particularly when it is desired to apply methods in an interactive fashion, as is often demanded by applications, for example when making choices about how to preprocess the data.

In this paper, we propose a simple method for estimating the structure of the conditional independence graph in the challenging latent variable GGM setting. To fix ideas, consider a data matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$ and an unobserved matrix of latent variables $\mathbf{L} \in \mathbb{R}^{n\times q}$. Suppose that the rows of the combined matrix $(\mathbf{X}, \mathbf{L}) \in \mathbb{R}^{n\times(p+q)}$ have independent $\mathcal{N}_p(0, \mathbf{\Sigma})$ distributions. The goal is to estimate the subgraph of the full conditional independence graph, or equivalently to locate the non-zero entries in the submatrix $\mathbf{\Omega^X} \in \mathbb{R}^{p\times p}$ of $\mathbf{\Omega} := \mathbf{\Sigma}^{-1}$, corresponding to the observed variables. Note that the marginal conditional independence graph of $\mathbf{X}$ alone may be dense even if $\Omega^{\mathbf{X}}$ is sparse, due to the latent confounding, and thus may not provide meaningful information about which variables are closely related to one another.

Our procedure is based on the following estimator:

$$\hat{\mathbf{G}} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}. \tag{3.1}$$

Whilst $\hat{\mathbf{G}}$ does not directly provide an estimate $\mathbf{\Omega^X}$, we demonstrate that the entries in $\hat{\mathbf{G}}$ with large magnitudes correspond to large magnitude entries in $\mathbf{\Omega^X}$. In this way, the method gives an effective way of screening for edges in the conditional independence subgraph for $\mathbf{X}$.

Our method is closely connected to employing nodewise ridge regressions but allowing the tuning parameter to tend to 0. Recently Wang and Leng (2015) studied such a scheme, which they call *high-dimensional ordinary least-squares projection* (HOLP) for variable selection in a sparse high-dimensional linear model setting. HOLP is equivalent to projections to the rowspace of the design. While their method is compelling, there seems to be a gap in their theoretical assessment of the procedure. Nevertheless, it seems that by imposing additional model assumptions their result could be saved. We will from now on refer to HOLP as rowspace projections (RP). As a single application of RP has a computational complexity of $O(n^2p)$ naively employing this for each nodewise regression would give an overall runtime of $O(n^2p^2)$. In contrast, our estimator (3.1) which we call the *graphical RP* has a complexity of $O(np^2)$: this is the same as computing the empirical covariance matrix.

When p is very large indeed, even performing a computation for each of the $p(p-1)/2$ potential edges in the graph can be problematic. We show that by employing fast methods for large inner product search, large entries in $\hat{\mathbf{G}}$ can be identified with a computational cost that is subquadratic in p. Specifically, we illustrate that using the *xyz* algorithm (Thanei et al., 2016b) can give an overall runtime of $O(Knp)$ where K may be chosen by the user depending on the available computational budget; we show empirically that choosing $K \approx n$ can give similar results to regular graphical RP.

We illustrate graphical RP's behaviour on simulated and real data. In particular, we apply our method to a collection of 13 independent gene expression datasets for which known confounders are available ($p \approx 15,000$, $n \approx 400$). Using an external source of validation, we demonstrate that the method is especially robust to confounding and produces estimates that are more biologically relevant than those of other state-of-the-art approaches, at a fraction of their computational cost: performing graphical model selection on these datasets using graphical RP (in combination with *xyz*) comes with a runtime in the order of a second compared to a few hours for neighbourhood selection with the Lasso.

3.1.1 Related work

As hinted at in the introduction, performing covariance selection (Dempster, 1972) when p is larger than n is a task which has garnered considerable attention over the past few decades. In such settings, the covariance matrix is singular. This prevents a straightforward computation of its inverse and prompts the need for additional assumptions. In the Gaussian setting, the one-to-one correspondence between the edges of the conditional independence graph and the non-zero entries of the precision matrix (Lauritzen, 1996) has motivated the use of sparsity as a criterion to select a parsimonious model, and thus address the problems inherent to high-dimensionality.

Two families of approaches for the estimation of sparse inverse covariance matrices have emerged in the literature. Some estimators penalise the Gaussian log-likelihood or the empirical risk (Friedman et al., 2008; Rothman et al., 2008; Yuan and Lin, 2007a; Zhang and Zou, 2014) – a noteworthy example being the so-called graphical lasso (Friedman et al., 2008; Yuan and Lin, 2007b). The graphical lasso (denoted GLasso henceforth) is a penalised maximum likelihood estimator (MLE) which adds a ℓ_1 penalty to the entries of the precision matrix; its theoretical properties are studied in (Ravikumar et al., 2010). Another family of methods proceeds by carrying out nodewise regressions – one for each of the p variables – in order to uncover the conditional dependencies of each node on the remaining ones (Cai et al., 2016; Cai et al., 2011; Meinshausen and Bühlmann, 2006; Ren et al., 2015; Yuan, 2010) Common regression methods to perform these p regressions are the Lasso, as in

Meinshausen and Bühlmann (2006), or the Dantzig selector. As is common in the literature, we will refer to neighbourhood selection with the Lasso by *NS*.

Providing theoretical guarantees for graphical model selection in the presence of latent variables is even more challenging: some edges might be included in the model as a byproduct of confounding, thus raising identifiability issues. As mentioned earlier, one possibility is to assume that the precision matrix of interest is sparse *conditional* on the unobserved variables. In the Low-Rank plus Sparse (LRpS) estimator suggested by Chandrasekaran et al. (2012), the inverse covariance matrix and the matrix which summarises the effect of the latent variables are estimated simultaneously using a regularised MLE which penalises a weighted combination of the ℓ_1-norm and trace norm of the parameters. A key requirement of LRpS is that the latent variables be few and influential. A simpler approach which is often used in practice is to proceed in two steps: 1) estimate the first k principal components (PCs) of the data matrix; 2) perform p univariate regressions of the $n-$dimensional columns on the k PCs, and use one of the methods mentioned in the previous paragraph on the residuals. Using this approach followed by NS will be denoted *PCA*. Both LRpS and PCA require the estimation of a low-rank matrix which encodes the effect of the latent variables on the observed ones – a task which cannot be performed consistently when $p > n$ due to the convergence rate of the extreme eigenvalues of the sample covariance matrix (Johnstone, 2001). A more modest goal is to focus only on the estimation of the sparse matrix of interest. The CLIME-like estimator of Ren and Zhou (2012) estimates a Gaussian graphical model in the presence of latent variables at a rate of $\sqrt{\log(p)/n}$. In Ren et al. (2015) consistency is obtained for a similar rate but under milder conditions.

In spite of the very attractive statistical properties of some estimators, their use is sometimes made impossible by their computational cost: the base estimator might itself be slow, or it might be just slow enough to prevent its use in combination with other popular approaches such as stability selection (Meinshausen and Bühlmann, 2010; Shah and Samworth, 2013). For that reason, large values of p might constrain practitioners to settle for methods that offer poorer statistical performances, but can be used interactively on a single computer. One such method is Sure Independence Screening (SIS) which performs screening via a thresholding of the absolute marginal correlations (Fan and Lv, 2008). Another approach which will be described in detail in the next section is projection to the rowspace RP which allows screening and model selection with $O(n^2p)$ cost in the regression case and $O(np^2)$ for graphical modelling. Surprisingly, RP is robust to the presence of latent confounders allowing for variable screening in the challenging latent GGM setting with a cost similar to SIS.

3.1.2 Organisation of the paper

We first discuss the idea of projecting to the rowspace and its application to variable selection. We then briefly argue by example how this technique is robust to latent confounders. From there we show the computational efficiency of graphical RP and discuss ways to gain further speedups. All these findings we underline with simulations and experiments on real data.

3.2 Ridge, rowspace projections and latent confounding

3.2.1 Model setting

Consider the standard linear model

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{3.2}$$

with response $\mathbf{Y} \in \mathbb{R}^n$, design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, vector of coefficients $\boldsymbol{\beta} \in \mathbb{R}^p$ and errors $\boldsymbol{\varepsilon} \sim \mathcal{N}_p(0, \sigma^2 I)$. We will treat the design as random and assume here that the rows of $\mathbf{X}$ are independent with distribution $\mathcal{N}_p(0, \boldsymbol{\Sigma})$. When $p \geq n$, $\mathbf{X}$ will have full row rank almost surely. $\boldsymbol{\beta}$ itself is assumed to be sparse with only a few nonzero entries defined by the set $S = \{j | \beta_j \neq 0\}$. It is typically assumed that the sparsity parameter $s = |S|$ is comparatively small: For example $s \leq \sqrt{n}$. Our main goal is to screen variables, that is we estimate a set of candidate variables $\hat{S}$ so that $S \subset \hat{S}$ is true with high probability.

3.2.2 A brief review of ridge and rowspace projections

In the above context, the Ridge estimator for $\boldsymbol{\beta}$ with $\lambda > 0$ is given by

$$\hat{\boldsymbol{\beta}}^{\lambda} := (\mathbf{X}^T\mathbf{X} + \lambda I_p)^{-1}\mathbf{X}^T\mathbf{Y} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda I_n)^{-1}\mathbf{Y} \tag{3.3}$$

The right hand side is usually referred to as kernel Ridge and is commonly used in high-dimensional data due to its lower computational complexity than classical Ridge (left hand side). The kernel Ridge estimator allows us to define Ridge in the limit where $\lambda \to 0$:

$$\hat{\boldsymbol{\beta}} = \lim_{\lambda \downarrow 0} \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda I)^{-1}\mathbf{Y} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{Y} \tag{3.4}$$

$\hat{\beta}$ is called the Rowspace Projection estimator (RP), as it projects the signal β onto the rowspace of $\mathbf{X}$. Another interpretation of $\hat{\boldsymbol{\beta}}$ is as the minimal l_2 norm vector perfectly predicting $\mathbf{Y}$:

$$\hat{\boldsymbol{\beta}} = \underset{\mathbf{Y}=\mathbf{X}\boldsymbol{\beta}}{\operatorname{argmin}} ||\boldsymbol{\beta}||_2^2$$

Whilst $\hat{\boldsymbol{\beta}}$ is unsuitable as a genuine estimate of $\boldsymbol{\beta}$ or for prediction of Y, the ordering of $|\hat{\boldsymbol{\beta}}|$ is nevertheless informative. There have been some attempts to understand the theoretical aspects of $\hat{\boldsymbol{\beta}}$ for variable screening. In Wang and Leng (2015) the authors show that the orderings of $|\hat{\boldsymbol{\beta}}|$ are such that the true nonzero coefficients of $\boldsymbol{\beta}$ correspond to the largest entries of $|\hat{\boldsymbol{\beta}}|$:

$$\hat{S}_k = \{(i) |\, |\hat{\boldsymbol{\beta}}|_{(i)} \geq |\hat{\boldsymbol{\beta}}|_{(k)}\} \supset S$$

with high probability for $k \in \mathbb{N}$, where k is much smaller than p. The theory in (Wang and Leng, 2015) holds for orthogonal designs ($\boldsymbol{\Sigma} = I_{p \times p}$). Nevertheless, there is strong evidence that these are only theoretical limitations and in practice many more general settings allow for variable screening using RP.

One might ask why it is necessary to consider yet another tool for variable screening when there are already numerous methods such as SIS and Lasso that do a fine job? It turns out that the key advantage of RP over the Lasso and other standard high-dimensional screening techniques is its ability to cope with latent confounders.

3.2.3 Latent confounders in linear models

The presence of latent confounders is equivalent to the model where the true signal decomposes into a sparse part $\boldsymbol{\beta}$ and a dense part $\boldsymbol{\gamma}$. We can see this as follows: In the regression setting we are given a set of p predictors $X_1, ..., X_p$. We assume each predictor has an independent source of variation Z_j and there exists a latent variable H (it could be more than one) which also contributes to the variation of X_j:

$$X_j = Z_j + \alpha_j H$$

The response Y is generated through a sparse linear model $\boldsymbol{\beta}$, a contribution from H and independent noise:

$$Y_i = \boldsymbol{\beta}^T \mathbf{X}_{i.} + \delta H_i + \varepsilon_i \;\; i \in \{1, ..., n\}$$

The contribution coming from δH_i can be decomposed into variation depending on $\mathbf{X}_{i.}$ and variation $\boldsymbol{\psi}$ that is uncorrelated to $\mathbf{X}$. Define

$$\boldsymbol{\gamma} = \operatorname*{argmin}_{\mathbf{b} \in \mathbb{R}^p} \mathbb{E}[||\delta\mathbf{H} - \mathbf{X}\mathbf{b}||_2^2]$$

and so

$$\delta\mathbf{H} = \mathbf{X}\boldsymbol{\gamma} + \boldsymbol{\psi},$$

where $Cor(\mathbf{X}, \psi) = 0$. The linear model for $\mathbf{Y}$ can be rephrased as

$$\mathbf{Y} = \mathbf{X}(\boldsymbol{\beta} + \boldsymbol{\gamma}) + \boldsymbol{\psi} + \boldsymbol{\epsilon}$$

Hence latent variables increase the noise variance on one hand but also distort the sparse signal. We can still detect the correct active set if $\boldsymbol{\gamma}$ is distributed onto many variables and has small entries. Essentially this requires both the correlations within $\mathbf{X}$ and the maximal element of $\boldsymbol{\gamma}$ ($||\boldsymbol{\gamma}||_\infty$) to be bounded from above. Of course $\boldsymbol{\beta}$ is not itself identifiable, but we can still hope to recover the variables with the largest $\boldsymbol{\beta}$ coefficients provided $\beta_{\min}$ is sufficiently large compared to $\|\boldsymbol{\gamma}\|_\infty$ and $||\varepsilon||_\infty$. Subsequently we will give some intuition on why Ridge and RP are able to recover large $\boldsymbol{\beta}$ coefficients even when there is structural noise such as $\boldsymbol{\gamma}$.

3.2.4 Why RP is only weakly affected by latent confounding

In this section, we want to gain an intuitive understanding for the reason for the robustness of RP to latent confounders. For this, we compare the

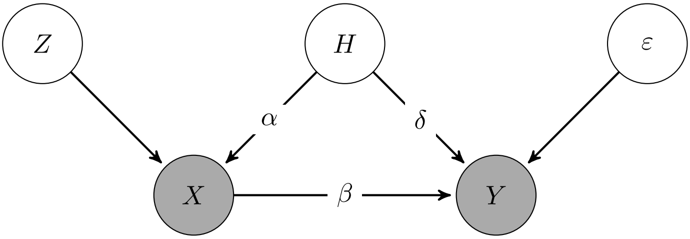


Figure 3.1: Observed variables variables are in grey. The signal from $\boldsymbol{\beta}$ is distorted by the presence of the hidden confounder H.

outcome of SIS, Lasso and RP for a simple model. We assume there is one latent variable H that has an influence on all but the first variable:

$$X_1 = Z_1 \text{ and } X_j = Z_j + H \;\; \forall j \in \{2, ..., p\}$$

The response shall be

$$Y_i = \boldsymbol{\beta}^T \mathbf{X}_{i.} + H_i \delta + \varepsilon_i = X_1 \beta_1 + H_i \delta + \varepsilon_i, \;\; i \in \{1, ..., n\}$$

so that the only significant effect to be discovered is X_1. We will argue now on approximate terms why RP is more robust to the latent variable H than SIS and the Lasso.

SIS orders variables based on marginal correlations, hence it would select X_1 as a first variable if the correlation of X_1 and Y is larger than the marginal correlation of any other variable X_j with Y:

$$\underbrace{|\mathrm{Cor}(\mathbf{X}_1, \mathbf{Y})|}_{\approx \beta_1} > \underbrace{|\mathrm{Cor}(\mathbf{X}_j, \mathbf{Y})|}_{\approx \delta} \;\; \forall j \in \{2, ..., p\}$$

If δ is considerably larger than β_1 it is unlikely that X_1 will be chosen as the strongest effect. The Lasso encounters a similar problem. It selects, just like SIS, the first variable based on the size of marginal correlations. Later in the path when λ is decreased a possible coefficient vector could look as follows:

$$\hat{\boldsymbol{\beta}}_{Lasso} \approx (\beta_1, 0, ..., 0, \delta, ..., 0)$$

The nonzero δ entry appears at a random position (at the best possible predictor for the confounding signal in Y). So if δ is really large in comparison to β_1 the ordering of the Lasso will not be informative.

RP distributes signal onto many correlated predictors because from a l_2 penalty perspective it is cheaper to use many small coefficients than a single large one to explain the signal of δH. Hence RP would likely return a coefficient vector that looks as follows:

$$\hat{\boldsymbol{\beta}}_{RP} \approx (\beta_1, \delta/(p-1), ..., \delta/(p-1))$$

And so the requirement for RP to return the correct ordering is more like $\beta_1 > \delta/p$ which is far weaker than $\beta_1 > \delta$.

The argued outcomes of $\hat{\boldsymbol{\beta}}_{Lasso}$ and $\hat{\boldsymbol{\beta}}_{RP}$ are also plausible in view of the l_1 and l_2 penalties:

$\boldsymbol{\beta}$	$\|\|\boldsymbol{\beta}\|\|_1$	$\|\|\boldsymbol{\beta}\|\|_2^2$
$\hat{\boldsymbol{\beta}}_{Lasso} \approx (\beta_1, 0, ..., 0, \delta, 0, ..., 0)$	$\|\beta_1\| + \|\delta\|$	$\beta_1^2 + \delta^2$
$\hat{\boldsymbol{\beta}}_{RP} \approx (\beta_1, \frac{\delta}{p-1}, ..., \frac{\delta}{p-1})$	$\|\beta_1\| + \|\delta\|$	$\beta_1^2 + \delta^2/(p-1)$

The Lasso does not benefit in terms of penalty from a dense coefficient vector, whereas RP distributes the latent signal over many coefficients and thus RP is likely to pick X_1 first, whereas for other approaches the orderings more crucially depend on the size of δ.

3.3 Learning graph structures with RP

3.3.1 Model setting

The preceding discussion translates directly to graphical models. Assume we are given data $\mathbf{X} \in \mathbb{R}^{n \times p}$ where each row of $\mathbf{X}$ has the distribution

$$\mathbf{X}_{i.} \sim \mathcal{N}(0, \boldsymbol{\Sigma}), \quad i \in \{1, ..., n\}$$

The conditional independence graph is defined through the set of nonzero entries of $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$. In the presence of latent confounders we assume that $\boldsymbol{\Sigma}$ decomposes as

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_0 + \boldsymbol{\Gamma}\boldsymbol{\Gamma}^T$$

where $\boldsymbol{\Gamma} \in \mathbb{R}^{p \times q}$ with $q < p$ denotes the contritbuiton of the confounders to the covariance. Our goal now is to isolate the nonzero structure of $\boldsymbol{\Omega} = \boldsymbol{\Sigma}_0^{-1}$. In the setting without confounders it is well known (Meinshausen and Bühlmann, 2006) that the set of nonzeros of $\boldsymbol{\Omega}$ can be uncovered through regressing each variable $\mathbf{X}_j$ on all others $\mathbf{X}_{-j}$ (excluding $\mathbf{X}_j$) using the Lasso. The resulting (sparse) coefficients $\boldsymbol{\beta}^j$ are estimates for the nonzero entries in the j-th column of $\mathbf{\Omega}$. This motivates the same procedure using RP (instead of Lasso) when there is an additional covariance term $\mathbf{\Gamma}\mathbf{\Gamma}^T$ for the latent confounders.

3.3.2 Nodewise RP

For each variable $\mathbf{X}_j$ we select its most important neighbours using RP. Technically this is achieved through the following estimator:

$$\hat{\boldsymbol{\beta}}^j = \mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j$$

which lets us select the neighbours of $\mathbf{X}_j$ based on the size of the absolute coefficients in $\hat{\boldsymbol{\beta}}^j$. By the arguments of the previous section $\hat{\boldsymbol{\beta}}^j$ will contain the correct ordering of neighbours and this ordering will not be strongly affected by confounders. Running this over each node j is called nodewise RP and incurs a computational cost of $\mathcal{O}(n^2p^2)$, the same cost as the nodewise Lasso. The nodewise RP estimator is denoted by a matrix $\hat{\mathbf{N}}$ that contains $\hat{\boldsymbol{\beta}}^j$ in the j-th column, where the value $\hat{N}_{jj}$ is undefined and set to one.

Through its ordering $\hat{\boldsymbol{\beta}}^j$ gives an estimate for the active set of the j-th column of Ω. However, one needs to be cautious when estimating a path of the full graph. The columns of $\hat{\mathbf{N}}$ are not necessarily comparable: The largest value in the first column of $\hat{\mathbf{N}}$ could be smaller than all entries in the second column (ignoring the diagonal). For a graphwise estimate, this would imply choosing all neighbours of the second node before choosing any neighbour of the first node. While practically possible, statistically it would be hard to trust such an estimate. This, however, happens quite frequently when using $\hat{\mathbf{N}}$ as an estimate for the full graph. To build a graph edge after edge there needs to be some way of comparing the orderings of one column to another. The simplest property that would allow us to do so is to symmetrize $\hat{\mathbf{N}}$. That is to find a diagonal matrix $\boldsymbol{D}_{\hat{\mathbf{N}}}$ so that $\hat{\mathbf{N}}\boldsymbol{D}_{\hat{\mathbf{N}}}$ is symmetric. Such an estimator is called graphwise. The next section shows that Rowspace Projections naturally fulfill this graphwise criterion.

3.3.3 Graphical RP

The graphical RP estimator is given as:

$$\hat{\mathbf{G}} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}$$

Compared to the nodewise estimate the graphical RP estimator is computationally more efficient; it requires only $\mathcal{O}(np^2)$ operations, shaving off a factor of n from the computational complexity of nodewise RP. Note that the computational complexity of $\hat{\mathbf{G}}$ is equivalent to that of SIS for graphical modelling and also the computation of the empirical covariance matrix. Also, note that $\hat{\mathbf{G}}$ literally is the projection to the rowspace.

We claim that the graphical RP estimator allows us to do build a path on the edges of the whole graph by considering the orderings of the full $\hat{\mathbf{G}}$ matrix. To see this first note that $\hat{\mathbf{G}}$ is symmetric, hence fulfilling the graphwise requirement. More importantly, each column is a valid nodewise estimate: The j-th column of $\hat{\mathbf{G}}$ is a rescaled version of $\hat{\boldsymbol{\beta}}^j$. This follows from the following Lemma:

Lemma 3.3.1. ***RP-Rescaling:*** *Define* $\hat{\boldsymbol{\beta}}^j = \mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j$ *and* $\hat{\mathbf{G}}_j = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}_{j(-j)}$. *Then there exists* $c > 0$ *such that*

$$c\hat{\boldsymbol{\beta}}^j = \hat{\mathbf{G}}_j$$

where

$$c = \frac{1}{1 + \mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}$$

Proof. See Appendix. □

This is a surprising property of both RP and also Ridge estimators. Back in the regression setting it means that by including $\mathbf{Y}$ as a predictor the resulting coefficient vector would only differ from the original one (where $\mathbf{Y}$ is not a predictor) by a scaling factor. This is an especially useful property in graphical modelling as it allows to design the nodewise regressions in a more efficient manner.

The Rescaling-Lemma tells us that the columns of $\hat{\mathbf{G}}$ are rescaled nodewise RP estimates. Therefore we call $\hat{\mathbf{G}}$ graphical RP (GRP).

3.3.4 SVD decomposition for numerical stability

An issue that arises when we normalize $\mathbf{X}$ to be centered and have unit variance is that $\mathbf{X}\mathbf{X}^T$ is not invertible, it will have rank $n-1$. One way to address this issue is to add a small λ penalty to make the inversion well conditioned:

$$\hat{\mathbf{G}}_\lambda = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda I)^{-1}\mathbf{X}$$

Another way of avoiding the singularity is to use the singular value decomposition (SVD). Decomposing $\mathbf{X}$ as $\mathbf{X} = \boldsymbol{UDV}^T$ we can compute

$$\hat{\mathbf{G}} = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X} = \boldsymbol{V}\boldsymbol{V}^T$$

And so the workflow would instead be to compute the SVD of $\mathbf{X}$ (at cost of $\mathcal{O}(n^2p)$) and then to find large entries in the inner product of $\boldsymbol{V}$ and $\boldsymbol{V}^T$. This approach has the advantage that it is numerically more stable than inverting $\mathbf{X}\mathbf{X}^T$.

3.3.5 Fast inner product computation and GRP-xyz

Each time we compute $\hat{\mathbf{G}}$ we need to compute an inner product between two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ and then we order the entries of $\boldsymbol{A}^T\boldsymbol{B}$. Possible examples for $\boldsymbol{A}$ and $\boldsymbol{B}$ are:

- **GRP:** $\boldsymbol{A} = \mathbf{X}$ and $\boldsymbol{B} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}$
- **GRP-svd:** $\boldsymbol{A} = \boldsymbol{V}^T$ and $\boldsymbol{B} = \boldsymbol{V}^T$, where $\boldsymbol{V}$ is coming from the SVD of $\mathbf{X} = \boldsymbol{UDV}^T$.

Choices for $\boldsymbol{A}$ and $\boldsymbol{B}$ could also include custom normalizations. The computation of the inner product between $\boldsymbol{A}$ and $\boldsymbol{B}$ costs $\mathcal{O}(np^2)$ and finally the search for large entries in $\boldsymbol{A}^T\boldsymbol{B}$ costs around $\mathcal{O}(p^2\log(p))$. So the total cost of GRP is $\mathcal{O}(np^2)$, assuming computations of $\boldsymbol{A}$ and $\boldsymbol{B}$ cost at most $\mathcal{O}(n^2p)$. Hence, the multiplication and search steps are the run time bottlenecks of GRP. These two steps are equivalent to large inner product search - i.e. searching for large inner products between the columns of $\boldsymbol{A}$ and $\boldsymbol{B}$. There has recently been some effort into research in this area of optimisation. For example, the xyz-algorithm (Thanei et al., 2016b) with high probability can list all large inner products between $\boldsymbol{A}$ and $\boldsymbol{B}$ in time $\mathcal{O}(npL)$, where L can be set by the user and the larger this value is chosen the more accurate this search gets. We will mostly use $L = n$, which yields a run time of $\mathcal{O}(n^2p)$ for GRP. We call this version GRP-xyz.

The xyz-algorithm finds large inner products by first transforming the data to binary form, that is $\boldsymbol{A}_{\text{bin}}, \boldsymbol{B}_{\text{bin}} \in \{-1, 1\}^{n\times p}$ are binary versions of $\boldsymbol{A}$ and $\boldsymbol{B}$. They are transformed in such a way that large inner products in $\boldsymbol{A}^T\boldsymbol{B}$ are still large in $\boldsymbol{A}_{\text{bin}}^T\boldsymbol{B}_{\text{bin}}$. Next one randomly subsamples rows of $\boldsymbol{A}_{\mathrm{bin}}$ and $\boldsymbol{B}_{\mathrm{bin}}$ and projects them to one dimension, giving vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^p$. This projection step is injective, each entry in $\boldsymbol{a}$ and $\boldsymbol{b}$ allows to reconstruct the original binary subsample. The final step is to look for equal entries in $\boldsymbol{a}$ and $\boldsymbol{b}$. Equal entries are more likely to stem from large inner products. The random subsampling and all subsequent steps are repeated L times to increase detection power

We typically choose $L = n$. Ignoring the dependence of n on p this gives an algorithm that allows to perform graphical model selection scaling linearly in p. It is computationally cheaper than computing the actual covariance $\hat{\boldsymbol{\Sigma}}$. We will demonstrate the capabilities of GRP-xyz in the next section.

3.4 Experiments

In the following experiments, we use simulated and real data to verify the claims of the previous sections.

3.4.1 Simulating a graphical model with latent confounders

X_j denotes a node in a graph with $\mathrm{Var}(X_j) = 1$. We may decompose the variance of X_j as follows:

$$1 = \mathrm{Var}(X_j) = \underbrace{\mathrm{Var}(\sum_{l \in Pa(j)} X_l A_{lj})}_{f_0} + \underbrace{\mathrm{Var}(\varepsilon_i)}_{f_1} + \underbrace{\mathrm{Var}(\sum_{k=1}^{q} H_k \gamma_{jk})}_{f_2} . \tag{3.5}$$

Here $\mathrm{Pa}(j)$ denotes the parental set of node j, $A \in \mathbb{R}^{p \times p}$ is the adjacency matrix of the graph (upper triagonal), γ_{jk} is the effect of the k-th latent variable on the j-th node. f_0 is the contribution of the actual signal, f_1 is the contribution of the noise and f_2 the contribution of the latent confounders. We construct a graph as follows:

1.) Draw a random causal order of the graph: Number the variables.

2.) Draw a parental set that fulfills the graph structure: Construct the adjacency matrix A. In a first step this is just zeros and ones, decribing the structure of the graph.

3.) For each parent X_l of X_j draw the parent coefficient $\beta_{lj} \sim N(0, 1)$

4.) Set $A_{lj} = \beta_{lj}/c$, where c is chosen so that decomposition 3.5 is fulfilled.

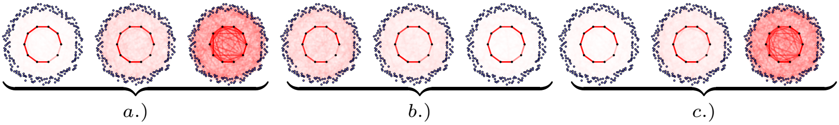


Figure 3.2: Averaged estimated graphs for SIS (a.), GRP (b.) and NS (c.). The correct graph is the circle in the center of each point cloud. The estimated edges are indicated in red. The edges are averaged over estimates from many different data realizations, strong red coloring indicates an edge often selected and more transparent coloring indicates that this edge was only chosen a few times. For each method, we test three different settings from left to right: No latent confounding, weak confounding and strong confounding. Both SIS and nodwise-Lasso estimate an arbitrarily wrong graph in the presence of strong confounders, whereas GRP is robust to latent confounding.

This procedure gives us the correctly normalized adjacency matrix A, from which we can easily compute the covariance matrix and its inverse:

$$\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1} = (\boldsymbol{I} - \boldsymbol{A})\boldsymbol{D}^{-1}(\boldsymbol{I} - \boldsymbol{A}^T),$$

Where $\boldsymbol{D} \in \mathbb{R}^{p \times p}$ is a diagonal matrix of the residual variance, hence $\boldsymbol{D} = \text{diag}(f_1)$. Finally, we add latent confounders adhering to decomposition 3.5. We simulate the graph in such a, rather involved, way because we want to control these different sources of variation. It is quite common in the literature (Luo et al., 2014) to construct an inverse covariance and then adjust the diagonal to make it symmetric positive definite. With such a construction the resulting model has a large residual variance f_1 and the signal to noise ratio f_0/f_1 varies a lot from node to node. This renders graphical estimation trivial for a few nodes and impossible for most others. Our framework allows us to directly control the residual variances and we can explicitly set the difficulty of the problem. We simulate from the following different graphs:

- **A.) Random graph:** If $i \neq j$ then $\Omega_{ij} \neq 0$ with probability p_s.
- **B.) Cluster:** Clusters of different sizes. If (i, j) belong to the same cluster, then they are connected with probability κ.

- **C.) Banded:** If $|i-j| < \rho$ then $\Omega_{ij} = 1$, where ρ is called the bandwidth. A chain graph would be a special case with $\rho = 1$.

To get a feeling for realistic values of f_0, f_1 and f_2 we compare the histogram of the simulated covariances to the histogram of real covariances.

We find that for graphs without latent variables the histogram of the covariance is skinny; there is only very few large entries and most entries are concentrated around zero. This does not compare well to real-world covariances (Figure 3.3), where there are many more large entries and they are more evenly distributed in $(-1, 1)$. When we add latent variables to the graph the distribution of the entries of the covariance matrix is similar to that of real covariances. We found that realistic values for the decomposition are for example $f_1 = 0.3$ and $f_2 = 0.6$. We are not arguing that in general latent variables are present in data, it seems however that at least for the examples above, latent confounders offer a simple way to overcome the unrealistic distribution of the covariance entries when the underlying graph is sparse.

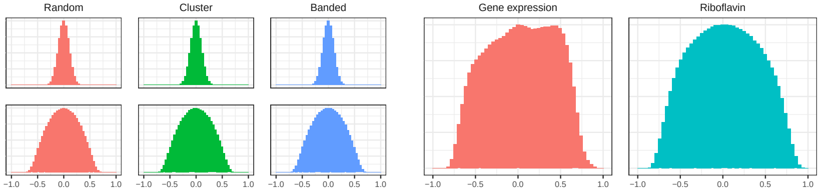


Figure 3.3: Left (2 rows): Correlation histogram of the three graph types A-C with $n = 100$ and $p = 1000$. The top row corresponds to the data without latent confounders $f_2 = 0$ and with residual variance $f_1 = 0.9$. The bottom row depicts the same covariances with $q = 5$ latents where $f_1 = 0.3$ and $f_2 = 0.6$. Right: Two real data covariances, left is a gene expression data set (Lonsdale and al., 2013) and right is the Riboflavin data set (Dezeure et al., 2015). Both of these real data sets show a similar covariance to the simulated ones where latent confounders are present.

3.4.2 Average precision

In this experiment, we compare different methods for selecting graphical models. The goal is to understand how the performance changes once more and more latent factors are added or their strength increases. As a key performance indicator, we use precision-recall curves. We compute

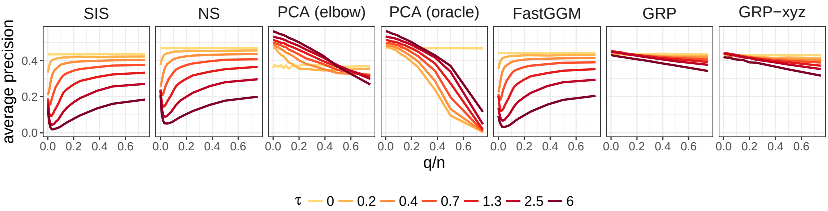


Figure 3.4: Comparison of average precision for learning the structure of a random graph. We fix the signal at $f_0 = 0.3$ and increase the number of latents to samples q/n from 0 up to 0.8. Each curve corresponds to a different value of $\tau = f_2/f_1$.

the area under these curves and compare it for different methods. This is called the average precision. An average precision close to 1 implies the method does a perfect job at estimating the graph structure, whereas a value around 0 indicates the method does not work and picks up random noise.

The experiments are designed as follows: We sample from a random graph with $n = 100$, $p = 1000$, sparsity roughly $s = 10$ and a signal strength of $f_0 = 0.3$. Let us encode the tradeoff between independent noise and latent variance by the ratio $\tau = f_2/f_1$. $\tau = 0$ means there are no latents affecting the data, τ large means that latents have a strong influence on the data. We vary both τ in an interval of $\tau \in [0, 6]$ and q the number of latents $q/n \in [0, 0.8]$. Each method is assessed according to the following two characteristics:

- Robustness to q: How do different numbers of latent variables affect the average precision.
- Robustness to τ: How do different strengths of latents affect the average precision.

We note the following key observations (Figure 3.4):

- Both nodewise Lasso and SIS react strongly to latents when their number is small ($q < 5$). After this initial drop the average precision increases as the number of latents increases while τ remains fixed. This comes from the fact that more latents with the same strength are less correlated and so the latent variation feels more like additional noise. If the number of latents keeps increasing we end up in the setting where the residual noise variance equals $f_1 + f_2$.

- In contrast to SIS and nodewise Lasso the PCA based methods struggle more when there are many latent confounders. Then the latent signal spreads over many principal components and is much harder to remove without destroying actual signal.
- GRP is the most stable of all the methods: Its performance decreases only slightly when the number of latents or their strength is increased. Compared to PCA it suffers in accuracy when there are only a few latents. Also, note that eventhough GRP-xyz is an approximate version of GRP, it nevertheless has a similar accuracy and behaviour to GRP overall.

3.4.3 Asymptotic run time behaviour

High-dimensional graphical model selection is computationally expensive. GRP and in particular GRP-xyz offers run time improvements over many traditional methods, which allows us to consider sparse graphs with node counts beyond 10^4. In this experiment, we want to demonstrate the computational advantage of GRP and its variants.

We simulate from a chain graph. For timing and memory reasons we don't consider a full chain graph, but instead simulate a chain of length n and the rest of the graph remains empty. We slowly increase n while p ranges from 100 up to 30000. The pairings of n and p are approximately $n \approx 2\sqrt{p}$.

The experiment shows (Figure 3.5) clearly how SIS and GRP have much lower run time than the other methods. The run time benefit of GRP-xyz becomes apparent as the dimensionality increases: It takes roughly 30 seconds to fit a graph of size $p = 30000$. More sophisticated methods such as LRpS or Glasso cannot be used within a reasonable time on data sets larger than $p = 500$ due to their restrictive run time behaviour. In general, most method's run time does not seem to be affected by the presence of latents except for FastGGM where the EM-type algorithm seems to have convergence issues in the presence of strong latents. The practical run time behaviour of GRP, its robustness to latent variables and its potential speed up through xyz opens up many areas of applications where the number of variables above 10^4 is the norm.

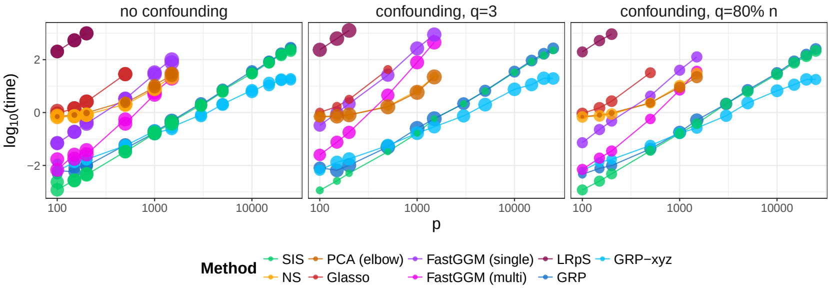


Figure 3.5: Comparison of run time (log-scale to basis of 10) of different methods and their average precision while fixing the signal strength to $f_0 = 0.2$. The average precision is indicated by the size of the dots. Depicted are three settings of a chain graph: One with no latents $\tau = 0$ (left), another one with $\tau = 1$ and $q = 3$ latents (center) and a third graph with $\tau = 1$ and many latents $q = 0.8n$. Note that not all methods can be run on a graph of size $p = 30000$ due to their massive run time and memory requirement for such data sizes.

3.5 Real data example: modelling gene regulatory networks across multiple tissues

We illustrate a few key properties of our approach on a collection of datasets made publicly available by the GTEX consortium (Lonsdale and al., 2013). The performance of the various methods is first assessed in terms of sensitivity to the addition of confounding. We then use an external source of validation – the gene ontology (Ashburner, 2000) – in order to measure the biological relevance of the estimated graphs.

3.5.1 The GTEX datasets

The GTEX consortium conducted a large-scale RNA-Seq experiment which generated gene expression data for more than fifty human tissues coming from hundreds of human donors. The resulting datasets have two features that make them particularly well-suited for demonstrating the behaviour of the suggested estimator. First, they are very high-dimensional since the number of genes expressed in any given tissue (p) is typically around $15,000$, while the sample size (n) is at most 500. Second, the kind of confounding we assume in this paper is ubiquitous in genomics applications (Gagnon-Bartsch et al., 2013; Leek and

Storey, 2007; Stegle et al., 2012). In their own analyses, the GTEX consortium accounted for confounding by leveraging external information (*e.g.* gender, genetic relatedness between donors) and by inferring some confounders from the data itself using PEER (Stegle et al., 2012). The resulting sets of covariates (around 60 per tissue) are available on their website.

For each tissue – *e.g.* whole blood, lung, thyroid – we use an $n \times p$ data matrix $\mathbf{X}$ of gene expression levels, along with the $n \times k$ matrix of covariates which are deemed confounders by the consortium. Each of the datasets available had already been fully processed, normalised and filtered by GTEX. We further filtered out tissues with $n < 300$. Thirteen tissues passed this filter, with a ratio n/p ranging between 0.02 and 0.03 and values of p ranging between $14,337$ and $16,306$. Such large values of p prevented us from running some of the methods mentioned in our numerical simulations. We restricted ourselves to neighbourhood selection (NS) with the Lasso, SIS, FastGGM, GRP and GRP-xyz with $L = 200$.

For a fixed tissue, we ordered the covariates according to how "influential" they are in $\mathbf{X}$ by performing a univariate linear regression of each of the p gene expression levels on each of the k covariates. The degree of influence of a covariate was defined to be its average r-squared over all p regressions, thus allowing us to rank covariates from strongest to weakest in terms of variance explained. Thanks to this ranking, we defined a sequence of datasets $\mathbf{X}^j$ as follows. For $j \in \{0, \dots, k\}$, we let $\mathbf{X}^j = \mathbf{X} - \mathbf{H}_j(\boldsymbol{H}_j^T \boldsymbol{H}_j)^{-1} \boldsymbol{H}_j^T \mathbf{X}$, where $\boldsymbol{H}_j$ is the $n \times j$ matrix with columns the j most influential covariates for the tissue under consideration. Thus, $\mathbf{X}^0$ denotes the "raw", or confounded, data. At the other end, $\mathbf{X}^k$ is the "unconfounded" dataset, the one that is typically in use in genomics applications (provided enough external information is available).

For any given tissue, each of the aforementioned methods yields a sequence of graphs (also called graph path) with an increasing number of edges. Consequently, for a given method M, we denote by $\mathbf{G}_M(i, \mathbf{X})$ the graph with i edges returned by method M when applied to dataset $\mathbf{X}$. For example, if M is neighbourhood selection, then $\mathbf{G}_{NS}(i, \mathbf{X})$ is obtained by reducing the value of the shrinkage parameter λ until the estimated graph has exactly i edges. Likewise, if $M \in$ {GRP, GRP-xyz, SIS, FastGGM}, $\mathbf{G}_M(i, \mathbf{X})$ is obtained by thresholding the absolute value of the coefficients in such a way that exactly i entries are above the threshold. Given any two graphs – $\mathbf{G}_1$ and $\mathbf{G}_2$, say – we measure their similarity by computing the Jaccard index of their edge sets. We write $\mathcal{J}(\mathbf{G}_1, \mathbf{G}_2)$.

3.5.2 Results

We first sought to assess the sensitivity of the various methods to the addition of confounding. To that end, we looked at the similarity between the paths estimated in the confounded and the unconfounded datasets. More precisely, for each tissue and each method, we computed $\mathrm{Sim}_{M,i,j} := \mathcal{J}(\mathbf{G}_M(i, \mathbf{X}^j), \mathbf{G}_M(i, \mathbf{X}^k))$ as a function of i, and for a few values of j : 0 (the raw dataset), $5, 10, 30$. An overall measure of stability, $\mathrm{AvgSim}_{M,i,j}$, is obtained by averaging the value of $\mathrm{Sim}_{M,i,j}$ over the 13 distinct tissues. In Figure 3.6 **a)**, we plot $\mathrm{AvgSim}_{M,i,j}$ for the first 100 graphs entering the path of each method (in the supplementary materials, a breakdown by tissue is also given). Unsurprisingly, as j gets closer to k, the datasets $\mathbf{X}^j$ and $\mathbf{X}^k$ become more and more similar and the Jaccard index between the estimates gets closer to 1. GRP's Jaccard index also improves as j increases, but very little since it is already rather high when $j = 0$. Given that a number of the covariates were estimated using *external* data (*e.g.* gender and genotype data), this is an encouraging result. It tends to show that GRP, while helped by the lack of confounding, remains consistent in its estimates. In contrast, SIS's path is strongly influenced by the presence of the covariates, with a Jaccard similarity between raw and unconfounded data nearing zero. Consistently returning the same set of edges irrespective of confounding does not imply anything about the quality of the estimates. It is possible that NS's and SIS's paths become better and better as covariates get regressed out. To test this hypothesis, we scored the graphs using a reference dataset: the gene ontology (Ashburner, 2000). Briefly, the gene ontology is a popular database which allows the annotation of each gene by a set of *terms* classified into three categories: cellular components, molecular function and biological process. Thus, genes that tend to perform similar functions or to interact are expected to be annotated by similar terms. By mapping each node of each graph to its GO terms, one can compute an "enrichment statistic" reflecting whether the graph contains edges between related genes more often than would be expected in a random graph with a similar topology (such a graph has an expected statistic of 1) (Frot et al., 2018, §6.2). By randomly permuting the an-

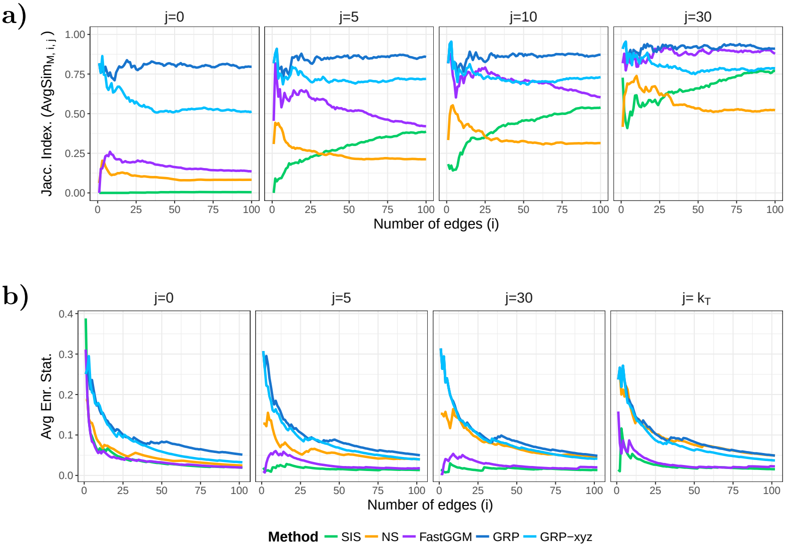


Figure 3.6: **a)** AvgSim$_{M,i,j}$. Jaccard index between the edge sets returned by the methods in the unconfounded and confounded datasets, as a function of the number of edges i (x-axis) and the number of covariates being regressed j. Results are averaged over 13 distinct tissues. **b)** Average enrichment statistic, as a function of the number of edges i, and the number of covariates being regressed j. For $j = k$, all available covariates are regressed out. Values of the enrichment statistic are first normalised within the range $[0, 1]$ for each tissue, and then averaged over tissues.

notations, empirical p-values can also be obtained. We computed this statistic for the first 100 graphs entering the path of each method, in each tissue. To make results comparable across tissues, we then divided the values of the statistic by the maximum value obtained by any of the methods in a given tissue. In summary, enrichment statistics were first normalised within the range $[0, 1]$ on a tissue basis, and then averaged across tissues. Figure 3.6 **b)** shows the value of the average enrichment statistic as a function of the number of edges and the number of covariates regressed out (a detailed plot with values for each tissue is given in the supplementary materials). First, we see that for most methods the first few edges entering the path tend to be of high quality. Second, GRP and GRP-xyz yield graphs that are at least as biologically relevant as other methods for all values of i, especially in the presence of confounding. Even though GRP-xyz appears less stable than GRP in Figure 3.6 **a)**, its performance according to the statistic is very similar to GRP's, which indicates that both methods are capable of picking up the same signal in these rather noisy data sets. We also note how the performance of NS steadily increases as more and more covariates are regressed out until it equals GRP's. This tends to confirm that the covariates estimated by the GTEX consortium were indeed masking true biological signal. Considering the speed-up provided by GRP-xyz, it is encouraging to see that its performance in the raw dataset is on par with NS's in the unconfounded one. On these datasets, the runtime of GRP-xyz was of the order of a second while GRP's was in the minutes; NS's and FastGGM's were in the hours up to a day. Finally, we recall once more that some of the covariates are estimated using external information that may not always be available in other applications. Thus, although NS and GRP offer similar performances when all covariates are accounted for, one would expect GRP to be more applicable in practice since it performs almost as well in the raw dataset on an unprocessed data and hence GRP is not in need of expert or domain knowledge in order to find and remove confounders. This potentially saves a lot of tedious human labor.

3.6 Appendix

3.6.1 Proof of the RP-Rescaling lemma

Proof. Use the rank one update formula:

$$\begin{aligned}
(\mathbf{X}\mathbf{X}^T)^{-1} &= (\mathbf{X}_{-j}\mathbf{X}_{-j}^T + \mathbf{X}_j\mathbf{X}_j^T)^{-1} \\
&= (\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1} - \frac{(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}}{1+\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}
\end{aligned}$$

$\hat{\mathbf{G}}_j$ is given by

$$\begin{aligned}
\hat{\mathbf{G}}_j &= \mathbf{X}_{-j}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}_j \\
&= \mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T + \mathbf{X}_j\mathbf{X}_j^T)^{-1}\mathbf{X}_j \\
&= \mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j \\
&- \frac{1}{1+\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}\mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j \\
&= \mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j \\
&- \frac{\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}{1+\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}\mathbf{X}_{-j}^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j \\
&= \hat{\boldsymbol{\beta}}_j - \frac{\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}{1+\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}\hat{\beta}_j \\
&= \hat{\boldsymbol{\beta}}_j\frac{1}{1+\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}
\end{aligned}$$

And so $c = \frac{1}{1+\mathbf{X}_j^T(\mathbf{X}_{-j}\mathbf{X}_{-j}^T)^{-1}\mathbf{X}_j}$ □

Bibliography

Achlioptas, D. (2003). "Database-friendly random projections: Johnson-Lindenstrauss with binary coins." *Journal of Computer and System Sciences.*

Agarwal, P., H. Edelsbrunner, O. Schwarzkopf, and E. Welzl (1991). "Euclidean minimum spanning trees and bichromatic closest pairs." *Discrete & Computational Geometry.*

Ailon, N. and B. Chazelle (2006). "Approximate Nearest Neighbors and the fast Johnson-Lindenstrauss Transform." *Proceedings of the 38th Annual ACM Symposium on Theory of Computing.*

Arkin, Y., E. Rahmani, M. Kleber, R. Laaksonen, W. März, and E. Halperin (2014). "EPIQ: efficient Detection of SNP–SNP epistatic interactions for quantitative traits." *Bioinformatics.*

Ashburner, M. et al. (2000). "Gene Ontology: Tool for the unification of biology." *Nature Genetics.*

Bickel, P., Y. Ritov, and A. Tsybakov (2010). "Hierarchical selection of variables in sparse high-dimensional regression." *IMS Collections.*

Bien, J., J. Taylor, and R. Tibshirani (2013). "A lasso for hierarchical Interactions." *Annals of Statistics.*

Blocki, J., A. Blum, A. Datta, and O. Sheffet (2012). "The Johnson-Lindenstrauss Transform itself preserves differential privacy." *53rd Annual Symposium on Foundations of Computer Science.*

Breiman, L. (2001). "Random forests." *Machine Learning.*

Breiman, L., J. Friedman, R. Olshen, and C. Stone (1984). "Classification and regression trees."

Bühlmann, P., M. Kalisch, and L. Meier (2014). "High-dimensional statistics with a view towards applications in biology." *Annual Review of Statistics and Its Application.*

Cai, T. T., W. Liu, and H. H. Zhou (2016). “Estimating sparse precision matrix: Optimal rates of convergence and adaptive estimation.” *Annals of Statistics.*

Cai, T., W. Liu, and X. Luo (2011). “A constrained ℓ_1 minimization approach to sparse precision matrix estimation.” *Journal of the American Statistical Association.*

Candès, E. J., X. Li, Y. Ma, and J. Wright (2011). “Robust principal component analysis?” *Journal of the ACM.*

Chandrasekaran, V., S. Sanghavi, P. A. Parrilo, and A. S. Willsky (2011). “Rank-sparsity incoherence for matrix decomposition.” *SIAM Journal on Optimization.*

Chandrasekaran, V., P. A. Parrilo, and A. S. Willsky (2012). “Latent variable graphical model selection via convex optimization.” *Annals of Statistics.*

Cook, R. D. (1977). “Detection of influential observations in linear regression.” *Technometrics.*

Dasgupta, S. and A. Gupta (2003). “An elementary proof of a theorem of Johnson and Lindenstrauss.” *Random Structures and Algorithms.*

Davie, A. and A. Stothers (2013). “Improved bound for complexity of matrix multiplication”. *Proceedings of the Royal Society of Edinburgh: Section A Mathematics.*

Dempster, A. (1972). “Covariance selection”. *Biometrics.*

Dezeure, R., P. Bühlmann, L. Meier, N. Meinshausen, et al. (2015). “High-dimensional Inference: Confidence intervals, p-values and R-software hdi.” *Statistical science.*

Dhillon, P. S., D. P. Foster, and S. Kakade (2013a). “A risk comparison of ordinary least squares vs ridge regression.” *The Journal of Machine Learning Research.*

Dhillon, P., Y. Lu, D. P. Foster, and L. Ungar (2013b). “New subsampling algorithms for fast least squares regression.” *Advances in Neural Information Processing Systems.*

Efron, B., T. Hastie, I. Johnstone, and R. Tibshirani (2004). “Least angle regression.” *Annals of Statistics.*

Fan, J. and J. Lv (2008). “Sure independence screening for ultrahigh-dimensional Feature Space.” *Journal of the Royal Statistical Society, Series B.*

Friedman, J. (1991). “Multivariate adaptive regression splines.” *Annals of Statistics.*

Friedman, J., T. Hastie, and R. Tibshirani (2010). “Regularization paths for generalized linear models via coordinate descent.” *Journal of Statistical Software.*

Friedman, J., T. Hastie, and R. Tibshirani (2008). “Sparse inverse covariance estimation with the graphical lasso.” *Biostatistics.*

Frot, B., L. Jostins, and G. McVean (2018). “Graphical model delection for gaussian conditional random fields in the presence of latent variables.” *Journal of the American Statistical Association.*

Gagnon-Bartsch, J. A., L. Jacob, and T. P. Speed (2013). *Removing unwanted variation from high-dimensional data with negative controls.* Tech. rep. Department of Statistics, University of California at Berkeley.

Hao, N. and H. Zhang (2014). “Interaction screening for ultrahigh-dimensional data.” *Journal of the American Statistical Association.*

Indyk, P. and R. Motwani (1998). “Approximate nearest neighbors: towards removing the curse of dimensionality.” *Proceedings of the 30th Annual ACM Symposium on Theory of Computing.*

Johnson, W. and J. Lindenstrauss (1984). “Extensions of lipschitz mappings into a hilbert space.” *Contemporary Mathematics: Conference on Modern Analysis and Probability.*

Johnstone, I. (2001). “On the distribution of the Largest eigenvalue in principal components analysis.” *Annals of Statistics.*

Kabán, A. (2014). “New bounds on compressive linear least squares regression”. *Artificial Intelligence and Statistics.*

Kemmeren, P. and et al. (2014). “Large-scale genetic perturbations reveal regulatory networks and an abundance of gene-specific repressors.” *Cell.*

Knutti, R., D. Masson, and A. Gettelman (2013). “Climate model genealogy: Generation CMIP5 and how we got there.” *Geophysical Research Letters.*

Kong, Y., D. Li, Y. Fan, and J. Lv (2016). “Interaction pursuit with feature screening and selection.” *arXiv:1605.08933.*

Lauritzen, S. (1996). “Graphical models.” *Oxford University Press.*

Le Gall, F. (2012). “Faster algorithms for rectangular matrix multiplication.” *53rd Annual Symposium on Foundations of Computer Science.*

Leek, J. T. and J. D. Storey (2007). “Capturing heterogeneity in gene expression studies by surrogate variable analysis.” *PLoS Genetics.*

Leskovec, J., A. Rajaraman, and J. Ullman (2014). “Mining of massive data sets”. *Cambridge University Press.*

Lonsdale, J. and et al. (2013). “The genotype-tissue expression (GTEx) project.” *Nature Genetics.*

Lu, Y., P. Dhillon, D. P. Foster, and L. Ungar (2013). “Faster ridge regression via the subsampled randomized Hadamard Transform.” *Advances in Neural Information Processing Systems.*

Luo, S., R. Song, and D. Witten (2014). “Sure screening for gaussian graphical models.” *arXiv:1407.7819.*

Mahoney, M. W. and P. Drineas (2009). “CUR matrix decompositions for improved data analysis.” *Proceedings of the National Academy of Sciences.*

Maillard, O. A. and R. Munos (2009). “Compressed least-squares regression”. *Advances in Neural Information Processing Systems.*

Marx, V. (2013). “The big challenges of big data.” *Nature.*

Marzetta, T., G. Tucci, and S. Simon (2011). “A random matrix theoretic approach to handling singular covariance estimates.” *IEEE Trans. Information Theory.*

McWilliams, B., G. Krummenacher, M. Lučić, and J. M. Buhmann (2014a). “Fast and robust least squares estimation in corrupted linear models.” *Advances in Neural Information Processing Systems.*

McWilliams, B., C. Heinze, N. Meinshausen, G. Krummenacher, and H. P. Vanchinathan (2014b). “LOCO: Distributing ridge regression with random projections.” *arXiv:1406.3469.*

Meinshausen, N. and P. Bühlmann (2006). “High-dimensional graphs and variable selection with the lasso.” *Annals of Statistics.*

– (2010). “Stability selection.” *Journal of the Royal Statistical Society, Series B.*

Petrov, V. (1964). “On local limit theorems for sums of independent random variables.” *Theory of Probability & Its Applications.*

Ravikumar, P., M. Wainwright, G. Raskutti, and B. Yu (2010). “High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence.” *Annals of Statistics.*

Ren, Z. and H. H. Zhou (2012). “Discussion: latent variable graphical model selection via convex optimization.” *Annals of Statistics.*

Ren, Z., T. Sun, C.-H. Zhang, H. H. Zhou, et al. (2015). “Asymptotic normality and optimalities in estimation of large gaussian graphical models.” *Annals of Statistics.*

Rothman, A., P. Bickel, E. Levina, and J. Zhu (2008). “Sparse permutation invariant covariance estimation.” *Electronic Journal of Statistics.*

Sedgewick, R. (1998). “Algorithms in C.”

Shah, R. (2016). “Modelling interactions in high-dimensional data with backtracking.” *Journal of Machine Learning Research.*

Shah, R. and N. Meinshausen (2014). “Random intersection trees.” *The Journal of Machine Learning Research.*

Shah, R. and R. J. Samworth (2013). “Variable selection with error control: another Look at stability selection.” *Journal of the Royal Statistical Society, Series B.*

Shamos, M. and D. Hoey (1975). “Closest-point problems”. *16th Symposium on Foundations of Computer Science.*

Stegle, O., L. Parts, M. Piipari, J. Winn, and R. Durbin (2012). “Using probabilistic estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses.” *Nature Protocols.*

Strassen, V. (1969). “Gaussian elimination is not optimal.” *Numerische Mathematik.*

Thanei, G.-A. (2016). *xyz R package.*

Thanei, G.-A., C. Heinze, and N. Meinshausen (2016a). “Random projections for large-scale regression.” *Big and Complex Data Analysis, Springer.*

Thanei, G.-A., N. Meinshausen, and R. D. Shah (2016b). “The xyz algorithm for fast interaction search in high-dimensional data.” *arXiv:1610.05108.*

Thanei, G.-A., B. Frot, N. Meinshausen, and R. D. Shah (2018). “Rowspace projections for fast graphical modelling in the presence of latent confounders.” *Manuscript.*

Tibshirani, R. (1996). “Regression shrinkage and selection via the lasso.” *Journal of the Royal Statistical Society, Series B.*

Tropp, J. A. (2011). “Improved analysis of the subsampled randomized hadamard transform.” *Advances in Adaptive Data Analysis.*

Wang, X. and C. Leng (2015). “High-dimensional ordinary least dquares projection for screening variables.” *Journal of the Royal Statistical Society, Series B.*

Williams, V. (2012). “Multiplying matrices faster than coppersmith-winograd.” *Proceedings of the 44th annual ACM symposium on Theory of Computing.*

Winkelmann, B., W. März, B. Boehm, R. Zotz, J. Hager, P. Hellstern, and J. Senges (2001). “Rationale and design of the LURIC study-a resource for functional genomics, pharmacogenomics and long-term prognosis of cardiovascular disease.” *Pharmacogenomics.*

Wu, J., B. Devlin, S. Ringquist, M. Trucco, and K. Roeder (2010). "Screen and clean: a tool for identifying interactions in genome-wide association studies." *Genetic Epidemiology.*

Yuan, M. and Y. Lin (2007a). "Model selection and estimation in the gaussian graphical model." *Biometrika.*

– (2007b). "On the non-negative garrotte estimator." *Journal of the Royal Statistical Society, Series B.*

Yuan, M. (2010). "High-dimensional inverse covariance matrix estimation via linear programming." *Journal of Machine Learning Research.*

Zhang, L., M. Mahdavi, R. Jin, T. Yang, and S. Zhu (2013). "Recovering optimal solution by dual random projection." *Conference on Learning Theory.*

Zhang, T. and H. Zou (2014). "Sparse precision matrix estimation via lasso penalized D-trace loss." *Biometrika.*

Zhou, S., J. D. Lafferty, and L. A. Wasserman. (2007). "Compressed regression." *Advances in Neural Information Processing Systems.*

Zou, H. and T. Hastie (2005). "Regularization and variable selection via the elastic net." *Journal of the Royal Statistical Society, Series B.*

Curriculum Vitae

Name	Gian-Andrea Thanei
Nationality	Swiss
Citizen of	Zürich
Date of birth	June 13, 1989

Education

- **ETH Zürich**: Doctoral study
    - Department of Mathematics (November 2014 - present)
      Supervisors: *Nicolai Meinshausen and Rajen Shah (Cambridge)*
- **ETH Zürich**: Master of Sciences in Mathematics (September 2010 - July 2014)

Working Experience

- Winton Capital, Zürich; July 2017 - September 2017
- Super Computing Systems, Zürich; August 2014 - October 2014

Academic Visits

- Newton Institute, University of Cambridge, Cambridge, UK; April 2018, Host: *Rajen Shah*

Submitted papers

- G. Thanei, R. Shah and N. Meinshausen. (2016). The xyz-algorithm for fast high-dimensional interaction search. Available at: arXiv:1610.05108. Resubmitted to JMLR (minor revision).

Working papers

- G. Thanei, B. Frot, R. Shah and N. Meinshausen. Rowspace projections for fast graphical model selection under latent confounding.

Publications

- G. Thanei, C. Heinze-Deml and N. Meinshausen (2016). Random projections for large-scale regression (2016). *Big and Complex Data Analysis, Methodologies and Applications, Springer Series*
- M. Sokolov, J. Ritscher, N. MacKinnon, J. Bielser, D. Brühlmann, D. Rothenhäusler, G. Thanei, M. Soos, M. Stettler, J. Souquet, H. Broly, M. Morbidelli and A. Butté (2016). Robust factor selection in early cell culture process development for the production of a biosimilar monoclonal antibody. *Biotechnology Progress.*

`R` packages

- **xyz:** On CRAN and github.

(Invited) Talks

- The xyz algorithm for fast interaction search in high-dimensional data. ERCIM, December 2017, London.
- The xyz algorithm for fast interaction search in high-dimensional data. EMS, July 2017, Helsinki.
- The causal feedback model in time series. March 2017, Oberwolfach.
- Fast interaction search for logistic regression. Google Workshop, April 2016, Zürich.

- Optimal projections for interaction search. March 2016, Oberwolfach.