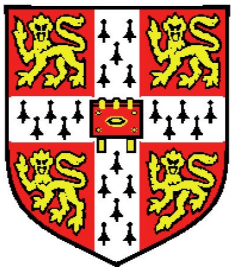
Topics in High-dimensional and Large-scale Data Analysis



Rajen Dinesh Shah

Trinity College and Statistical Laboratory

University of Cambridge

A thesis submitted for the degree of

Doctor of Philosophy

Topics in High-dimensional and Large-scale Data Analysis

Rajen Dinesh Shah

Trinity College and Statistical Laboratory
University of Cambridge

A thesis submitted for the degree of
Doctor of Philosophy

This thesis concerns the analysis of high-dimensional and large-scale data that have become ubiquitous in today's information-driven age. It consists of four main chapters. The first studies the problem of variable selection, where out of potentially thousands of measured variables, one wishes to select just a few that are relevant for a particular phenomenon of interest. Here, we develop further methodology and theory for Stability Selection, an important variable selection technique introduced in Meinshausen and Bühlmann (2010) that provides an upper bound on the expected number of irrelevant variables selected. Unfortunately the bound requires a strong exchangeability condition and it can be rather weak at times. We introduce a version of the bound that does not require exchangeability assumptions, and in addition, under some mild conditions, we obtain tighter bounds that can be used by practitioners to yield more true discoveries for the same level of error tolerance.

In Chapter 2 we consider the problem of high-dimensional regression when there may be interacting variables. We introduce a new method that searches for interactions in a hierarchical fashion. The procedure is very general: it can be incorporated into many high-dimensional regression algorithms for fitting only additive models. It is computationally fast, making use of parallel processing, and can deal with situations where there may be thousands of variables or more. We also study some theoretical properties of our method when used in conjunction with the Lasso (Tibshirani, 1996).

In Chapter 3, we return to the problem of detecting interactions, but this time in the context of classification with sparse binary data. We develop a method that is able to uncover high order interactions without requiring that some of their lower order interactions are also informative. The computational complexity of our procedure is of order p^κ, where p is the number of predictor variables and the value of κ can reach values as low as 1 for very sparse data; in many more general settings, it will still beat the exponent s obtained when using a brute force search constrained to order s interactions.

In Chapter 4 we study large-scale regression analysis with sparse data where both the number of variables and the number of observations may be large and in the order of millions or more. Our approach for dealing with this data is based on b-bit min-wise hashing (Li and König, 2011), a dimensionality reduction technique for sparse binary matrices. We propose a variant that also handles the real-valued case and allows for the construction of variable importance measures. For linear and logistic models, we give finite-sample bounds on the prediction error of procedures that perform regression in the new lower-dimensional space after applying our dimension reduction. We also show that ordinary least squares or ridge regression applied to the reduced data can allow us to capture interactions in the original data.

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text. No part of this dissertation is substantially the same as any that I have submitted or will be submitting for a degree or diploma or other qualification at this or any other University.

Chapter 1 is joint work Richard Samworth and Chapters 3 and 4 are joint work with Nicolai Meinshausen (ETH Zürich). Chapter 1 has appeared in the *Journal of the Royal Statistical Society, Series B* as Shah and Samworth (2013). The remaining chapters have all been submitted for publication.

Rajen Shah

Cambridge, November 2013

Acknowledgements

There are many people without whom this work would not have been possible. Firstly, I am indebted to my PhD supervisor, Richard Samworth, for many things including persuading me to do a PhD in Statistics in the first place! Without his encouragement this PhD dissertation would never have been started, let alone completed. I feel very lucky to have had the opportunity to collaborate with him during my PhD, and his guidance has been invaluable throughout the process. I am extremely grateful to Nicolai Meinshausen and Peter Bühlmann, for arranging visits, and giving me the honour of working together with them. Yining Chen and Pawel Zaczkowski have been a constant source of advice over the last three years and their enthusiasm has made doing a PhD a much more enjoyable experience for me. I would also like to thank Bodhisattva Sen and Simon Tavaré for their sound comments and helpful suggestions. While working in the Centre for Mathematical Sciences, I have benefited from the friendly environment created by people like Julia Blackwell, Tim Cannings, Alexandra Carpentier, Moritz Dümbgen, Robin Evans, Arlene Kim, John Shimmon, Arun Thillaisundaram, Ashok Thillaisundaram, Liyan Yu, Yi Yu and Lee Zhuo Zhao. Finally, I would like to thank my family, for their unwavering love and support.

Contents

Chapter 1

Variable selection with error control: Another look at Stability Selection

1.1 Introduction

The problem of variable selection has received a huge amount of attention over the last 15 years, motivated by the desire to understand structure in massive datasets that are now routinely encountered across many scientific disciplines. It is now very common, e.g. in biological applications, image analysis and portfolio allocation problems as well as many others, for the number of variables (or predictors) p that are measured to exceed the number of observations n. In such circumstances, variable selection is essential for model interpretation.

In a notable recent contribution to the now vast literature on this topic, Meinshausen and Bühlmann (2010) proposed Stability Selection as a very general technique designed to improve the performance of a variable selection algorithm. The basic idea is that instead of applying one's favourite algorithm to the whole dataset to determine the selected set of variables, one instead applies it several times to random subsamples of the data of size $\lfloor n/2 \rfloor$, and chooses those variables that are selected most frequently on the subsamples. Stability Selection is therefore intimately connected with bagging (Breiman, 1996, 1999) and subagging (Bühlmann and Yu, 2002).

A particularly attractive feature of Stability Selection is the error control provided by an upper bound on the expected number of falsely selected variables (Meinshausen and Bühlmann, 2010, Theorem 1). Such control is typically unavailable when applying the original selection procedure to the whole dataset, and allows the practitioner to select the threshold τ for the proportion of subsamples for which a variable must be selected in order for it to be declared significant.

However, the bound does have a couple of drawbacks. Firstly, it applies to the 'population version' of the subsampling process, i.e. to the version of the procedure that aggregates results over the non-random choice of all $\binom{n}{\lfloor n/2 \rfloor}$ subsamples. Even for n as small as 15, it is unrealistic to expect this version to be used in practice, and in fact choosing around 100 random subsamples is probably typical. More seriously, the bound is derived under a very strong exchangeability assumption on the selection of noise variables (as well as a weak one on the quality of the original selection procedure, namely that it is not worse than random guessing).

In this chapter, we develop the methodology and conceptual understanding of Stability Selection in several respects. We introduce a variant of Stability Selection, where the subsamples are drawn as complementary pairs from $\{1, \ldots, n\}$. Thus the subsampling procedure outputs index sets $\{(A_{2j-1}, A_{2j}) : j = 1, \ldots, B\}$, where each A_j is a subset of $\{1, \ldots, n\}$ of size $\lfloor n/2 \rfloor$, and $A_{2j-1} \cap A_{2j} = \emptyset$. We call this variant Complementary Pairs Stability Selection (CPSS).

The modified procedure has the advantage that the Meinshausen–Bühlmann bound holds regardless of the number of complementary pairs B chosen — even with $B = 1$. In addition, we show that there is a corresponding bound for the number of important variables excluded by CPSS.

Our results have no conditions on the original selection procedure, and in particular do not require the strong exchangeability assumption on the selection of noise variables. Indeed, we argue that even a precise definition of 'signal' and 'noise' variables is not helpful in trying to understand the properties of CPSS, and we instead state the bounds in terms of the expected number of variables chosen by CPSS that have low selection probability under the base selection procedure, and the expected number of high selection probability variables that are excluded by CPSS. See Section 1.2 for further discussion.

The bound on the number of low selection probability variables chosen by CPSS can be significantly sharpened under mild shape restrictions (e.g. unimodality or r-concavity) on the distribution of the proportion of times a variable is selected in both A_{2j-1} and A_{2j}. We discuss these conditions in detail in Sections 1.3.2 and 1.3.3 respectively, and compare both the original and new bounds to demonstrate the marked improvement.

Our improved bounds are based on new versions of Markov's inequality that hold for random variables whose distributions are unimodal or r-concave. However, it is important to note at this point that the results are not just a theoretical contribution; they allow the practitioner to reduce τ (and therefore select more variables) for the same control of the number of low selection probability variables chosen by CPSS. In Section 1.3.4, we give recommendations on how a practitioner can make use of the bounds in applying CPSS.

In Section 1.4.1, we present the results of an extensive simulation study designed to illustrate the appropriateness of our shape restrictions, and to compare Stability Selection and CPSS with their base selection procedures. Section 1.4.2 gives an application of the methodology to a colon cancer dataset.

A review of some of the extensive literature on variable selection can be found in Fan and Lv (2010). Work related more specifically to Stability Selection includes Bach (2008), who studied the Bolasso (short for Bootstrapped enhanced Lasso). This involves applying the Lasso (Tibshirani, 1996) to bootstrap (with replacement) samples from the original data, rather than subsampling without replacement. A final estimate is obtained by applying the Lasso to the intersection of the set of variables selected across the bootstrap samples.

After the work in this chapter had been completed, there have very recently been a number of proposals for constructing confidence intervals and finding p-values in high-dimensional regression settings based on the Lasso and related procedures. These include Zhang and Zhang (2013), Bühlmann (2013), van de Geer et al. (2013), Javanmard and Montanari (2013), Lockhart et al. (2013) and Meinshausen (2013).

Various authors, particularly in the machine learning literature, have considered the *stability* of a feature selection algorithm, i.e. the insensitivity of the output of the algorithm to variations in the training set; such studies include Lange et al. (2003), Kalousis et al. (2007), Kuncheva (2007), Loscalzo et al. (2009) and Han and Yu (2010). Saeys et al. (2008) consider obtaining a final feature

ranking by aggregating the rankings across bootstrap samples.

1.2 Complementary Pairs Stability Selection

In order to keep our discussion rather general, we only assume that we have vector-valued data $z_1, \ldots, z_n$ which we take to be a realisation of independent and identically distributed random elements $Z_1, \ldots, Z_n$. Informally, we think of some of the components of Z_i as being 'signal variables', and others as being 'noise variables', though for our purposes it is not necessary to define these notions precisely. Formally, we let $S \subseteq \{1, \ldots, p\}$ and $N := \{1, \ldots, p\} \setminus S$, thought of as the index sets of the signal and noise variables respectively. A *variable selection procedure* is a statistic $\hat{S}_n := \hat{S}_n(Z_1, \ldots, Z_n)$ taking values in the set of all subsets of $\{1, \ldots, p\}$, and we think of $\hat{S}_n$ as an estimator of S. As a typical example, we may often write $Z_i = (X_i, Y_i)$ with the covariate $X_i \in \mathbb{R}^p$ and the response $Y_i \in \mathbb{R}$, and our (pseudo) log-likelihood might be of the form

$$\sum_{i=1}^{n} L(Y_i, X_i^T \beta), \tag{1.1}$$

for some $\beta \in \mathbb{R}^p$. In this context, we regard $S := \{k : \beta_k \neq 0\}$ as the signal indices, $N = \{k : \beta_k = 0\}$ as noise indices. Examples from graphical modelling can also be cast within our framework. Note however that we do not require a (pseudo) log-likelihood of the form (1.1).

We define the selection probability of a variable index $k \in \{1, \ldots, p\}$ under $\hat{S}_n$ as

$$p_{k,n} := \mathbb{P}(k \in \hat{S}_n) = \mathbb{E}(\mathbb{1}_{\{k \in \hat{S}_n\}}). \tag{1.2}$$

We take the view that for understanding the properties of Stability Selection, the selection probabilities $p_{k,n}$ are the fundamental quantities of interest. Since an application of Stability Selection is contingent on a choice of base selection procedure $\hat{S}_n$, all we can hope is that it selects variables having high selection probability under the base procedure, and avoids selecting those variables with low selection probability. Indeed this turns out to be the case; see Theorem 1 below.

Of course, $\mathbb{1}_{\{k \in \hat{S}_n\}}$ has a Bernoulli distribution with parameter $p_{k,n}$, so we may view $\mathbb{1}_{\{k \in \hat{S}_n\}}$ as an unbiased estimator of $p_{k,n}$ (though $p_{k,n}$ is not a model parameter in the conventional sense). The key idea of Stability Selection is to improve on this simple estimator of $p_{k,n}$ through subsampling.

For a subset $A = \{i_1, \ldots, i_{|A|}\} \subset \{1, \ldots, n\}$ with $i_1 < \cdots < i_{|A|}$, we shall write

$$\hat{S}(A) := \hat{S}_{|A|}(Z_{i_1}, \ldots, Z_{i_{|A|}}).$$

Definition 1 (Complementary Pairs Stability Selection)**.** Let $\{(A_{2j-1}, A_{2j}) : j = 1, \ldots, B\}$ be randomly chosen independent pairs of subsets of $\{1, \ldots, n\}$ of size $\lfloor n/2 \rfloor$ such that $A_{2j-1} \cap A_{2j} = \emptyset$. For $\tau \in [0, 1]$, the Complementary Pairs Stability Selection version of a variable selection procedure $\hat{S}_n$ is $\hat{S}^{\mathrm{CPSS}}_{n,\tau} := \{k : \hat{\Pi}_B(k) \geq \tau\}$, where the function $\hat{\Pi}_B : \{1, \ldots, p\} \to \{0, \frac{1}{2B}, \frac{1}{B}, \ldots, 1\}$ is given by

$$\hat{\Pi}_B(k) := \frac{1}{2B} \sum_{j=1}^{2B} \mathbb{1}_{\{k \in \hat{S}(A_j)\}}. \tag{1.3}$$

Note that $\hat{\Pi}_B(k)$ is an unbiased estimator of $p_{k,\lfloor n/2 \rfloor}$, but, in general, a biased estimator of $p_{k,n}$. However, by means of the averaging involved in (1.3), we hope that $\hat{\Pi}_B(k)$ will have reduced variance compared with $\mathbb{1}_{\{k \in \hat{S}_n\}}$, and that this increased stability will more than compensate

for the bias incurred. Indeed, this is the case in other situations where bagging and subagging have been successfully applied, such as classification trees (Breiman, 1996) or nearest neighbour classifiers (Biau et al., 2010; Hall and Samworth, 2005; Samworth, 2012).

An alternative to subsampling complementary pairs would be to use bootstrap sampling as in Bach (2008). We have found that this gives very similar estimates of $p_{k,n}$, though most of our theoretical arguments do not apply when the bootstrap is used (the approach in Section 1.3.3.1 is an exception in this regard). In fact, taking subsamples of size $\lfloor n/2 \rfloor$ can be thought of as the subsampling scheme that most closely mimics the bootstrap (e.g. Dümbgen et al., 2012).

It is convenient at this stage to define another related selection procedure based on sample splitting.

Definition 2 (Simultaneous Selection)**.** Let $\{(A_{2j-1}, A_{2j}) : j = 1, \ldots, B\}$ be randomly chosen independent pairs of subsets of $\{1, \ldots, n\}$ of size $\lfloor n/2 \rfloor$ such that $A_{2j-1} \cap A_{2j} = \emptyset$. For $\tau \in [0, 1]$, the Simultaneous Selection version of $\hat{S}_n$ is $\hat{S}^{\mathrm{SIM}}_{n,\tau} := \{k : \tilde{\Pi}_B(k) \geq \tau\}$, where

$$\tilde{\Pi}_B(k) := \frac{1}{B} \sum_{j=1}^{B} \mathbb{1}_{\{k \in \hat{S}(A_{2j-1})\}} \mathbb{1}_{\{k \in \hat{S}(A_{2j})\}}. \tag{1.4}$$

For our purposes, Simultaneous Selection is a tool for understanding the properties of CPSS. However, the special case of $B = 1$ of Simultaneous Selection was studied by Fan et al. (2009), and a variant involving all possible disjoint pairs of subsets was considered in Meinshausen and Bühlmann (2010).

1.3 Theoretical properties

1.3.1 Worst-case bounds

In Theorem 1 below, we show that the expected number of low selection probability variables chosen by CPSS is controlled in terms of the expected number chosen by the original selection procedure, with a corresponding result for the expected number of high selection probability variables not chosen by CPSS. The appealing feature of these results is their generality: they require no assumptions on the underlying model or on the quality of the original selection procedure, and they apply regardless of the number B of complementary pairs of subsets chosen.

For $\theta \in [0, 1]$, let $L_\theta := \{k : p_{k,\lfloor n/2 \rfloor} \leq \theta\}$ denote the set of variable indices that have low selection probability under $\hat{S}_{\lfloor n/2 \rfloor}$, and let $H_\theta := \{k : p_{k,\lfloor n/2 \rfloor} > \theta\}$ denote the set of those that have high selection probability.

Theorem 1. *(i) If $\tau \in (\frac{1}{2}, 1]$, then*

$$\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta| \leq \frac{\theta}{2\tau - 1} \mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor} \cap L_\theta|.$$

(ii) Let $\hat{N}^{\mathrm{CPSS}}_{n,\tau} := \{1, \ldots, p\} \setminus \hat{S}^{\mathrm{CPSS}}_{n,\tau}$ and $\hat{N}_n := \{1, \ldots, p\} \setminus \hat{S}_n$. If $\tau \in [0, \frac{1}{2})$, then

$$\mathbb{E}|\hat{N}^{\mathrm{CPSS}}_{n,\tau} \cap H_\theta| \leq \frac{1 - \theta}{1 - 2\tau} \mathbb{E}|\hat{N}_{\lfloor n/2 \rfloor} \cap H_\theta|.$$

In many applications, and for a good base selection procedure, we imagine that the set of selection probabilities $\{p_{k,\lfloor n/2 \rfloor} : k = 1, \ldots, p\}$ is positively skewed in $[0, 1]$, with many selection

probabilities being very low (predominantly noise variables), and with just a few being large (including at least some of the signal variables). To illustrate Theorem 1(i), consider a situation with $p = 1000$ variables and where the base selection procedure chooses 50 of them. Then Theorem 1(i) shows that on average CPSS with $\tau = 0.6$ selects no more than a quarter of the below average selection probability variables chosen by $\hat{S}_{\lfloor n/2 \rfloor}$.

Our Theorem 1(i) is analogous to Theorem 1 of Meinshausen and Bühlmann (2010). The differences are that we do not require the condition that $\{\mathbb{1}_{\{k \in \hat{S}_{\lfloor n/2 \rfloor}\}} : k \in N\}$ is exchangeable, nor that the original procedure is no worse than random guessing, and our result holds for all B. The price we pay is that the bound is stated in terms of the expected number of low selection probability variables chosen by CPSS, rather than the expected number of noise variables, which we do for the reasons described in Section 1.2. If the exchangeability and random guessing conditions mentioned above do hold, then, writing $q := \mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor}|$, we recover

$$\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap N| \leq \frac{1}{2\tau - 1}\Big(\frac{q}{p}\Big)\mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor} \cap L_{q/p}| \leq \frac{1}{2\tau - 1}\Big(\frac{q^2}{p}\Big).$$

The final bound here was obtained in Theorem 1 of Meinshausen and Bühlmann (2010) for the population version of Stability Selection.

1.3.2 Improved bounds under unimodality

Despite the attractions of Theorem 1, the following observations suggest there may be scope for improvement. Firstly, we expect we should be able to obtain tighter bounds as B increases. Secondly, and more importantly, examination of the proof of Theorem 1(i) shows that our bound relies on first noting that

$$1 + \tilde{\Pi}_B(k) \geq 2\hat{\Pi}_B(k), \tag{1.5}$$

and then applying Markov's inequality to $\tilde{\Pi}_B(k)$. For equality in Markov's inequality, $\tilde{\Pi}_B(k)$ must be a mixture of point masses at 0 and $2\tau - 1$, but Figure 1.1 suggests that the distribution of $\tilde{\Pi}_B(k)$, which is supported on $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$, can be very different from this. Indeed, our experience, based on extensive simulation studies, is that when θ is close to q/p (which is where the bound in Theorem 1(i) is probably of most interest), the distribution of $\tilde{\Pi}_B(k)$ over $k \in L_\theta$ is remarkably consistent over different data generating processes, and Figure 1.1 is typical. It is therefore natural to consider placing shape restrictions on the distribution of $\tilde{\Pi}_B(k)$ that encompass what we see in practice, and that yield stronger versions of Markov's inequality. As a first step in this direction, we consider the assumption of unimodality.

Theorem 2. *Suppose that the distribution of $\tilde{\Pi}_B(k)$ is unimodal for each $k \in L_\theta$. If $\tau \in \{\frac{1}{2} + \frac{1}{B}, \frac{1}{2} + \frac{3}{2B}, \frac{1}{2} + \frac{2}{B}, \ldots, 1\}$, then*

$$\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta| \leq C(\tau, B)\, \theta\, \mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor} \cap L_\theta|,$$

where, when $\theta \leq 1/\sqrt{3}$,

$$C(\tau, B) = \begin{cases} \dfrac{1}{2(2\tau - 1 - 1/2B)} & \text{if } \tau \in (\min(\frac{1}{2} + \theta^2, \frac{1}{2} + \frac{1}{2B} + \frac{3}{4}\theta^2), \frac{3}{4}] \\ \dfrac{4(1 - \tau + 1/2B)}{1 + 1/B} & \text{if } \tau \in (\frac{3}{4}, 1]. \end{cases}$$

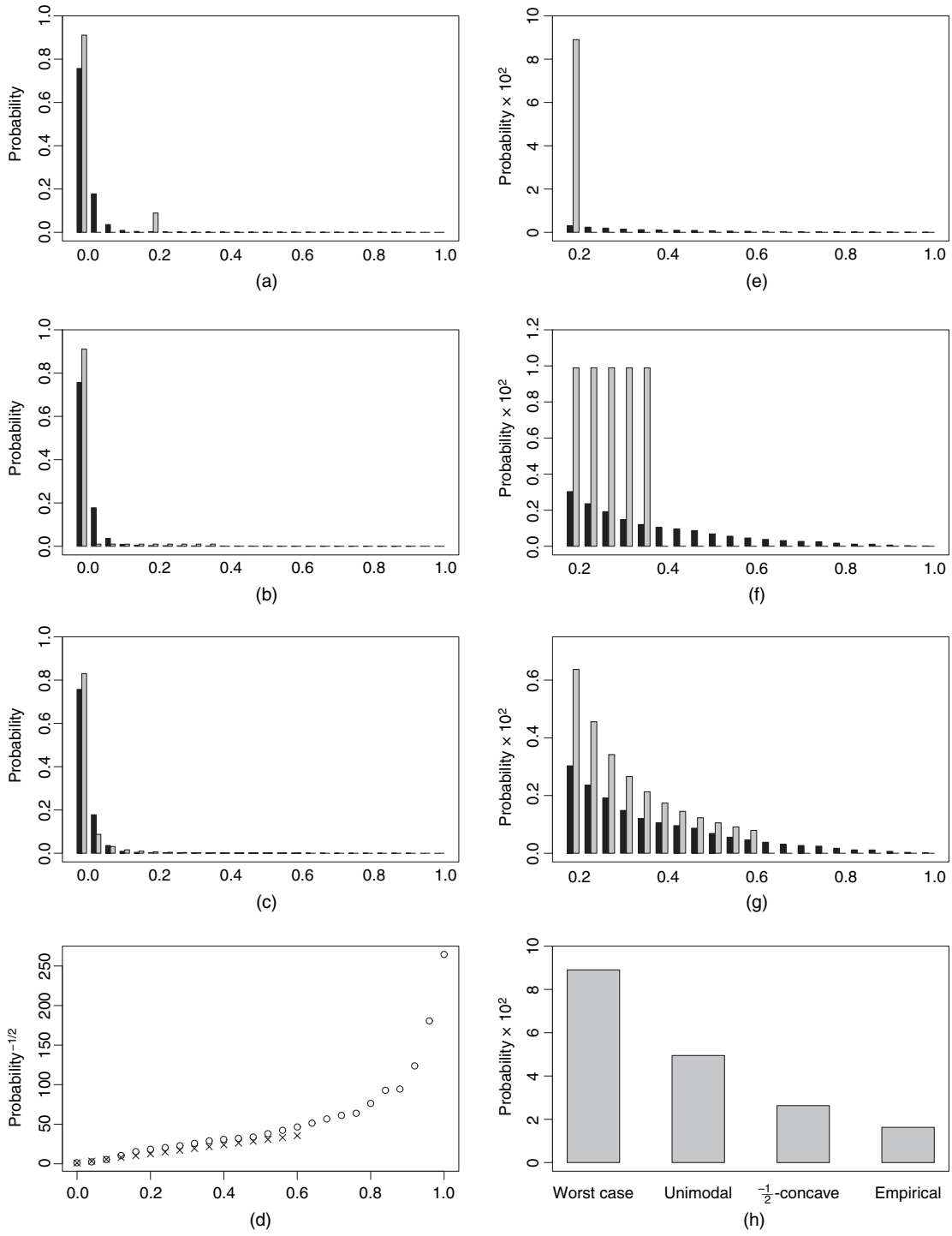


Figure 1.1: Typical example of (a)–(c) the full probability mass function and (e)–(g) zoomed in from 0.2 onwards of $\tilde{\Pi}_{25}(k)$ for $k \in L_{q/p}$ (black); alongside (a), (e) the unrestricted; (b), (f) unimodal; and (c), (g) $-1/2$-concave distributions respectively (grey), which have maximum tail probability beyond 0.2. This situation corresponds to selecting $\tau = 0.6$. Shown in (d) are the observed mass function (circles) and the extremal $-1/2$-concave mass function (crosses) on the $x^{-1/2}$ scale; in (h) tail probabilities from 0.2 onwards for each of the distributions.

The proof of Theorem 2 is based on a new version of Markov's inequality (Theorem 9 in the appendix) for random variables with unimodal distributions supported on a finite lattice. There is also an explicit expression for $C(\tau, B)$ when $\theta > 1/\sqrt{3}$, which follows from Theorem 9 in the same way, but we do not present it here because it is a little more complicated, and because we anticipate the bound when θ is (much) smaller than $1/\sqrt{3}$ being of most use in practice. See Section 1.3.4 for further discussion.

Figure 1.2 compares the bounds provided by Theorems 1 and Theorem 2 as a function of τ, for the illustration discussed after the statement of Theorem 1.

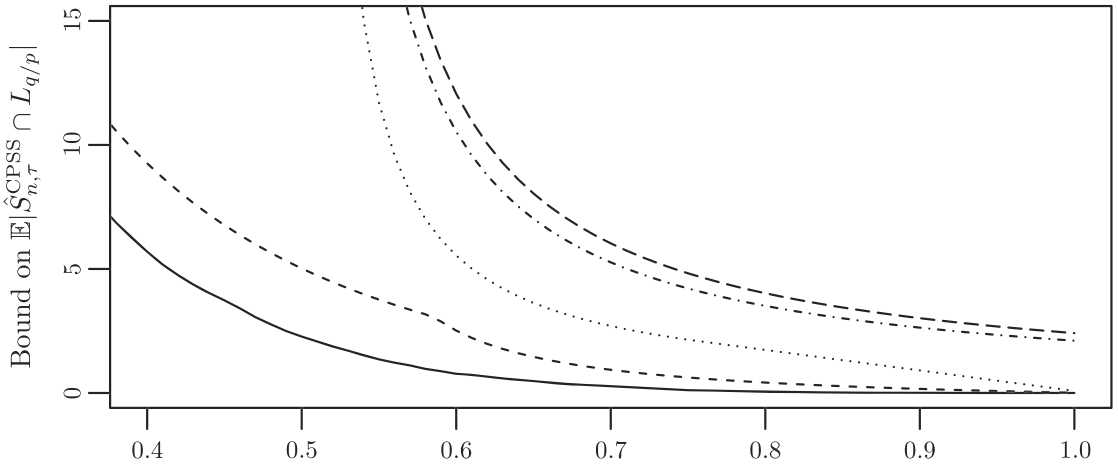


Figure 1.2: Comparison of the bounds on $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_{q/p}|$ for different values of the threshold τ: the original bound from Theorem 1 of Meinshausen and Bühlmann (2010) (long dashes), our worst case bound (dots and dashes), the unimodal bound (dots) and the r-concave bound (1.8) (short dashes). The solid line is the true value of $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_{q/p}|$ for a simulated example. In this case $p = 1000$, $q = 50$ and the number of signal variables was 8.

1.3.3 Further improvements under r-concavity

The unimodal assumption allows for a significant improvement in the bounds attainable from a naive application of Markov's inequality. However, Figure 1.1 suggests that further gains may be realised by placing tighter constraints on the family of distributions for $\tilde{\Pi}_B(k)$ that we consider, in order to match better the empirical distributions that we see in practice.

A very natural constraint to impose on the distribution of $\tilde{\Pi}_B(k)$ is log-concavity. By this, we mean that, if f denotes the probability mass function of $\tilde{\Pi}_B(k)$, then the linear interpolant to $\{(i, f(i/B)) : i = 0, 1, \ldots, B\}$ is a log-concave function on $[0, 1]$. Log-concavity is a shape constraint that has received a great deal of attention recently (e.g. Cule et al. (2010); Dümbgen and Rufibach (2009); Walther (2002)), and at first sight it seems reasonable in our context, because if the summands in (1.4) were independent, then we would have $\tilde{\Pi}_B(k) \sim \frac{1}{B}\mathrm{Bin}(B, p^2_{k,\lfloor n/2 \rfloor})$, which is log-concave.

It is indeed possible to obtain a version of Markov's inequality under log-concavity that leads to another improvement in the bound on $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta|$. However, we found that in practice, the dependence structure of the summands in (1.4) meant that the log-concavity constraint was a little too strong. We therefore consider instead the class of *r-concave* distributions, which we claim defines a continuum of constraints that interpolate between log-concavity and unimodality

(see Propositions 3 and 4 below). This constraint has also been studied recently in the context of density estimation by Seregin and Wellner (2010) and Koenker and Mizera (2010); see also Dharmadhikari and Joag-Dev (1988).

To define the class, we recall that the r^{th} generalised mean $M_r(a, b; \lambda)$ of $a, b \geq 0$ is given by

$$M_r(a, b; \lambda) = \{(1 - \lambda)a^r + \lambda b^r\}^{1/r}$$

for $r > 0$. This is also well-defined for $r < 0$ if we take $M_r(a, b; \lambda) = 0$ when $ab = 0$, and define $0^r = \infty$. In addition, we may define

$$M_0(a, b; \lambda) := \lim_{r \to 0} M_r(a, b; \lambda) = a^{1-\lambda} b^{\lambda}$$
$$M_{-\infty}(a, b; \lambda) := \lim_{r \to -\infty} M_r(a, b; \lambda) = \min(a, b).$$

We are now in a position to define r-concavity.

Definition 3. A non-negative function f on an interval $I \subset \mathbb{R}$ is *r-concave* if for every $x, y \in I$ and $\lambda \in (0, 1)$, we have

$$f((1 - \lambda)x + \lambda y) \geq M_r(f(x), f(y); \lambda).$$

Definition 4. A probability mass function f supported on $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$ is *r-concave* if the linear interpolant to $\{(i, f(i/B)) : i = 0, 1, \ldots, B\}$ is r-concave.

When $r < 0$, it is easy to see that f is r-concave if and only if f^r is convex. Let $\mathcal{F}_r$ denote the class of r-concave probability mass functions on $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$. Then each $f \in \mathcal{F}_r$ is unimodal, and as $M_r(a, b; \lambda)$ is non-decreasing in r for fixed a and b, we have $\mathcal{F}_r \supset \mathcal{F}_{r'}$ for $r < r'$. Furthermore, f is unimodal if it is $-\infty$-concave, and f is log-concave if it is 0-concave. The following two results further support the interpretation of r-concavity for $r \in [-\infty, 0]$ as an interpolation between log-concavity and unimodality.

Proposition 3. *A function f is log-concave if and only if it is r-concave for every $r < 0$.*

Proposition 4. *Let f be a unimodal probability mass function supported on $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$ and suppose both that $f(0) < \ldots < f(\frac{l}{B}) = f(\frac{l+1}{B}) = \ldots = f(\frac{u}{B})$ and that $f(\frac{u}{B}) > f(\frac{u+1}{B}) > \ldots > f(1)$, for some $l \leq u$. Then f is r-concave for some $r < 0$.*

In Proposition 11 in the appendix of this chapter, we present a result that characterises those r-concave distributions that attain equality in a version of Markov's inequality for random variables with r-concave distributions on $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$. If we assume that the distribution of $\tilde{\Pi}_B(k)$ is r-concave for all $k \in L_\theta$, using (1.5), for these variables we can obtain a bound of the form

$$\mathbb{P}(\hat{\Pi}_B(k) \geq \tau) \leq D(p^2_{k,\lfloor n/2 \rfloor}, 2\tau - 1, B, r) \leq D(\theta^2, 2\tau - 1, B, r) \tag{1.6}$$

where $D(\eta, t, B, r)$ denotes the maximum of $\mathbb{P}(X \geq t)$ over all r-concave random variables supported on $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$ with $\mathbb{E}(X) \leq \eta$. Although D does not appear to have a closed form, it is straightforward to compute numerically, as we describe in Section 1.5.4. The lack of a simple form means a direct analogue Theorem 2 is not available. We can nevertheless obtain the following bound on the expected number of low selection probability variables chosen by CPSS:

$$\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tau} \cap L_\theta| = \sum_{k \in L_\theta} \mathbb{P}(\hat{\Pi}_B(k) \geq \tau) \leq D(\theta^2, 2\tau - 1, B, r)|L_\theta|. \tag{1.7}$$

Our simulation studies suggest that $r = -1/2$ is a sensible choice to use for the bound. In other words, if f denotes the probability mass function of $\tilde{\Pi}_B(k)$, then the linear interpolant to $\{(i, f(i/B)^{-1/2}) : i = 0, 1, \ldots, B\}$ is typically well approximated by a convex function. This is illustrated in the bottom left panel of Figure 1.1 (note that the right-hand tail in this plot corresponds to tiny probabilities).

1.3.3.1 Lowering the threshold τ

The bounds obtained thus far have used the relationship (1.5) to convert a Markov bound for $\tilde{\Pi}_B(k)$ into a corresponding one for the statistic of interest, $\hat{\Pi}_B(k)$. The advantage of this approach is that $\mathbb{E}(\tilde{\Pi}_B(k)) = p^2_{k,\lfloor n/2 \rfloor}$ is much smaller than $\mathbb{E}(\hat{\Pi}_B(k)) = p_{k,\lfloor n/2 \rfloor}$ for variables with low selection probability, so the Markov bound is quite tight. However, for τ close to $1/2$, the inequality (1.5) starts to become weak, and bounds can only be obtained for $\tau > 1/2$ in any case.

To solve this problem, we can apply our versions of Markov's inequality directly to $\hat{\Pi}_B(k)$. We have found, through our simulations, that for variables with low selection probability, the distribution of $\hat{\Pi}_B(k)$ can be modelled very well as a $-1/4$-concave distribution (see Figure 1.3). That the distribution of $\hat{\Pi}_B(k)$ is closer to log-concavity than that of $\tilde{\Pi}_B(k)$ is intuitive because although the summands in (1.3) are not independent, terms involving subsamples which have little overlap will be close to independent. If we assume that $\tilde{\Pi}_B(k)$ is $-1/2$-concave and that $\hat{\Pi}_B(k)$ is $-1/4$-concave for all $k \in L_\theta$, we can obtain our best bound

$$\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta| \leq \min\{D(\theta^2, 2\tau - 1, B, -1/2), D(\theta, \tau, 2B, -1/4)\}|L_\theta|, \tag{1.8}$$

which is valid for all $\tau \in (\theta, 1]$, provided we adopt the convention that $D(\cdot, t, \cdot, \cdot) = 1$ for $t \leq 0$. The resulting improvements in the bounds can been seen in Figure 1.2. Note the kink in Figure 1.2 for the r-concave bound (1.8) just before $\tau = 0.6$. This corresponds to the transition from where $D(\theta, \tau, 2B, -1/4)$ is smaller to where $D(\theta^2, 2\tau - 1, B, -1/2)$ is smaller.

We applied the algorithm described in Section 1.5.4 to produce tables of values of

$$\min\{D(\theta^2, 2\tau - 1, 50, -1/2), D(\theta, \tau, 100, -1/4)\}$$

over a grid of θ and τ values; see Table 1.2 and Table 1.3.

1.3.4 How to use these bounds in practice

The quantities $|L_\theta|$ and $\mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor} \cap L_\theta|$, which appear on the right hand sides of the bounds, will in general be unknown to the statistician. Thus when using the bounds, they will typically need to be replaced by p and q respectively. In addition, several parameters must be selected, and in this section we go through each of these in turn and give guidance on how to choose them. Below, we also outline how to use the r-concave bound to obtain an upper bound on the p-value for the null hypothesis $H^k_0 : k \in L_\theta$. These surrogates for p-values can then be used as part of a multiple testing procedure or reported as a measure of significance of each individual variable. Our discussion will focus primarily on the r-concave bound (1.8), though much of what is said will be equally valid for the worst case or unimodal bounds.

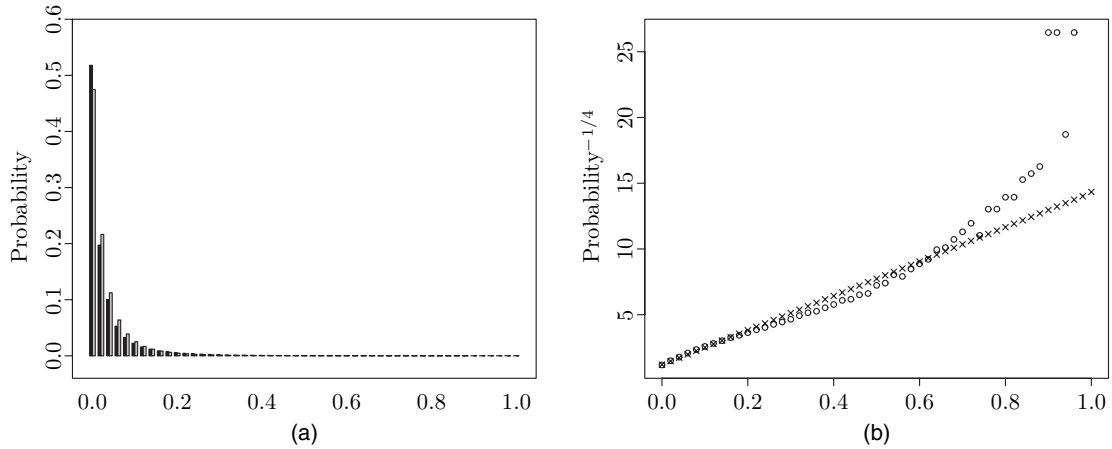


Figure 1.3: A typical example of the probability mass function of $\hat{\Pi}_{25}(k)$ for $k \in L_{q/p}$ (black bars and circles), alongside the $-1/4$-concave distribution (grey bars and crosses), which has maximum tail probability beyond 0.4.

1.3.4.1 Choice of parameters

Choice of B. We recommend $B = 50$ as a default value. Choosing B larger than this increases the computational burden, and may lead to the r-concavity assumptions being violated.

Choice of θ. As mentioned at the beginning of Section 1.3.2, $\theta = q/p$ is a natural choice. In other words, we regard the below average selection probability variables as the irrelevant variables. Other choices of θ are possible, but the use of (1.6) and (1.7) to construct the bound suggests that the inequality will be tightest when most of the variables have a selection probability close to θ.

Choice of q and threshold τ. One can regard the choice of $q = \mathbb{E}(|\hat{S}_{\lfloor n/2 \rfloor}|)$ (which is usually fixed through a tuning parameter λ) as part of the choice of the base selection procedure. One option is to fix q by varying λ at each evaluation of the selection procedure until it selects q variables. However, if the number of variables selected at each iteration is unknown in advance (e.g. if λ is fixed, or if cross-validation is used to choose λ at each iteration), then q can be estimated by $\sum_{k=1}^{p} \hat{\Pi}_B(k)$.

An important point to note is that although choosing λ or q is usually crucial when carrying out variable selection, this is not the case when using CPSS. Our experience is that the performance of CPSS is surprisingly insensitive to the choice of q (see also Meinshausen and Bühlmann (2010)). That is to say, $L_{q/p}$ does not vary much as q varies, and also the final selected sets for different values of q tend to be similar (where different thresholds are chosen to control the selection of variables in $L_{q/p}$ at a pre-specified level). Thus, when using CPSS, it is the threshold τ that plays a role similar to that of a tuning parameter for the base procedure. The great advantage of CPSS is that our bounds allow one to choose τ to control the expected number of low selection probability variables selected.

To summarise: we recommend as a sensible default CPSS procedure taking $B = 50$ and $\theta = q/p$. We then choose τ using the bound (1.8) with $|L_\theta|$ replaced by p to control the expected number of low selection probability variables chosen.

1.3.4.2 Obtaining upper bounds on p-values

Assume that $\tilde{\Pi}_B(k)$ is $-1/2$-concave and $\hat{\Pi}_B(k)$ is $-1/4$-concave for all $k \in L_\theta$, and let H_0^k denote the hypothesis that $k \in L_\theta$ (i.e. that $p_{k,\lfloor n/2 \rfloor} \leq \theta$). To carry out the significance test for H_0^k, we use $\hat{\Pi}_B(k)$ as a test statistic. Writing $\hat{\pi}_B(k)$ for the observed value of the test statistic, we have that

$$\sup_{p_{k,\lfloor n/2 \rfloor} \leq \theta} \mathbb{P}_{p_{k,\lfloor n/2 \rfloor}}(\hat{\Pi}_B(k) \geq \hat{\pi}_B(k)) \leq \min\{D(\theta^2, 2\tau - 1, B, -1/2), D(\theta, \tau, 2B, -1/4)\},$$

and thus the right-hand side is an upper bound on a p-value for H_0^k.

1.4 Numerical properties

1.4.1 Simulation Study

In this section we investigate the performance and validity of the bounds derived in the previous section by applying CPSS to simulated data. We consider both linear and logistic regression and different values of p and n. In each of these settings, we first generate independent explanatory vectors $X_1, \ldots, X_n$ with each $X_i \sim N_p(0, \Sigma)$. We use a Toeplitz covariance matrix Σ with entries

$$\Sigma_{ij} = \rho^{||i-j|-p/2|-p/2},$$

and we look at various values of ρ in $[0, 1)$. So the correlation between the components decays exponentially with the distance between them in $\mathbb{Z}_p$.

For linear regression, we generate a vector of errors $\epsilon \sim N_n(0, \sigma^2 I)$ and set

$$Y = X\beta + \epsilon,$$

where the design matrix X has i^{th} row X_i^T. The error variance σ^2 is chosen to achieve different values of the signal-to-noise ratio (SNR), which we define here by

$$\text{SNR}^2 = \frac{\mathbb{E}\|X\beta\|^2}{\mathbb{E}\|\epsilon\|^2}.$$

For logistic regression, we generate independent responses

$$Y_i \sim \text{Bin}(1, p_i), \quad i = 1, \ldots, n,$$

where

$$\log\left(\frac{p_i}{1 - p_i}\right) = \gamma X_i^T \beta.$$

Here γ is a scaling factor which is chosen to achieve a particular Bayes error rate.

In both cases, we fix the p-dimensional vector of coefficients β to have $s \ll p$ non-zero components, $s/2$ of which we choose as equally spaced points within $[-1, -0.5]$ with the remaining $s/2$ equally spaced in $[0.5, 1]$. The indices of the non-zero components, S, are chosen to follow a geometric progression up to rounding, with first term 1 and $(s+1)^{\text{th}}$ term $p + 1$. The values are then randomly assigned to each index in S, but this choice is then fixed for each particular simulation setting.

With $\rho > 0$, this setup will have several signal variables correlated amongst themselves, and also some signal correlated with noise. In this way, the framework above includes a very wide variety of different data generating processes on which we can test the theory of the previous section.

By varying the base selection procedure, its tuning parameters, the values of ρ, n, p, s and also the SNR and Bayes error rates, we have applied CPSS in several hundred different simulation settings. For reasons of space, we present only a subset of these numerical experiments below, but the results from those omitted are not qualitatively different.

In the graphs which follow, we look at CPSS applied to the Lasso (Tibshirani, 1996), which we implemented using the package `glmnet` (Friedman et al., 2010) in `R` (R Development Core Team, 2005). We follow the original stability selection procedure put forward in Meinshausen and Bühlmann (2010) and compare this to the method suggested by our r-concave bound (1.8). Thus we first choose the level l at which we wish to control the expected number of low selection probability variables (so we aim to have $\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tau} \cap L_{q/p}| \leq l$). Then we fix $q = \sqrt{0.8lp}$ and set the threshold τ at 0.9. This ensures that, according to the original worst case bound, we control the expected number of low selection probability variables selected at the required level. In the r-concave case, we take our threshold as

$$\tilde{\tau} = \min\{\tau \in \{0, 1/2B, \ldots, 1\} : \min\{D(q^2/p^2, 2\tau - 1, B, -1/2), D(q/p, \tau, 2B, -1/4)\} \leq l/p\}.$$

We also give the results one would obtain using the Lasso alone, but with the benefit of an oracle which knows the optimal value of the tuning parameter λ. That is, we take $\hat{S}^{\lambda^*}_n$ as our selected set, where

$$\lambda^* = \inf\{\lambda : \mathbb{E}|\hat{S}^{\lambda}_n \cap L_{q/p}| \leq l\},$$

and $\hat{S}^{\lambda}_n$ is the selected set when using the Lasso with tuning parameter λ applied to the whole dataset.

We present all of our results relative to the performance of CPSS using an oracle-driven threshold τ^*, where τ^* is defined by

$$\tau^* = \min\{\tau \in \{0, 1/2B, \ldots, 1\} : \mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tau} \cap L_{q/p}| \leq l\}.$$

Referring to Figures 1.4-1.7, the heights of the black bars, grey bars and crosses are given by

$$\frac{\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,0.9} \cap S|}{\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tau^*} \cap S|}, \quad \frac{\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tilde{\tau}} \cap S|}{\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tau^*} \cap S|} \quad \text{and} \quad \frac{\mathbb{E}|\hat{S}^{\lambda^*}_{n} \cap S|}{\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tau^*} \cap S|},$$

respectively. Thus the heights of the black and grey bars relate to the loss of power in using the threshold suggested by the corresponding bounds. In all of our simulations, we used $B = 50$. Each scenario was run 500 times, and in order to determine the set $L_{q/p}$, in each scenario, we applied the particular selection procedure $\hat{S}_{\lfloor n/2 \rfloor}$ to 50,000 independent datasets.

It is immediately obvious from the results that using the r-concave bound, we are able to recover significantly more variables in S than when using the the worst case bound. Furthermore, though it is not shown in the graphs explicitly, we also achieve the required level of error control in all but one case (where the r-concavity assumption fails). In fact the one particular example is hardly exceptional in that we have $\mathbb{E}|\hat{S}^{\text{CPSS}}_{n,\tilde{\tau}} \cap L_{q/p}| = 1.034 > 1 = l$. Thus in close accordance with our theory, there are no significant violations of the r-concave bound.

We also see that the loss in power due to using $\tilde{\tau}$ rather than τ^*, is very low. In almost

all of the scenarios, we are able to select more than 75% of the signal we could select with the benefit of an oracle, and usually much more than this. It is interesting that the performance of the oracle CPSS and oracle Lasso procedures are fairly similar. The key advantage of CPSS is that it allows for error control whereas there is in general no way of determining (or even approximating) the optimal λ^* that achieves the required error control. In fact, the performance of CPSS with our bound is only slightly worse then that of the oracle Lasso procedure, and in a few cases, particularly when ρ is small, it is even slightly better. In the cases where $\rho \geq 0.75$, we see that CPSS is not quite as powerful. This is because having such large correlations between variables causes $\{p_{k,\lfloor n/2 \rfloor} : k = 1, \ldots, p\}$ to be relatively spread out in $[0, 1]$. As explained in Section 1.3.4, we expect our bound to weaken in this situation. However, even when the correlation is as high as 0.9, we recover a sizeable proportion of the signal we would select had we used the optimal τ^*.

1.4.2 Real data example

Here we illustrate our CPSS methodology on the widely studied colon dataset of Alon et al. (1999), freely available at `http://microarray.princeton.edu/oncology/affydata/index.html` . The data consist of 2000 gene expression levels from 40 colon tumour samples and 22 normal colon tissue samples, measured using Affymetrix oligonucleotide arrays. Our goal is to identify a small subset of genes which we are confident are linked with the development of colon cancer. Such a task is important for improving scientific understanding of the disease and for selecting genes as potential drug targets.

The data were first preprocessed by averaging over the expression levels for repeated genes (which had been tiled more than once on each array), log-transforming each gene expression level, standardising each row to have mean zero and unit variance, and finally removing the columns corresponding to control genes, so that $p = 1908$ genes remained. The transformation and standardisation are very common preprocessing steps to reduce skewness in the data and help eliminate the effects of systematic variations between different microarrays (see for example Amaratunga and Cabrera (2004) and Dudoit et al. (2002)).

We applied CPSS with ℓ_1 (Lasso) penalised logistic regression as the base procedure, with $B = 50$, and choosing τ both using the r-concave bound of Section 1.3.4, and the original bound of Meinshausen and Bühlmann (2010). We estimated the expected classification error in the two cases by averaging over 128 repetitions of stratified random subsampling validation, taking 8 cancerous and 4 normal observations in each test set. Thus when applying CPSS, we had $n = 40+22-12 = 50$. We looked at $q = 8$, 10 and 12, and set τ to control $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_{q/p}| \leq l$ with $l = 0.1$ and 0.5.

Rather than subsampling completely at random when using CPSS, we also stratified these subsamples to include the same proportion of cancerous to normal samples as in the training data supplied to the procedure. Without this step, some of the subsamples may not include any samples from one of the classes, and applying $\hat{S}_{\lfloor n/2 \rfloor}$ to such a subsample would give misleading results. Using stratified random subsampling is still compatible with our theory, provided that $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta|$ is interpreted as an expectation over random data which contain the same class proportions as observed in the original data. In general, this approach of stratified random subsampling is useful when the response is categorical.

The results in Table 1.1 show that, as expected, the new error bounds allow one to select more variables than the conservative bounds of Meinshausen and Bühlmann (2010) for the same level of error control, and as a consequence, the expected prediction error is reduced. Figure 1.8

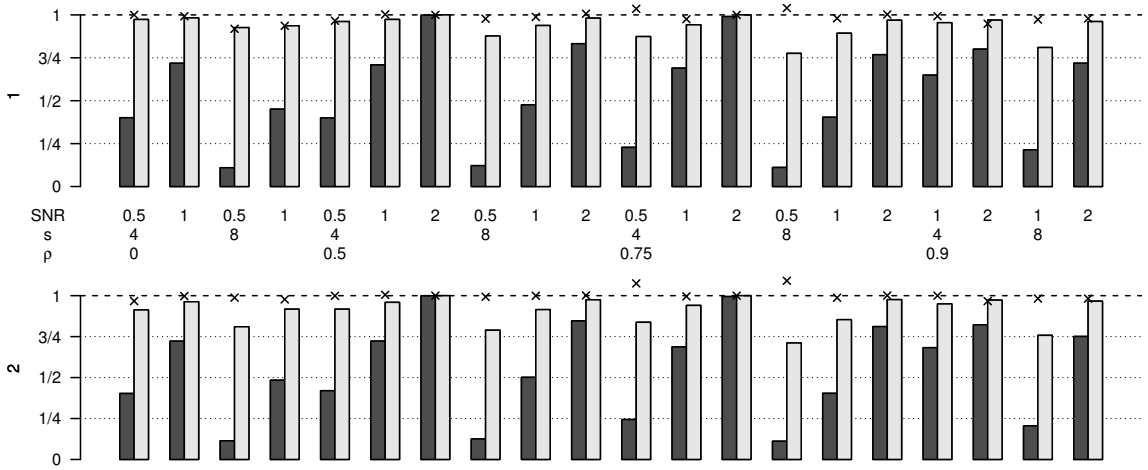


Figure 1.4: Linear regression with $n = 200$, $p = 1000$. The black and grey bars correspond to the worst case and r-concave procedures respectively, with higher bars being preferred. The crosses correspond to a theoretical oracle-driven Lasso procedure (see the beginning of Section 1.4.1 for further details). The y-axis label gives the error control level l.

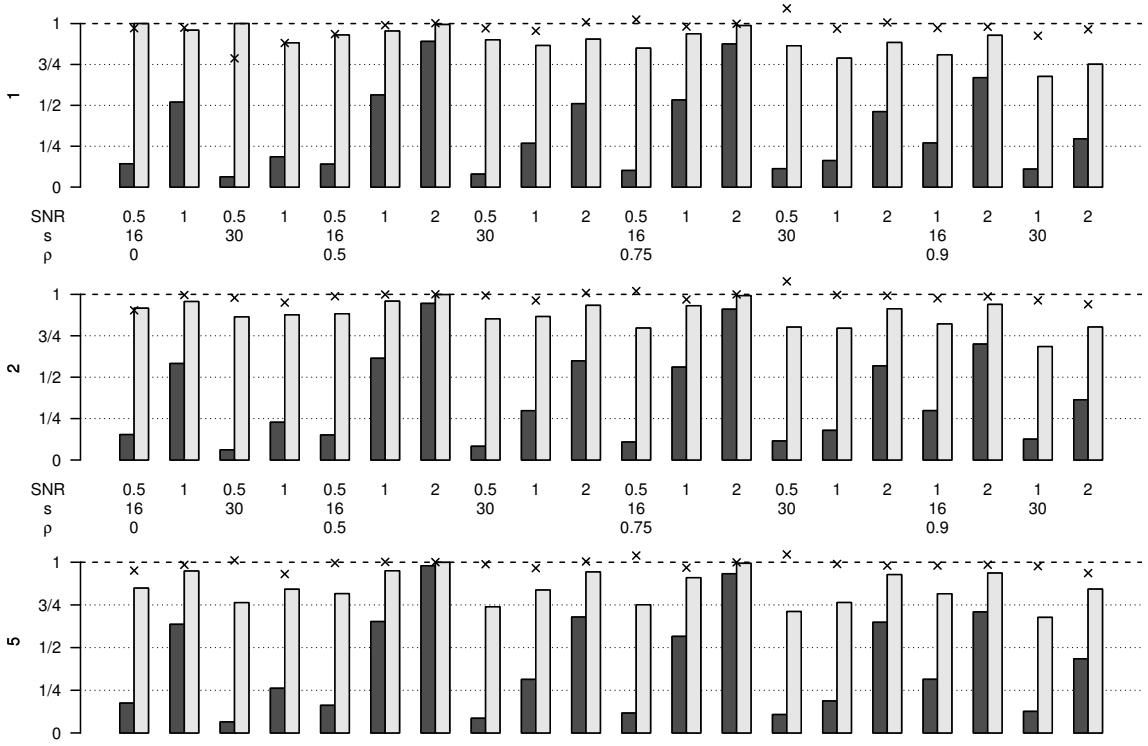


Figure 1.5: As above but $n = 500$, $p = 2000$.

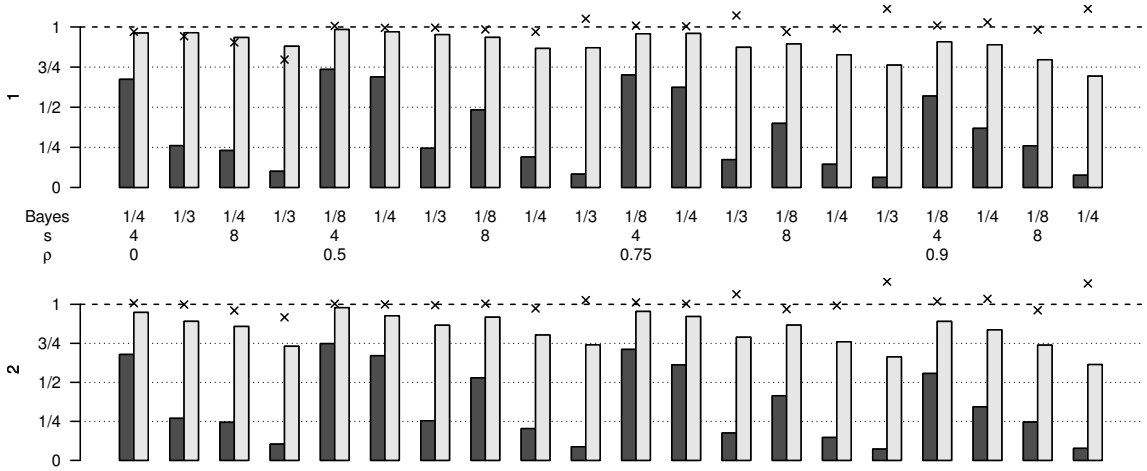


Figure 1.6: As Figure 1.4 but with logistic regression (and $n = 200$, $p = 1000$).

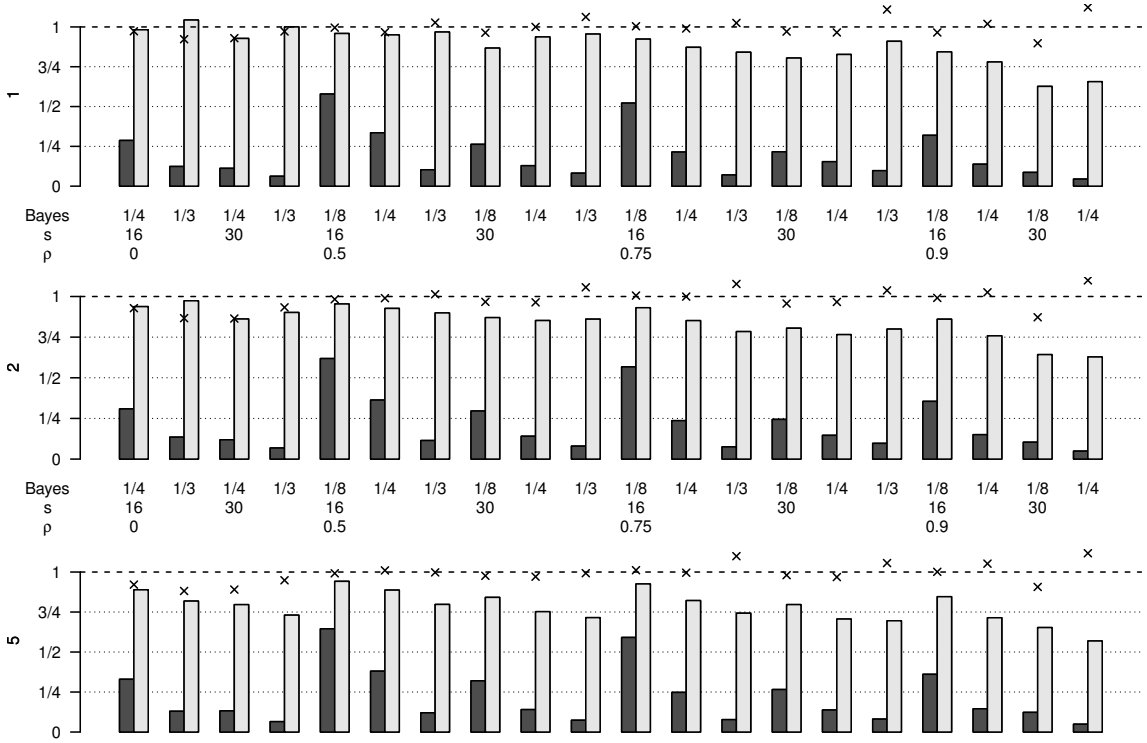


Figure 1.7: As above but with with $n = 500$, $p = 2000$.

Table 1.1: Improvement in classification error (%) over the naive classifier which always determines the data to be from a cancerous tissue. Thus the classification errors are $33\frac{1}{3}\%$ minus these quantities. We also give the average number of variables selected in parentheses.

	Worst case procedure		r-concave procedure	
q	$l = 0.1$	$l = 0.5$	$l = 0.1$	$l = 0.5$
8	4.9 (0.5)	11.6 (1.1)	16 (2.3)	17.5 (5.1)
10	0.9 (0.1)	10.6 (0.9)	14.7 (1.6)	15.8 (4.4)
12	0.0 (0.0)	9.4 (0.8)	12.8 (1.1)	15.8 (4.1)

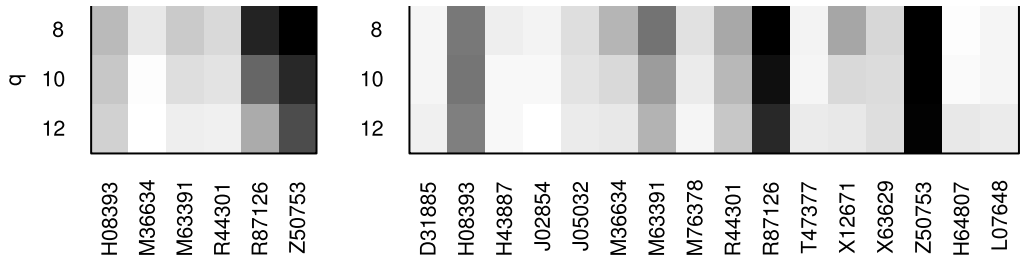


Figure 1.8: For $l = 0.1$ (left) and $l = 0.5$ (right), we have plotted the proportion of times a gene was selected by our r-concave CPSS procedure for all genes which were selected at least 5% of the time among the 128 repetitions. Solid black means the gene was selected in every repetition, and white means it was never selected. Thus dark vertical lines indicate that the choice of q has little effect on the end result of CPSS.

demonstrates the robustness of the selected set to the different values of q. Finally, we also applied CPSS on the entire dataset with $q = 8$ and $B = 50$ and using the r-concave bound of Section 1.3.4 to choose τ to control $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_{q/p}| \leq 0.5$ (cf. Figure 1.9). We see that with just 5 genes out of 1908, we manage to separate the two classes quite well.

1.5 Appendix

1.5.1 Proof of Theorem 1

The proof of Theorem 1 requires the following lemma.

Lemma 5. *(i) If $\tau \in (\frac{1}{2}, 1]$, then*

$$\mathbb{P}(k \in \hat{S}^{\mathrm{CPSS}}_{n,\tau}) \leq \frac{1}{2\tau - 1} p^2_{k,\lfloor n/2 \rfloor}.$$

(ii) If $\tau \in [0, \frac{1}{2})$, then

$$\mathbb{P}(k \notin \hat{S}^{\mathrm{CPSS}}_{n,\tau}) \leq \frac{1}{1 - 2\tau}(1 - p_{k,\lfloor n/2 \rfloor})^2.$$

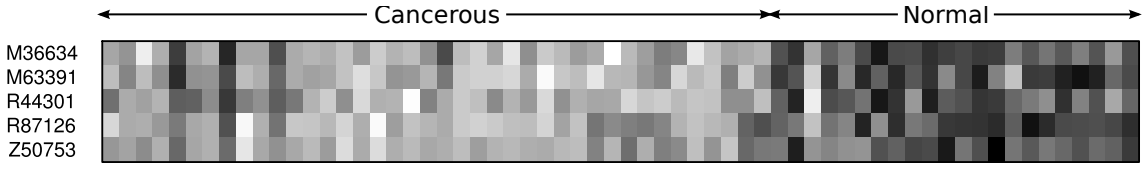


Figure 1.9: A heatmap of the normalised, centered, log intensity values of the genes selected when we use the r-concave bound to choose τ such that we control $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_{q/p}| \leq 0.5$.

Proof. (i) Let $\mathcal{A} = \{(A_{2j-1}, A_{2j}) : j = 1, \ldots, B\}$ be randomly chosen independent pairs of subsets of $\{1, \ldots, n\}$ of size $\lfloor n/2 \rfloor$ such that $A_{2j-1} \cap A_{2j} = \emptyset$. Then

$$0 \leq \frac{1}{B} \sum_{j=1}^{B} \big\{1 - \mathbb{1}_{\{k \in \hat{S}(A_{2j-1})\}}\big\}\big\{1 - \mathbb{1}_{\{k \in \hat{S}(A_{2j})\}}\big\} = 1 - 2\hat{\Pi}_B(k) + \tilde{\Pi}_B(k). \tag{1.9}$$

Now $\mathbb{E}\{\tilde{\Pi}_B(k)\} = \mathbb{E}\{\mathbb{E}(\tilde{\Pi}_B(k)|\mathcal{A})\} = p^2_{k,\lfloor n/2 \rfloor}$ because $\hat{S}(A_{2j-1})$ and $\hat{S}(A_{2j})$ are independent conditional on $\mathcal{A}$. It follows using (1.9) that

$$\begin{aligned}\mathbb{P}(k \in \hat{S}^{\mathrm{CPSS}}_{n,\tau}) = \mathbb{P}\{\hat{\Pi}_B(k) \geq \tau\} \leq \mathbb{P}\big\{\tfrac{1}{2}(1 + \tilde{\Pi}_B(k)) \geq \tau\big\} &= \mathbb{P}\{\tilde{\Pi}_B(k) \geq 2\tau - 1\} \\ &\leq \frac{1}{2\tau - 1} p^2_{k,\lfloor n/2 \rfloor}, \end{aligned} \tag{1.10}$$

where we have used Markov's inequality in the final step.

(ii) Define $\hat{\Pi}^{\hat{N}_n}_B$ and $\tilde{\Pi}^{\hat{N}_n}_B$ by replacing $\hat{S}_n$ with $\hat{N}_n := \{1, \ldots, p\} \setminus \hat{S}_n$ in the definitions of $\hat{\Pi}_B$ and $\tilde{\Pi}_B$ respectively. Then, using the bound corresponding to (1.9) and Markov's inequality again,

$$\begin{aligned}\mathbb{P}(k \notin \hat{S}^{\mathrm{CPSS}}_{n,\tau}) = \mathbb{P}\{\hat{\Pi}_B(k) < \tau\} = \mathbb{P}\{\hat{\Pi}^{\hat{N}_n}_B(k) > 1 - \tau\} &\leq \mathbb{P}\{\tilde{\Pi}^{\hat{N}_n}_B(k) > 1 - 2\tau\} \\ &\leq \frac{1}{1 - 2\tau}(1 - p_{k,\lfloor n/2 \rfloor})^2. \end{aligned}$$

□

Proof of Theorem 1. (i) Note that

$$\mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor} \cap L_\theta| = \mathbb{E}\bigg(\sum_{k=1}^{p} \mathbb{1}_{\{k \in \hat{S}_{\lfloor n/2 \rfloor}\}} \mathbb{1}_{\{p_{k,\lfloor n/2 \rfloor} \leq \theta\}}\bigg) = \sum_{k=1}^{p} p_{k,\lfloor n/2 \rfloor} \mathbb{1}_{\{p_{k,\lfloor n/2 \rfloor} \leq \theta\}}.$$

By Lemma 5, it follows that

$$\begin{aligned}\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta| = \mathbb{E}\bigg(\sum_{k=1}^{p} \mathbb{1}_{\{k \in \hat{S}^{\mathrm{CPSS}}_{n,\tau}\}} \mathbb{1}_{\{p_{k,\lfloor n/2 \rfloor} \leq \theta\}}\bigg) &= \sum_{k=1}^{p} \mathbb{P}(k \in \hat{S}^{\mathrm{CPSS}}_{n,\tau}) \mathbb{1}_{\{p_{k,\lfloor n/2 \rfloor} \leq \theta\}} \\ &\leq \frac{1}{2\tau - 1} \sum_{k=1}^{p} p^2_{k,\lfloor n/2 \rfloor} \mathbb{1}_{\{p_{k,\lfloor n/2 \rfloor} \leq \theta\}} \leq \frac{\theta}{2\tau - 1} \mathbb{E}|\hat{S}_{\lfloor n/2 \rfloor} \cap L_\theta|. \end{aligned}$$

(ii) This proof is very similar to that of (i) and is omitted. □

1.5.2 Proof of Theorem 2.

The proof of Theorem 2 requires several preliminary results, and we use the following notation. Let G denote the finite lattice $\{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\} = \frac{1}{B}\mathbb{Z} \cap [0, 1]$. If f is a probability mass function on G, we write f_i for $f(i/B)$, thereby associating f with $(f_0, f_1, \ldots, f_B) \in \mathbb{R}^{B+1}$.

For $t \in G$, we denote the probability that a random variable distributed according to f takes values greater than or equal to t by $\mathcal{T}_t(f) := \sum_{i \geq Bt} f_i$. We also write $\mathcal{E}(f) := \sum_{i=1}^{B} \frac{i}{B} f_i$ for the expectation of this random variable and $\mathrm{supp}(f) := \{i/B \in G : f_i > 0\}$ for the support of f.

Let $\mathcal{U}$ be the set of all unimodal probability mass functions f on G, and let $\mathcal{U}_\eta := \{f \in U : \mathcal{E}(f) \leq \eta\}$. We consider the problem of maximising $\mathcal{T}_t$ over $f \in \mathcal{U}_\eta$. Since the cases $\eta = 0$ and $t \leq \eta$ are trivial, there is no loss of generality in assuming throughout that $0 < \eta < t$ and $t \in G$,

so in particular $t \geq 1/B$.

Lemma 6. *There exists a maximiser of $\mathcal{T}_t$ in $\mathcal{U}_\eta$.*

Proof. Since $\mathcal{T}_t : \mathbb{R}^{B+1} \to \mathbb{R}$ is linear and therefore continuous, it suffices to show that $\mathcal{U}_\eta \subset \mathbb{R}^{B+1}$ is closed and bounded. Now $\mathcal{U}_\eta$ is bounded as $\mathcal{U}_\eta \subset [0,1]^{B+1}$. Moreover, the hyperplane $H = \{(x_0, \ldots, x_B) : x_0 + x_1 + \ldots + x_B = 1\}$ is closed. Also, $\mathcal{E}$ is a continuous function on $\mathbb{R}^{B+1}$, so $\mathcal{E}^{-1}([0,\eta])$ is closed. Now let $O = \{f \in \mathbb{R}^{B+1} : f \text{ is not unimodal}\}$. If $f \in O$ then there must exist $i_1 < i_2 < i_3$ such that $f_{i_2} < \min\{f_{i_1}, f_{i_3}\}$. Clearly this inequality must hold for all g in a sufficiently small open ball about f, so O is open. We see that

$$\mathcal{U}_\eta = H \cap \mathcal{E}^{-1}([0,\eta]) \cap O^c.$$

Thus $\mathcal{U}_\eta$ is an intersection of closed sets and hence is closed. □

We will make frequent use of the following simple proposition in subsequent proofs.

Proposition 7. *Suppose that $(x_1, \ldots, x_n) \in \mathbb{R}^n$ and $(y_1, \ldots, y_n) \in \mathbb{R}^n$ satisfy*

$$\sum_{i=1}^n x_i = \sum_{i=1}^n y_i,$$

and that there exists some $i^* \in \{1, \ldots, n\}$ with $x_i \geq y_i$ for all $i \leq i^*$ and $x_i \leq y_i$ for all $i > i^*$. Then

$$\sum_{i=1}^n i x_i \leq \sum_{i=1}^n i y_i,$$

with equality if and only if $x_i = y_i$ for $i = 1, \ldots, n$.

Proof. We have

$$\sum_{i \leq i^*} i(x_i - y_i) \leq i^* \sum_{i \leq i^*} (x_i - y_i) = i^* \sum_{i > i^*} (y_i - x_i) \leq \sum_{i > i^*} i(y_i - x_i).$$

□

The following result characterises the extremal elements of $\mathcal{U}_\eta$ in the sense of maximising the tail probability $\mathcal{T}_t$. In particular, it shows that such extremal elements can take only one of two simple forms.

Proposition 8. *Any maximiser $f^* \in \mathcal{U}_\eta$ of $\mathcal{T}_t$ satisfies*

(i) $\mathcal{E}(f^*) = \eta$,

(ii) writing i_M for $B\max(\text{supp}(f^*))$, we have either

(a) $f_0^* > f_1^* = f_2^* = \ldots = f_{i_{M-1}}^* \geq f_{i_M}^*$, *or*

(b) $i_M = t$ *and* $f_0^* = f_1^* = \ldots = f_{i_{M-1}}^* \leq f_{i_M}^*$.

Proof. (i) Suppose $f^* \in \mathcal{U}_\eta$ maximises $\mathcal{T}_t$, but that $\mathcal{E}(f^*) < \eta$. Define $i_m := \min(\text{supp}(f^*))$. As $\eta < \tau$, we must have $i_m < Bt$. Define g by

$$g_i = \begin{cases} 0 & \text{if } i < i_m \\ f_i^* - \epsilon_1 & \text{if } i = i_m \\ f_i^* + \epsilon_2 & \text{if } i > i_m \end{cases}$$

where $\epsilon_1, \epsilon_2 > 0$ are chosen such that $\sum_{i=0}^{B} g_i = 1$, but are small enough that $\mathcal{E}(g) \leq \eta$. Then $g \in \mathcal{U}_\eta$ but $\mathcal{T}_t(g) > \mathcal{T}_t(f^*)$, a contradiction.

(ii) Suppose first that there exists a mode of f^* which is at least t. Let $g \in \mathcal{U}_\eta$ be such that $g_i = f_i^*$ for $i \geq Bt$ and $g_i = \frac{1}{Bt}\sum_{\ell=0}^{Bt-1} f_\ell^*$ for $i < Bt$. As $f_0^* \leq f_1^* \leq \ldots \leq f_{Bt}^*$, we can apply Proposition 7 to see that

$$\mathcal{E}(g) \leq \mathcal{E}(f^*). \tag{1.11}$$

But $\mathcal{T}_t(g) = \mathcal{T}_t(f^*)$, so by optimality of f^* we must have equality in (1.11). Thus Proposition 7 gives us that $f^* = g$.

Next, define $h \in \mathcal{U}_\eta$ by $h_i = f_i^*$ for $i < Bt$, $h_{Bt} = \mathcal{T}_t(f^*)$, and $h_i = 0$ for $i > Bt$. Then $\mathcal{T}_t(h) = \mathcal{T}_t(f^*)$. Again Proposition 7 and the optimality of f^* give that $f^* = h$. Thus f^* satisfies property (ii)(b) of the theorem.

Now suppose that there is no mode of f^* which is at least t, so $f_{Bt}^* \geq f_{Bt+1}^* \geq \ldots \geq f_B^*$. Let $g \in \mathcal{U}_\eta$ satisfy $g_i = f_i^*$ for $i \geq Bt$ and $g_1 = \ldots = g_{Bt}$. We must have $g_0 > g_1$, otherwise f^* would have a mode at t. As $\mathcal{T}_t(g) = \mathcal{T}_t(f^*)$, optimality of f^* and Proposition 7 imply $f^* = g$.

Finally, let $h \in \mathcal{U}_\eta$ satisfy $h_i = f_i^*$ for $i \leq Bt$ and $h_{Bt} = h_{Bt+1} = \ldots = h_{k-1} \geq h_k$, where k and h_k are chosen such that $\sum_{i=0}^{B} h_i = 1$. As before, Proposition 7 allows us to deduce that $f^* = h$. Thus f^* satisfies property (ii)(a) of the theorem. □

We are now in a position to state Markov's inequality for random variables with unimodal distributions on G, which may be of some independent interest.

Theorem 9 (Markov's inequality under unimodality)**.** *Let X be a random variable with a unimodal distribution on $G = \{0, \frac{1}{B}, \frac{2}{B}, \ldots, 1\}$, and let $t \in G$. If $\eta := \mathbb{E}(X) \leq 1/3$, then*

$$\mathbb{P}(X \geq t) \leq \begin{cases} \dfrac{2\eta - t + \frac{1}{B}}{t + \frac{1}{B}} & \text{if} \quad t \in \left(\eta,\, \min\left(\frac{3}{2}\eta + \frac{1}{2B}, 2\eta\right)\right] \\ \dfrac{\eta}{2t - \frac{1}{B}} & \text{if} \quad t \in \left(\min\left(\frac{3}{2}\eta + \frac{1}{2B}, 2\eta\right),\, \frac{1}{2}\right] \\ \dfrac{2\eta(1 - t + \frac{1}{B})}{1 + \frac{1}{B}} & \text{if} \quad t \in \left(\frac{1}{2},\, 1\right]. \end{cases}$$

Let d be defined by

$$d := d(\eta, B) = -2\left(\eta - \tfrac{1}{2}\right)(6\eta + 1) + \frac{2 - 4\eta}{B} + \frac{(4\eta - 1)^2}{B^2}.$$

If $\eta > 1/3$ and $d > 0$, then

$$\mathbb{P}(X \geq t) \leq \begin{cases} \dfrac{2\eta - t + \frac{1}{B}}{t + \frac{1}{B}} & \text{if} \quad t \in \left(\eta,\, \frac{1}{2} + \frac{1}{4\eta}(1 + \frac{1}{B} - d^{1/2})\right] \\ \dfrac{2\eta(1 - t + \frac{1}{B})}{1 + \frac{1}{B}} & \text{if} \quad t \in \left(\frac{1}{2} + \frac{1}{4\eta}(1 + \frac{1}{B} - d^{1/2}),\, 1\right]. \end{cases}$$

Finally, if $\eta > 1/3$ and $d \leq 0$, then

$$\mathbb{P}(X \geq t) \leq \frac{2\eta - t + \frac{1}{B}}{t + \frac{1}{B}}.$$

Proof. Proposition 8 tells us that $\mathbb{P}(X \geq t)$ must be at most the maximum of the optimal solutions

to the following two optimisation problems:

$$\begin{array}{lll} (P): & \text{Maximise} & b(s-Bt)+c \text{ in } a,b,c,s \\ & \text{subject to} & a+(s-1)b+c=1 \\ & & \frac{s}{2}(s-1)b+sc=B\eta \\ & & a>b\geq c\geq 0 \\ & & s\in\{Bt,Bt+1,\ldots,B\} \end{array} \qquad \begin{array}{lll} (Q): & \text{Maximise} & b \text{ in } a,b \\ & \text{subject to} & Bta+b=1 \\ & & \frac{Bt}{2}(Bt-1)a+Btb=B\eta \\ & & b\geq a\geq 0. \end{array}$$

Problem (P) corresponds to case (ii)(a) of Proposition 8, and problem (Q) to case (ii)(b).

The solution to (Q) is determined entirely by the constraints, and we see that the optimal value is

$$\frac{2\eta-t+\frac{1}{B}}{t+\frac{1}{B}}. \tag{1.12}$$

To solve (P), we break it into $B(1-t)+1$ subproblems: for $s\in\{Bt,Bt+1,\ldots,B\}$, we define subproblem $(P(s))$ as follows:

$$\begin{array}{lll} (P(s)): & \text{Maximise} & b(s-Bt)+c \text{ in } a,b,c \\ & \text{subject to} & a+(s-1)b+c=1 \\ & & \frac{s}{2}(s-1)b+sc=B\eta \\ & & b\geq c, \\ & & a,b,c\geq 0. \end{array}$$

Notice that we have not included the $a>b$ constraint. This is because Proposition 8 ensures that this constraint is always satisfied at an optimal solution of (P), so there exists s^* such that every optimal solution of $(P(s^*))$ corresponds to an optimal solution of (P).

Now each subproblem is a standard linear programming problem, so we know that one of the basic feasible solutions must be optimal. Since $a>0$, all basic feasible solutions must have either $c=0$ or $b=c$. Thus we may replace the subproblems $(P(s))$ by

$$\begin{array}{lll} (P'(s)): & \text{Maximise} & b(s-Bt+1) \text{ in } a,b \\ & \text{subject to} & a+sb=1 \\ & & \frac{s}{2}(s+1)b=B\eta \\ & & a,b\geq 0. \end{array}$$

The second constraint is enough to determine that the optimal value of $P'(s)$ is

$$\frac{2B\eta(s-Bt+1)}{s(s+1)} =: \gamma(s). \tag{1.13}$$

Now we can proceed to find an s^* which maximises γ over $\{Bt,Bt+1,\ldots,B\}$. The sign of $\gamma'(s)$ is the sign of

$$-s^2+2(Bt-1)s+Bt-1.$$

This quadratic in s has roots

$$Bt-1\pm\sqrt{(Bt-1)^2+Bt-1}.$$

So $\gamma(s)$ is increasing for all $s \in \{Bt, Bt+1, \ldots, B\}$ with

$$s \leq Bt - 1 + \sqrt{\left(Bt - \tfrac{1}{2}\right)^2 - \tfrac{1}{4}} =: s_0. \tag{1.14}$$

When $s_0 < B$, we must have $s^* \in \{2Bt-2, 2Bt-1\}$. In fact, by examining (1.13), we see that $\gamma(2Bt-2) = \gamma(2Bt-1)$. Also, from (1.14), we see that when $t > 1/2$, we have that $s_0 \geq B$, so $s^* = B$. So far, we have shown that

$$\mathbb{P}(X \geq t) \leq \max(b_1, b_2, b_3),$$

where bounds b_1, b_2 and b_3 are given by

$$\begin{aligned}
b_1 := b_1(t,\eta,B) &= \frac{2\eta - t + \frac{1}{B}}{t + \frac{1}{B}} \mathbb{1}_{\{\eta < t \leq \min(2\eta, 1)\}} \\
b_2 := b_2(t,\eta,B) &= \frac{\eta}{2t - \frac{1}{B}} \mathbb{1}_{\{\eta < t \leq 1/2\}} \\
b_3 := b_3(t,\eta,B) &= \frac{2\eta(1 - t + \frac{1}{B})}{1 + \frac{1}{B}} \mathbb{1}_{\{\max(\eta, 1/2) \leq t \leq 1\}}.
\end{aligned}$$

All that remains now is to determine which of b_1, b_2 and b_3 have the largest value. We first consider the case when $\eta \leq \frac{1}{3}$. When $t \leq \min(1/2, 2\eta)$,

$$\operatorname{sgn}(b_2 - b_1) = \operatorname{sgn}\left\{\left(t - \tfrac{3}{2}\eta - \tfrac{1}{2B}\right)\left(t - \tfrac{1}{B}\right)\right\}.$$

Now for $1/2 < t \leq 2\eta$,

$$\frac{\partial b_3}{\partial t} = -\frac{2\eta}{1 + \frac{1}{B}} \geq -\frac{(2\eta + \frac{2}{B})}{(t + \frac{1}{B})^2} = \frac{\partial b_1}{\partial t}.$$

Furthermore, .

$$b_3\left(\tfrac{1}{2} + \tfrac{1}{2B}, \eta, B\right) = \eta \geq \frac{2\eta - \frac{1}{2} + \frac{1}{2B}}{\frac{1}{2} + \frac{3}{2B}} = b_1\left(\tfrac{1}{2} + \tfrac{1}{2B}, \eta, B\right).$$

Putting this together gives the required bound for $\eta \leq 1/3$.

When $\eta > 1/3$, we can ignore b_2 as it is dominated by b_1. Comparing b_1 and b_3, we get the final cases of the bound. □

Proof of Theorem 2. Recalling that $\mathbb{E}\{\tilde{\Pi}_B(k)\} = p^2_{k,\lfloor n/2 \rfloor}$, we follow the proof of Lemma 5, but apply Theorem 9 at the last step of (1.10) with $t = 2\tau - 1$ to deduce that if the distribution of $\tilde{\Pi}_B(k)$ is unimodal, then

$$\mathbb{P}(k \in \hat{S}^{\mathrm{CPSS}}_{n,\tau}) \leq \mathbb{P}\{\tilde{\Pi}_B(k) \geq 2\tau - 1\} \leq C(\tau, B) p^2_{k,\lfloor n/2 \rfloor},$$

where $C(\tau, B)$ is given in the statement of Theorem 2. The bound for $\mathbb{E}|\hat{S}^{\mathrm{CPSS}}_{n,\tau} \cap L_\theta|$ then follows in the same way that Theorem 1 follows from Lemma 5. □

1.5.3 Proofs of results on r-concavity

Proof of Proposition 3. Suppose that f is log-concave, so we may write $f = e^{-\phi}$ where ϕ is a convex function. If $r < 0$, then $-r\phi$ is convex, and as the exponential function is increasing and convex, $f^r = e^{-r\phi}$ is convex.

Conversely, suppose that f is not log-concave, so there exist x, y and $\lambda \in (0,1)$ with $f(\lambda x + (1-\lambda)y) < f(x)^\lambda f(y)^{1-\lambda}$. Then as $M_r(f(x), f(y); \lambda) \to f(x)^\lambda f(y)^{1-\lambda}$ as $r \to 0$, we must have $f(\lambda x + (1-\lambda)y) < M_r(f(x), f(y); \lambda)$ for some $r < 0$, and so f cannot be r-concave. □

Proof of Proposition 4. Let $I = \{1, \ldots, l\} \cup \{u, \ldots, B-1\}$. The conditions on f imply that

$$f_i > \min\{f_{i-1}, f_{i+1}\}, \quad i \in I.$$

Then as $M_r(f_{i-1}, f_{i+1}, \frac{1}{2}) \to \min\{f_{i-1}, f_{i+1}\}$ as $r \to -\infty$, for each $i \in I$, may choose an $r_i < 0$ with

$$f_i > M_{r_i}(f_{i-1}, f_{i+1}; \tfrac{1}{2}). \tag{1.15}$$

Set $r = \min_{i \in I} r_i$. Observe that as $M_r(a, b; \frac{1}{2})$ is increasing in r for all fixed a and b, the inequalities (1.15) are all satisfied when $r_i = r$. Thus $f_i^r \leq \frac{1}{2}(f_{i-1}^r + f_{i+1}^r)$ for all $i \in \{1, \ldots, B-1\}$, so f is r-concave. □

By analogy with the unimodal case, let $\mathcal{F}_{r,\eta} := \{f \in \mathcal{F}_r : \mathcal{E}(f) \leq \eta\}$. In maximising $\mathcal{T}_t$ over $\mathcal{F}_{r,\eta}$, there is again no loss of generality in assuming $0 < \eta < t$.

Lemma 10. *For each $r < 0$, there exists a maximiser of $\mathcal{T}_t$ in $\mathcal{F}_{r,\eta}$.*

Proof. This proof is almost identical to that of Lemma 6, except here we let $O = \{f \in \mathbb{R}^{B+1} : f^r \text{ is not convex}\}$. If $f \in O$, then there must exist $i_1 < i_2 < i_3$ such that

$$(i_3 - i_2) f_{i_1}^r + (i_2 - i_1) f_{i_3}^r < (i_3 - i_1) f_{i_2}^r$$

and it is clear that the above inequality must hold for all g in a sufficiently small open ball about f. Thus O is open, and the rest of the proof is clear. □

Proposition 11. *Any maximiser $f^* \in \mathcal{F}_{r,\eta}$ of $\mathcal{T}_t$ satisfies*

(i) $\mathcal{E}(f^*) = \eta$

(ii) f^{*r} *is linear between* f_0^{*r} *and* $f_{i_{M-1}}^{*r}$, *where* $i_M = B \max(\mathrm{supp}(f^*))$.

Proof. (i) Suppose that $\mathcal{E}(f^*) < \eta$. Define $i_m := B \min(\mathrm{supp}(f^*))$. Let $\phi = f^{*r}$ and define a new sequence $\psi := (\psi_i : i = 0, \ldots, B)$ by

$$\psi_i = \begin{cases} \infty & \text{if } i < i_m \\ \phi_i + \epsilon_1 & \text{if } i = i_m \\ \phi_i - \epsilon_2 & \text{if } i > i_m \end{cases}$$

where $\epsilon_1, \epsilon_2 > 0$ are chosen such that $\sum_{i=0}^B \psi_i^{1/r} = 1$, but are small enough that $\mathcal{E}(\psi^{1/r}) \leq \eta$. Then ψ is convex, so $\psi^{1/r} \in \mathcal{F}_{r,\eta}$. Since $\eta > 0$, we must have $\mathcal{T}_t(f^*) > 0$ so $\max(\mathrm{supp}(f^*)) \geq t$. Also, as we are assuming $\eta < \tau$, we must have $i_m < t$. Therefore $\mathcal{T}_t(\psi^{1/r}) > \mathcal{T}_t(f^*)$, which is a contradiction.

(ii) Set $\phi = f^{*r}$, so ϕ is convex and $\phi^{1/r} = f^*$. Define $\psi' = (\psi'_0, \ldots, \psi'_B) \in \mathbb{R}^{B+1}$ as follows. Take $\psi'_i = \phi_i$ for $i \geq Bt$, but make ψ' linear between ψ'_0 and ψ'_{Bt} such that $g := \psi'^{1/r}$ has $\sum_{i=0}^B g_i = 1$ and $g_0 > 0$. This is possible since $\mathcal{E}(f^*) \leq \eta < t$, so $\min(\mathrm{supp}(f^*)) < t$. Note that ψ' is still convex since we must have $\psi'_{Bt} - \psi'_{Bt-1} \leq \phi_{Bt} - \phi_{Bt-1}$. Also $\mathcal{T}_t(g) = \mathcal{T}_t(f^*)$. Applying

Proposition 7, we see that $\mathcal{E}(g) \leq \mathcal{E}(f^*)$. Optimality of f^* means that equality must hold, so $f^* = g$ and also $\phi = \psi'$.

Now if ϕ is in fact linear between ϕ_0 and ϕ_B, condition (ii) of the theorem is satisfied and we are done. Otherwise we may assume ϕ is not a linear function between ϕ_{Bt-1} and ϕ_B and we can define ψ such that $\psi_i = \phi_i$ for $i \leq Bt$, that ψ is linear between ψ_{Bt-1} and ψ_{k-1} and $\psi_i = \infty$ for $i > k$. Here, k is chosen such that $g := \psi^{1/r}$ has $\sum_{i=0}^{B} g_i = 1$, and the convexity of ϕ ensures that such a $k \leq B$ exists. Applying Proposition 7, we see that $\mathcal{E}(g) \leq \mathcal{E}(f^*)$. Since $\mathcal{T}_t(g) = \mathcal{T}_t(f^*)$, as before, optimality of f^* allows us to conclude that $f^* = g$. □

1.5.4 Computing the r-concave tail probability bound

Here we describe a numerical algorithm that computes the function D defined in Section 1.3.3. Note that this is the maximum of $\mathcal{T}_t(f)$ over $f \in \mathcal{F}_{r,\eta}$. We shall only discuss the case where f^* is decreasing, as is always the case when $t > 2\eta$. The increasing case is very similar and less important for our application. We first note that we may parametrise the r-concave probability mass functions whose r^{th} powers are linear as follows:

$$f_{a,k;i} = \frac{(a+i)^{1/r}}{\sum_{j=0}^{k}(a+j)^{1/r}}, \quad i = 0, 1, \ldots, k \tag{1.16}$$

where $k \leq B$. As $\mathcal{E}(f_{a,k})$ is strictly increasing in a, for each k, there is a unique a_k for which $\mathcal{E}(f_{a_k,k}) = \eta$. We also note here that a_k decreases with k. This is easily seen by observing that, regardless of the value of k, the parameter a in (1.16) determines the ratio of $f_{a,k;i}$ to $f_{a,k;j}$, each i, j.

According to Proposition 11, if $f^* \in \mathcal{F}_{r,\eta}$ maximises $\mathcal{T}_t$, then f^{*r} is linear up to its penultimate support point. We can parametrise these in the following way. Write

$$\frac{\sum_{i=1}^{k} i(a+i)^{1/r} + (k+1)c}{\sum_{j=0}^{k}(a+j)^{1/r} + c} = B\eta,$$

and then solve for c:

$$c = c(a,k) = \frac{B\eta \sum_{j=0}^{k}(a+j)^{1/r} - \sum_{i=1}^{k} i(a+i)^{1/r}}{k+1-B\eta}.$$

We see that as a ranges through $[a_{k+1}, a_k]$, we obtain all the relevant probability mass functions supported on $0, 1, \ldots, k+1$ via

$$g_{a,k;i} = \frac{(a+i)^{1/r}}{\sum_{j=0}^{k}(a+j)^{1/r} + c(a,k)}, \quad i = 0, 1, \ldots, k$$
$$g_{a,k;k+1} = \frac{c(a,k)}{\sum_{j=0}^{k}(a+j)^{1/r} + c(a,k)}.$$

The tail probability of $g_{a,k}$, when the threshold is t, is

$$\mathcal{T}_t(g_{a,k}) = 1 - \frac{(k+1-B\eta)\sum_{i=0}^{Bt-1}(a+i)^{1/r}}{\sum_{i=0}^{k}(k+1-i)(a+i)^{1/r}} \tag{1.17}$$

and we may maximise this over $a \in [a_{k+1}, a_k]$ to obtain an optimal a_k^* for each k. This is easily

accomplished using a general purpose optimiser such as `optimize` in `R`. To summarise, we have the following simple procedure for computing $\mathcal{T}_t(f^*)$.

1. For each $k \in \{t, \ldots, B\}$, determine (numerically), the solution in a_k to $\mathcal{E}(f_{a,k}) = \eta$.
2. Find $a_k^* := \text{argmax}_{a \in [a_{k+1}, a_k]} \mathcal{T}_t(g_{a,k})$, for each k.
3. Let $k^*(t) := \text{argmax}_k \mathcal{T}_t(g_{a_k^*, k})$.

Then $\mathcal{T}_t(f^*) = \mathcal{T}_t(g_{a^*_{k^*(t)}, k^*(t)})$. When we wish to evaluate $\mathcal{T}_t(f^*)$ for a range of values of t, the process is simplified by the observation that $k^*(t)$ is increasing in t, and thus in Step 2 we need only consider those k which are at least $k^*(t - 1/B)$.

Using the algorithm described above, we have computed

$$\min\{D(\theta^2, 2\tau - 1, 50, -1/2), D(\theta, \tau, 100, -1/4)\}$$

over a grid of θ and τ values (cf. Tables 1.2 and 1.3). An `R` implementation of the algorithm is available from `http://www.statslab.cam.ac.uk/~rds37/papers/r_concave_tail.R`.

Table 1.2: Table of values of $\min\{D(\theta^2, 2\tau - 1, 50, -1/2), D(\theta, \tau, 100, -1/4)\}$ for $\theta \in \{0.01, 0.02, 0.03, 0.04, 0.05\}$.

	θ				
τ	0.01	0.02	0.03	0.04	0.05
0.30	6.11×10^{-4}	2.70×10^{-3}	6.51×10^{-3}	1.21×10^{-2}	1.93×10^{-2}
0.31	5.57×10^{-4}	2.47×10^{-3}	5.99×10^{-3}	1.12×10^{-2}	1.79×10^{-2}
0.32	5.08×10^{-4}	2.26×10^{-3}	5.52×10^{-3}	1.03×10^{-2}	1.66×10^{-2}
0.33	4.65×10^{-4}	2.08×10^{-3}	5.10×10^{-3}	9.57×10^{-3}	1.55×10^{-2}
0.34	4.27×10^{-4}	1.92×10^{-3}	4.71×10^{-3}	8.88×10^{-3}	1.44×10^{-2}
0.35	3.92×10^{-4}	1.77×10^{-3}	4.36×10^{-3}	8.25×10^{-3}	1.34×10^{-2}
0.36	3.61×10^{-4}	1.64×10^{-3}	4.05×10^{-3}	7.68×10^{-3}	1.25×10^{-2}
0.37	3.33×10^{-4}	1.51×10^{-3}	3.76×10^{-3}	7.15×10^{-3}	1.17×10^{-2}
0.38	3.08×10^{-4}	1.40×10^{-3}	3.50×10^{-3}	6.67×10^{-3}	1.09×10^{-2}
0.39	2.85×10^{-4}	1.30×10^{-3}	3.26×10^{-3}	6.23×10^{-3}	1.02×10^{-2}
0.40	2.64×10^{-4}	1.21×10^{-3}	3.04×10^{-3}	5.82×10^{-3}	9.59×10^{-3}
0.41	2.45×10^{-4}	1.13×10^{-3}	2.83×10^{-3}	5.45×10^{-3}	9.00×10^{-3}
0.42	2.27×10^{-4}	1.05×10^{-3}	2.65×10^{-3}	5.10×10^{-3}	8.44×10^{-3}
0.43	2.12×10^{-4}	9.81×10^{-4}	2.48×10^{-3}	4.78×10^{-3}	7.93×10^{-3}
0.44	1.97×10^{-4}	9.16×10^{-4}	2.32×10^{-3}	4.48×10^{-3}	7.45×10^{-3}
0.45	1.84×10^{-4}	8.56×10^{-4}	2.17×10^{-3}	4.21×10^{-3}	7.01×10^{-3}
0.46	1.71×10^{-4}	8.01×10^{-4}	2.03×10^{-3}	3.95×10^{-3}	6.60×10^{-3}
0.47	1.60×10^{-4}	7.50×10^{-4}	1.91×10^{-3}	3.72×10^{-3}	6.21×10^{-3}
0.48	1.50×10^{-4}	7.02×10^{-4}	1.79×10^{-3}	3.50×10^{-3}	5.85×10^{-3}
0.49	1.40×10^{-4}	6.58×10^{-4}	1.68×10^{-3}	3.29×10^{-3}	5.52×10^{-3}
0.50	1.31×10^{-4}	6.18×10^{-4}	1.58×10^{-3}	3.10×10^{-3}	5.20×10^{-3}
0.51	1.23×10^{-4}	5.80×10^{-4}	1.49×10^{-3}	2.92×10^{-3}	4.91×10^{-3}
0.52	1.15×10^{-4}	5.45×10^{-4}	1.40×10^{-3}	2.75×10^{-3}	4.63×10^{-3}
0.53	1.08×10^{-4}	5.12×10^{-4}	1.32×10^{-3}	2.59×10^{-3}	4.37×10^{-3}
0.54	1.01×10^{-4}	4.81×10^{-4}	1.24×10^{-3}	2.44×10^{-3}	4.13×10^{-3}
0.55	9.51×10^{-5}	4.52×10^{-4}	1.17×10^{-3}	2.30×10^{-3}	3.90×10^{-3}
0.56	8.93×10^{-5}	4.26×10^{-4}	1.10×10^{-3}	2.17×10^{-3}	3.68×10^{-3}
0.57	8.39×10^{-5}	4.01×10^{-4}	1.04×10^{-3}	2.05×10^{-3}	3.48×10^{-3}
0.58	7.89×10^{-5}	3.77×10^{-4}	9.78×10^{-4}	1.94×10^{-3}	3.29×10^{-3}
0.59	7.41×10^{-5}	3.55×10^{-4}	9.22×10^{-4}	1.83×10^{-3}	2.99×10^{-3}
0.60	6.97×10^{-5}	3.34×10^{-4}	8.69×10^{-4}	1.64×10^{-3}	2.61×10^{-3}
0.61	6.56×10^{-5}	3.15×10^{-4}	7.99×10^{-4}	1.45×10^{-3}	2.30×10^{-3}
0.62	6.16×10^{-5}	2.96×10^{-4}	7.12×10^{-4}	1.29×10^{-3}	2.05×10^{-3}
0.63	5.80×10^{-5}	2.78×10^{-4}	6.38×10^{-4}	1.16×10^{-3}	1.84×10^{-3}
0.64	5.45×10^{-5}	2.51×10^{-4}	5.76×10^{-4}	1.04×10^{-3}	1.66×10^{-3}
0.65	5.13×10^{-5}	2.27×10^{-4}	5.22×10^{-4}	9.46×10^{-4}	1.51×10^{-3}
0.66	4.82×10^{-5}	2.07×10^{-4}	4.75×10^{-4}	8.61×10^{-4}	1.37×10^{-3}
0.67	4.53×10^{-5}	1.89×10^{-4}	4.33×10^{-4}	7.86×10^{-4}	1.25×10^{-3}
0.68	4.23×10^{-5}	1.73×10^{-4}	3.97×10^{-4}	7.20×10^{-4}	1.15×10^{-3}
0.69	3.88×10^{-5}	1.58×10^{-4}	3.64×10^{-4}	6.60×10^{-4}	1.05×10^{-3}
0.70	3.56×10^{-5}	1.45×10^{-4}	3.35×10^{-4}	6.07×10^{-4}	9.68×10^{-4}
0.71	3.28×10^{-5}	1.34×10^{-4}	3.08×10^{-4}	5.59×10^{-4}	8.91×10^{-4}
0.72	3.02×10^{-5}	1.23×10^{-4}	2.84×10^{-4}	5.15×10^{-4}	8.21×10^{-4}
0.73	2.79×10^{-5}	1.14×10^{-4}	2.62×10^{-4}	4.76×10^{-4}	7.58×10^{-4}
0.74	2.57×10^{-5}	1.05×10^{-4}	2.42×10^{-4}	4.39×10^{-4}	7.00×10^{-4}
0.75	2.37×10^{-5}	9.70×10^{-5}	2.23×10^{-4}	4.06×10^{-4}	6.47×10^{-4}
0.76	2.19×10^{-5}	8.95×10^{-5}	2.06×10^{-4}	3.75×10^{-4}	5.97×10^{-4}
0.77	2.02×10^{-5}	8.27×10^{-5}	1.90×10^{-4}	3.46×10^{-4}	5.52×10^{-4}
0.78	1.87×10^{-5}	7.63×10^{-5}	1.76×10^{-4}	3.20×10^{-4}	5.10×10^{-4}
0.79	1.72×10^{-5}	7.04×10^{-5}	1.62×10^{-4}	2.95×10^{-4}	4.70×10^{-4}
0.80	1.59×10^{-5}	6.48×10^{-5}	1.50×10^{-4}	2.72×10^{-4}	4.34×10^{-4}
0.81	1.46×10^{-5}	5.97×10^{-5}	1.38×10^{-4}	2.51×10^{-4}	3.99×10^{-4}
0.82	1.34×10^{-5}	5.48×10^{-5}	1.27×10^{-4}	2.30×10^{-4}	3.67×10^{-4}
0.83	1.23×10^{-5}	5.03×10^{-5}	1.16×10^{-4}	2.12×10^{-4}	3.37×10^{-4}
0.84	1.13×10^{-5}	4.60×10^{-5}	1.06×10^{-4}	1.94×10^{-4}	3.09×10^{-4}
0.85	1.03×10^{-5}	4.20×10^{-5}	9.71×10^{-5}	1.77×10^{-4}	2.82×10^{-4}
0.86	9.35×10^{-6}	3.82×10^{-5}	8.84×10^{-5}	1.61×10^{-4}	2.57×10^{-4}
0.87	8.47×10^{-6}	3.46×10^{-5}	8.02×10^{-5}	1.46×10^{-4}	2.33×10^{-4}
0.88	7.64×10^{-6}	3.12×10^{-5}	7.24×10^{-5}	1.32×10^{-4}	2.11×10^{-4}
0.89	6.85×10^{-6}	2.80×10^{-5}	6.50×10^{-5}	1.19×10^{-4}	1.89×10^{-4}
0.90	6.10×10^{-6}	2.49×10^{-5}	5.80×10^{-5}	1.06×10^{-4}	1.69×10^{-4}

Table 1.3: Table of values of $\min\{D(\theta^2, 2\tau - 1, 50, -1/2), D(\theta, \tau, 100, -1/4)\}$ for $\theta \in \{0.06, 0.07, 0.08, 0.09, 0.1\}$.

	θ				
τ	0.06	0.07	0.08	0.09	0.10
0.30	2.81×10^{-2}	3.82×10^{-2}	4.97×10^{-2}	6.24×10^{-2}	7.63×10^{-2}
0.31	2.61×10^{-2}	3.57×10^{-2}	4.64×10^{-2}	5.84×10^{-2}	7.14×10^{-2}
0.32	2.43×10^{-2}	3.33×10^{-2}	4.35×10^{-2}	5.47×10^{-2}	6.70×10^{-2}
0.33	2.27×10^{-2}	3.12×10^{-2}	4.08×10^{-2}	5.14×10^{-2}	6.30×10^{-2}
0.34	2.12×10^{-2}	2.92×10^{-2}	3.83×10^{-2}	4.83×10^{-2}	5.93×10^{-2}
0.35	1.98×10^{-2}	2.73×10^{-2}	3.59×10^{-2}	4.55×10^{-2}	5.59×10^{-2}
0.36	1.85×10^{-2}	2.57×10^{-2}	3.38×10^{-2}	4.29×10^{-2}	5.28×10^{-2}
0.37	1.74×10^{-2}	2.41×10^{-2}	3.18×10^{-2}	4.04×10^{-2}	4.99×10^{-2}
0.38	1.63×10^{-2}	2.26×10^{-2}	2.99×10^{-2}	3.81×10^{-2}	4.72×10^{-2}
0.39	1.53×10^{-2}	2.13×10^{-2}	2.82×10^{-2}	3.60×10^{-2}	4.46×10^{-2}
0.40	1.43×10^{-2}	2.00×10^{-2}	2.66×10^{-2}	3.40×10^{-2}	4.22×10^{-2}
0.41	1.35×10^{-2}	1.89×10^{-2}	2.51×10^{-2}	3.22×10^{-2}	4.00×10^{-2}
0.42	1.27×10^{-2}	1.78×10^{-2}	2.37×10^{-2}	3.04×10^{-2}	3.79×10^{-2}
0.43	1.19×10^{-2}	1.68×10^{-2}	2.24×10^{-2}	2.88×10^{-2}	3.59×10^{-2}
0.44	1.12×10^{-2}	1.58×10^{-2}	2.11×10^{-2}	2.72×10^{-2}	3.40×10^{-2}
0.45	1.06×10^{-2}	1.49×10^{-2}	2.00×10^{-2}	2.58×10^{-2}	3.23×10^{-2}
0.46	9.98×10^{-3}	1.41×10^{-2}	1.89×10^{-2}	2.44×10^{-2}	3.06×10^{-2}
0.47	9.41×10^{-3}	1.33×10^{-2}	1.79×10^{-2}	2.31×10^{-2}	2.90×10^{-2}
0.48	8.88×10^{-3}	1.26×10^{-2}	1.69×10^{-2}	2.19×10^{-2}	2.76×10^{-2}
0.49	8.38×10^{-3}	1.19×10^{-2}	1.60×10^{-2}	2.08×10^{-2}	2.62×10^{-2}
0.50	7.92×10^{-3}	1.12×10^{-2}	1.52×10^{-2}	1.97×10^{-2}	2.48×10^{-2}
0.51	7.48×10^{-3}	1.06×10^{-2}	1.44×10^{-2}	1.87×10^{-2}	2.36×10^{-2}
0.52	7.07×10^{-3}	1.01×10^{-2}	1.36×10^{-2}	1.77×10^{-2}	2.24×10^{-2}
0.53	6.68×10^{-3}	9.53×10^{-3}	1.29×10^{-2}	1.68×10^{-2}	2.13×10^{-2}
0.54	6.32×10^{-3}	9.02×10^{-3}	1.22×10^{-2}	1.60×10^{-2}	2.02×10^{-2}
0.55	5.98×10^{-3}	8.54×10^{-3}	1.16×10^{-2}	1.52×10^{-2}	1.92×10^{-2}
0.56	5.65×10^{-3}	8.09×10^{-3}	1.10×10^{-2}	1.44×10^{-2}	1.83×10^{-2}
0.57	5.35×10^{-3}	7.66×10^{-3}	1.04×10^{-2}	1.37×10^{-2}	1.73×10^{-2}
0.58	5.06×10^{-3}	7.13×10^{-3}	9.49×10^{-3}	1.22×10^{-2}	1.54×10^{-2}
0.59	4.39×10^{-3}	6.09×10^{-3}	8.10×10^{-3}	1.04×10^{-2}	1.31×10^{-2}
0.60	3.82×10^{-3}	5.30×10^{-3}	7.04×10^{-3}	9.08×10^{-3}	1.14×10^{-2}
0.61	3.37×10^{-3}	4.67×10^{-3}	6.21×10^{-3}	8.00×10^{-3}	1.01×10^{-2}
0.62	3.01×10^{-3}	4.17×10^{-3}	5.54×10^{-3}	7.14×10^{-3}	8.97×10^{-3}
0.63	2.70×10^{-3}	3.74×10^{-3}	4.98×10^{-3}	6.42×10^{-3}	8.06×10^{-3}
0.64	2.44×10^{-3}	3.38×10^{-3}	4.50×10^{-3}	5.80×10^{-3}	7.29×10^{-3}
0.65	2.21×10^{-3}	3.07×10^{-3}	4.08×10^{-3}	5.26×10^{-3}	6.62×10^{-3}
0.66	2.01×10^{-3}	2.79×10^{-3}	3.72×10^{-3}	4.79×10^{-3}	6.03×10^{-3}
0.67	1.84×10^{-3}	2.55×10^{-3}	3.40×10^{-3}	4.38×10^{-3}	5.51×10^{-3}
0.68	1.68×10^{-3}	2.34×10^{-3}	3.11×10^{-3}	4.01×10^{-3}	5.05×10^{-3}
0.69	1.55×10^{-3}	2.14×10^{-3}	2.86×10^{-3}	3.68×10^{-3}	4.64×10^{-3}
0.70	1.42×10^{-3}	1.97×10^{-3}	2.63×10^{-3}	3.39×10^{-3}	4.27×10^{-3}
0.71	1.31×10^{-3}	1.82×10^{-3}	2.42×10^{-3}	3.12×10^{-3}	3.93×10^{-3}
0.72	1.21×10^{-3}	1.68×10^{-3}	2.23×10^{-3}	2.88×10^{-3}	3.63×10^{-3}
0.73	1.11×10^{-3}	1.55×10^{-3}	2.06×10^{-3}	2.66×10^{-3}	3.35×10^{-3}
0.74	1.03×10^{-3}	1.43×10^{-3}	1.90×10^{-3}	2.46×10^{-3}	3.09×10^{-3}
0.75	9.51×10^{-4}	1.32×10^{-3}	1.76×10^{-3}	2.27×10^{-3}	2.86×10^{-3}
0.76	8.78×10^{-4}	1.22×10^{-3}	1.63×10^{-3}	2.10×10^{-3}	2.64×10^{-3}
0.77	8.12×10^{-4}	1.13×10^{-3}	1.50×10^{-3}	1.94×10^{-3}	2.44×10^{-3}
0.78	7.50×10^{-4}	1.04×10^{-3}	1.39×10^{-3}	1.79×10^{-3}	2.26×10^{-3}
0.79	6.92×10^{-4}	9.61×10^{-4}	1.28×10^{-3}	1.65×10^{-3}	2.08×10^{-3}
0.80	6.38×10^{-4}	8.86×10^{-4}	1.18×10^{-3}	1.53×10^{-3}	1.92×10^{-3}
0.81	5.88×10^{-4}	8.16×10^{-4}	1.09×10^{-3}	1.41×10^{-3}	1.77×10^{-3}
0.82	5.41×10^{-4}	7.51×10^{-4}	1.00×10^{-3}	1.29×10^{-3}	1.63×10^{-3}
0.83	4.97×10^{-4}	6.89×10^{-4}	9.20×10^{-4}	1.19×10^{-3}	1.50×10^{-3}
0.84	4.55×10^{-4}	6.32×10^{-4}	8.43×10^{-4}	1.09×10^{-3}	1.37×10^{-3}
0.85	4.16×10^{-4}	5.77×10^{-4}	7.71×10^{-4}	9.95×10^{-4}	1.25×10^{-3}
0.86	3.79×10^{-4}	5.26×10^{-4}	7.02×10^{-4}	9.07×10^{-4}	1.14×10^{-3}
0.87	3.44×10^{-4}	4.77×10^{-4}	6.37×10^{-4}	8.23×10^{-4}	1.04×10^{-3}
0.88	3.11×10^{-4}	4.31×10^{-4}	5.76×10^{-4}	7.44×10^{-4}	9.37×10^{-4}
0.89	2.79×10^{-4}	3.88×10^{-4}	5.18×10^{-4}	6.69×10^{-4}	8.42×10^{-4}
0.90	2.49×10^{-4}	3.46×10^{-4}	4.63×10^{-4}	5.97×10^{-4}	7.53×10^{-4}

Chapter 2

Modelling interactions in high-dimensional data with Backtracking

2.1 Introduction

In recent years, there has been a lot of progress in the field of high-dimensional regression. Much of the development has centred around the Lasso (Tibshirani, 1996), which given a vector of responses $\mathbf{Y} \in \mathbb{R}^n$ and design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, solves

$$(\hat{\mu}, \hat{\boldsymbol{\beta}}) := \underset{(\mu, \boldsymbol{\beta}) \in \mathbb{R} \times \mathbb{R}^p}{\arg\min} \{ \tfrac{1}{2n} \|\mathbf{Y} - \mu \mathbf{1} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \}, \tag{2.1}$$

where $\mathbf{1}$ is an n-vector of ones and the regularisation parameter λ controls the relative contribution of the penalty term to the objective. The many extensions of the Lasso allow most familiar models from classical (low-dimensional) statistics to now be fitted in situations where the number of variables p may be tens of thousands and even greatly exceed the number of observations n (see the recent monograph Bühlmann and van de Geer (2011a) and references therein).

However, despite the advances, fitting models with interactions remains a challenge. Two issues that arise are:

(i) Since there are $p(p-1)/2$ possible first-order interactions, the main effects can be swamped by the vastly more numerous interaction terms and without proper regularisation, stand little chance of being selected in the final model (see Figure 2.1b).

(ii) Monitoring the coefficients of all the interaction terms quickly becomes infeasible as p runs into the thousands.

Though in many situations, the 'true model' may not contain strong interactions, note that particularly in high-dimensional settings where it is not even clear which variables are important, one is unlikely to be confident about the presence or absence of interactions. Therefore there is a real need for developing methods that detect interactions automatically.

For situations where $p < 1000$ or thereabouts and the case of two-way interactions, a lot of work has been done in recent years to address this need. To tackle (i), many of the proposals use

penalty functions and constraints designed to enforce that if an interaction term is in the fitted model, one or both main effects are also present (Bach et al., 2012a,b; Bien et al., 2013; Jenatton et al., 2011; Lin and Zhang, 2006; Radchenko and James, 2010; Yuan et al., 2009; Zhao et al., 2009). See also Turlach (2004) and Yuan et al. (2007), which consider modifications of the LAR algorithm Efron et al. (2004) that impose this type of condition.

In the moderate-dimensional setting that these methods are designed for, the computational issue (ii) is just about manageable. However, when p is larger—the situation of interest in this work—it becomes necessary to narrow the search for interactions. Comparatively little work has been done on fitting models with interactions to data of this sort of dimension.

One option is to screen for important variables and only consider interactions involving the selected set. Wu et al. (2010) and others take this approach: the Lasso is first used to select main effects; then interactions between the selected main effects are added to the design matrix, and the Lasso is run once more to give the final model.

The success of this method relies on all main effects involved in interactions being selected in the initial screening stage. However, this may well not happen. Certain interactions may need to be included in the model before some main effects can be selected. To address this issue, Bickel et al. (2010) propose a procedure involving sequential Lasso fits which, for some predefined number K, selects K variables from each fit and then adds all interactions between those variables as candidate variables for the following fit. The process continues until all interactions to be added are already present. However, it is not clear how one should choose K: a large K may result in a large number of spurious interactions being added at each stage, whereas a small K could cause the procedure to terminate before it has had a chance to include important interactions.

Rather than adding interactions in one or more distinct stages, when variables are selected in a greedy fashion, the set of candidate interactions can be updated after each selection. This dynamic updating of interactions available for selection is present in the popular MARS procedure of Friedman (1991). One problem with this approach is that particularly in high-dimensional situations, greedy selection can produce unstable final models and predictive performance can suffer as a consequence. Bagging (Breiman, 1996) can be very successful at reducing instability (such as in Random Forests (Breiman, 2001)), but the final aggregated model is likely to be difficult to interpret as it will tend to contain small contributions from a great number of variables.

In this chapter, we propose a new method we call Backtracking, for incorporating a similar model building strategy to that of MARS into methods based on sparsity-inducing penalty functions. Such methods can be more stable than greedy forward selection approaches (see Efron et al. (2004)), and as a consequence tend to give better predictive performance. When used with the Lasso, Backtracking begins by computing the Lasso solution path, decreasing λ from ∞. A second solution path, P_2, is then produced, where the design matrix contains all main effects, and also the interaction between the first two active variables in the initial path. Continuing iteratively, subsequent solution paths $P_3, \ldots, P_T$ are computed where the set of main effects and interactions in the design matrix for the k^{th} path is determined based on the previous path P_{k-1}. Thus if in the third path, a key interaction was included and so variable selection was then more accurate, the selection of interactions for all future paths would benefit. In this way information is used as soon as it is available, rather than at discrete stages as with the method of Bickel et al. (2010). In addition, if all important interactions have already been included by P_3, we have a solution path unhindered by the addition of further spurious interactions.

It may seem that a drawback of our proposed approach is that the computational cost of

producing all T solution paths will usually be unacceptably large. However, computation of the full collection of solution paths is very fast indeed. This is because rather than computing each of the solution paths from scratch, for each new solution path P_{k+1}, we first track along the previous path P_k to find where P_{k+1} departs from P_k. This is the origin of the name Backtracking. Typically, checking whether a given trial solution is on a solution path requires much less computation than calculating the solution path itself, and so this Backtracking step can be made very fast. Furthermore, when the solution paths do separate, the tail portions of the paths can be computed in parallel.

The rest of the chapter is organised as follows. In Section 2.2 we describe an example which provides some motivation for our Backtracking method. In Section 2.3 we develop our method in the context of the Lasso for the linear model. We build up to our final algorithm in stages, presenting the final version in Section 2.3.2.2. In Section 2.4, we describe how our method can be extended beyond the case of the Lasso for the linear model. In Section 2.5 we report the results of some simulation experiments and real data analyses that demonstrate the effectiveness of Backtracking. Finally, in Section 2.6, we present some theoretical results which aim to give a deeper understanding of the way in which Backtracking works. Proofs are collected in the appendix of this chapter.

2.2 Motivation

In this section we introduce a toy example where approaches that select candidate interactions based on selected main effects will tend to perform poorly. The data follow a linear model with interactions,

$$Y_i = \sum_{j=1}^{6} \beta_j X_{ij} + \beta_7 X_{i1} X_{i2} + \beta_8 X_{i3} X_{i4} + \beta_9 X_{i5} X_{i6} + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2), \quad i = 1, \ldots, n, \quad (2.2)$$

with the matrix of predictors $\mathbf{X}$ designed such that X_{i5} is uncorrelated with the response, and also with any linear combination of $\{X_{ij} : j \neq 5\}$.

The construction of $\mathbf{X}$ is as follows. First, consider (Z_{i1}, Z_{i2}, Z_{i3}) generated from a mean zero multivariate normal distribution with $\mathrm{Var}(Z_{ij}) = 1$, $j = 1, 2, 3$, $\mathrm{Cov}(Z_{i1}, Z_{i2}) = 0$ and $\mathrm{Cov}(Z_{i1}, Z_{i3}) = \mathrm{Cov}(Z_{i2}, Z_{i3}) = 1/2$. Independently generate R_{i1} and R_{i2} each of which takes only the values $\{-1, 1\}$, each with probability $1/2$. We form the i^{th} row of the design matrix as follows:

$$\begin{aligned}
X_{i1} &= R_{i1} \operatorname{sgn}(Z_{i1}) |Z_{i1}|^{1/4}, \\
X_{i2} &= R_{i1} |Z_{i1}|^{3/4}, \\
X_{i3} &= R_{i2} \operatorname{sgn}(Z_{i2}) |Z_{i2}|^{1/4}, \\
X_{i4} &= R_{i2} |Z_{i2}|^{3/4}, \\
X_{i5} &= Z_{i3}.
\end{aligned}$$

The remaining X_{ij}, $j = 6, \ldots, p$ are independently generated from a standard normal distribution. Note that the random signs R_{i1} and R_{i2} ensure that X_{i5} is uncorrelated with each of $X_{i1}, \ldots, X_{i4}$. Furthermore, the fact that $X_{i1} X_{i2} = Z_{i1}$ and $X_{i3} X_{i4} = Z_{i2}$, means that when $\beta_5 = -\frac{1}{2}(\beta_7 + \beta_8)$, X_{i5} is uncorrelated with the response.

If we first select important main effects using the Lasso, for example, when p is large it is very unlikely that variable 5 will be selected. Then if we add all two-way interactions between the selected variables and fit the Lasso once more, the interaction between variables 5 and 6 will not be included. Of course, one can again add interactions between selected variables and compute another Lasso fit, and then there is a chance the interaction will be selected. Thus it is very likely that at least three Lasso fits will be needed in order to select the right variables.

Figure 2.1a shows the result of applying the Lasso to data generated according to (2.2) with 200 independent and identically distributed (i.i.d.) observations, $p = 500$, σ chosen to give a signal-to-noise ratio (SNR) of 4, and

$$\boldsymbol{\beta} = (-1.25, -0.75, 0.75, -0.5, -2, 1.5, 2, 2, 1)^T.$$

As expected, we see variable 5 is nowhere to be seen and instead many unwanted variables are selected as λ is decreased. Figure 2.1b illustrates the effect of including all $p(p-1)/2$ possible interactions in the design matrix. Even in our rather moderate-dimensional situation, we are not able to recover the true signal. Though all the true interaction terms are selected, now both variables 4 and 5 are not present in the solution paths and many false interactions are selected.

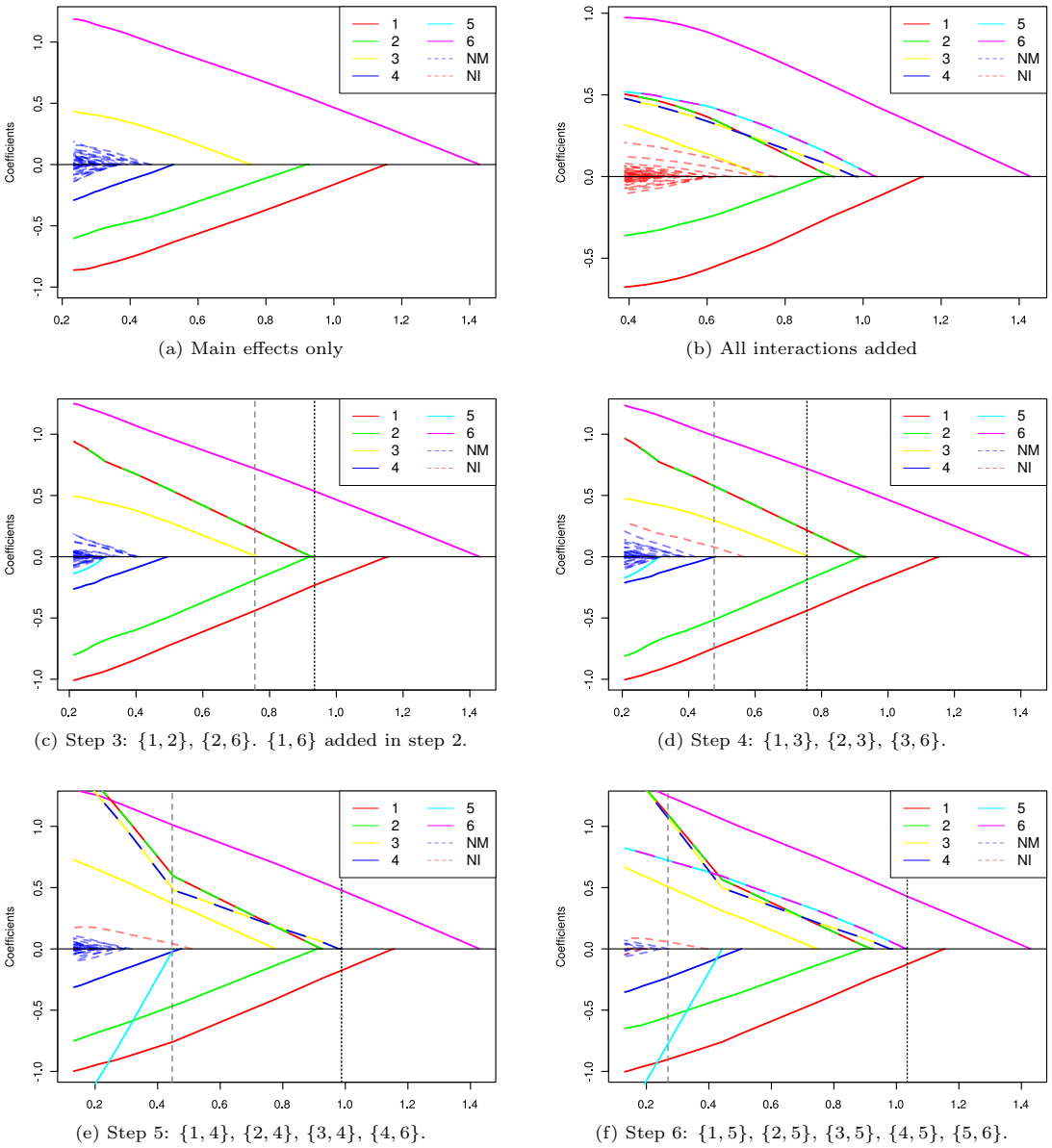
Although this example is rather contrived, it illustrates how sometimes the right interactions need to be augmented to the design matrix in order for certain variables to be selected. Even when interactions are only present if the corresponding main effects are too, main effects can be missed by a procedure that does not consider interactions. In fact, we can see the same phenomenon occurring when the design matrix has i.i.d. Gaussian entries (see Section 2.5.1). In our case here, except purely by chance, variable 5 can only be selected by the Lasso if either the interactions between variables 1 and 2 or 3 and 4 are present in the design matrix. We also see that multiple Lasso fits might be needed to have any chance of selecting the right model.

This raises the question of which tuning parameters to use in the multiple Lasso fits. One option, which we shall refer to as the iterated Lasso, is to select tuning parameters by cross-validation each time. A drawback of this approach, though, is that the number of interactions to add can be quite large if cross-validation chooses a large active set. This is often the case when the presence of interactions makes some important main effects hard to distinguish from noise variables in the initial Lasso fit. Then cross-validation may choose a low λ in order to try to select those variables, but this would result in many noise variables also being included in the active set.

We take an alternative approach here and include suspected interactions in the design matrix as soon as possible. That is, if we progress along the solution path from $\lambda = \infty$, and two variables enter the model, we immediately add their interaction to the design matrix and start computing the Lasso again. We could now disregard the original path, but there is little to lose, and possibly much to gain, in continuing the original path in parallel with the new one. We can then repeat this process, adding new interactions when necessary, and restarting the Lasso, whilst still continuing all previous paths in parallel. We show in the next section how computation can be made very fast since many of these solution paths will share the same initial portions.

2.3 Backtracking with the Lasso

In this section we introduce a version of the Backtracking algorithm applied to the Lasso (2.1). First, we present a naive version of the algorithm, which is easy to understand. Later in Sec-



(a) Main effects only

(b) All interactions added

(c) Step 3: $\{1,2\}$, $\{2,6\}$. $\{1,6\}$ added in step 2.

(d) Step 4: $\{1,3\}$, $\{2,3\}$, $\{3,6\}$.

(e) Step 5: $\{1,4\}$, $\{2,4\}$, $\{3,4\}$, $\{4,6\}$.

(f) Step 6: $\{1,5\}$, $\{2,5\}$, $\{3,5\}$, $\{4,5\}$, $\{5,6\}$.

Figure 2.1: For data generated as described in Section 2.2, the coefficient paths against λ of the Lasso with main effects only, (a); the Lasso with all interactions added, (b); and Backtracking with $k = 3, \ldots, 6$, ((c)–(d)); when applied to the example in Section 2.2. Below the Backtracking solution paths we give $C_k \setminus C_{k-1}$: the interactions which have been added in the current step. The solid red, green, yellow, blue, cyan and magenta lines trace the coefficients of variables $1, \ldots, 6$ respectively, with the alternately coloured lines representing the corresponding interactions. The dotted blue and red coefficient paths indicate noise main effect ('NM') and interaction ('NI') terms respectively. Vertical dotted black and dashed grey lines give the values of λ_k^{start} and λ_k^{add} respectively.

tion 2.3.2, we show that this algorithm performs a large number of unnecessary calculations, and we give a far more efficient version.

2.3.1 A naive algorithm

As well as a base regression procedure, the other key ingredient that Backtracking requires is a way of suggesting candidate interactions based on selected main effects, or more generally a way of suggesting higher order interactions based on lower order interactions. In order to discuss this and present our algorithm, we first introduce some notation concerning interactions.

Let $\mathbf{X}$ be the original $n \times p$ design matrix, with no interactions. In order to consider interactions in our models, rather than indexing variables by a single number j, we use subsets of $\{1, \ldots, p\}$. Thus by variable $\{1, 2\}$, we mean the interaction between variables 1 and 2, or in our new notation, variables $\{1\}$ and $\{2\}$. As we are using the Lasso as the base regression procedure here, interaction $\{1, 2\}$ will be the componentwise product of the first two columns of $\mathbf{X}$. We will write $\mathbf{X}_v \in \mathbb{R}^n$ for variable v, so in particular, $\mathbf{X}_{\{j\}}$ will be the j^{th} column of $\mathbf{X}$.

The choice of whether and how to scale and centre interactions and main effects can be a rather delicate one, where domain knowledge may play a key role. In this work, we will centre all main effects, and scale them to have ℓ_2-norm $\sqrt{n}$. The interactions will be created using these centred and scaled main effects, and they themselves will also be centred and scaled to have ℓ_2-norm $\sqrt{n}$. Note that when the design matrix is sparse, for example, it may make more sense not to centre and scale at all.

Let us denote the power set operator by $\mathcal{P}$. For $C \subseteq \mathcal{P}(\{1, \ldots, p\})$, we can form a modified design matrix $\mathbf{X}_C$, where the columns of $\mathbf{X}_C$ are given by the variables in C, centred and scaled as described above. Thus C is the set of candidate variables available for selection when design matrix $\mathbf{X}_C$ is used. This subsetting operation will always be taken to have been performed before any further operations on the matrix, so in particular $\mathbf{X}_C^T$ means $(\mathbf{X}_C)^T$.

We will consider all associated vectors and matrices as indexed by variables, so we may speak of component $\{1, 2\}$ of $\boldsymbol{\beta}$, denoted $\beta_{\{1,2\}}$, if $\boldsymbol{\beta}$ were multiplying a design matrix which included $\{1, 2\}$. Further, for any collection of variables A, we will write $\boldsymbol{\beta}_A$ for the subvector whose components are those indexed by A. To represent an arbitrary variable, we shall use v or u rather than j, to remind us that variables are now indexed by sets.

We will often need to express the dependence of the Lasso solution $\hat{\boldsymbol{\beta}}$ (2.1) on the tuning parameter λ and the design matrix used. We shall write $\hat{\boldsymbol{\beta}}(\lambda, C)$ when $\mathbf{X}_C$ is the design matrix. We will denote the set of active components of a solution $\hat{\boldsymbol{\beta}}$ by $\mathcal{A}(\hat{\boldsymbol{\beta}}) = \{v : \hat{\beta}_v \neq 0\}$.

We now introduce a function $\mathcal{I}$ that given a set of variables A, suggests a set of interactions to add to the design matrix. The choice of $\mathcal{I}$ we use here is as follows:

$$\mathcal{I}(A) = \{v \subseteq \{1, \ldots, p\} : \text{ for all } u \subsetneq v,\, u \neq \emptyset,\, u \in A\}.$$

In other words, $\mathcal{I}(A)$ is the set of variables not in A, all of whose corresponding lower order interactions are present in A. For example, $\mathcal{I}(\{\{1\}, \{2\}\}) = \{\{1, 2\}\}$, and $\mathcal{I}(\{\{1\}, \{2\}.\{3\}\}) = \{\{1, 2\}, \{2, 3\}, \{1, 3\}\}$. Note $\{1, 2, 3\} \notin \mathcal{I}(\{\{1\}, \{2\}, \{3\}\})$ as the lower order interaction $\{1, 2\}$ of $\{1, 2, 3\}$ is not in $\{\{1\}, \{2\}, \{3\}\}$, for example. Other choices for $\mathcal{I}$ can be made, and we discuss some further possibilities in Section 2.4.

As mentioned in section 2.1, Backtracking relies on a path algorithm for computing, in our case here, the solution path of (2.1). Several such algorithms are available: the homotopy method of

Osborne et al. (2000a) and Osborne et al. (2000b), and the closely related LARS algorithm (Efron et al., 2004) make use of the piecewise linearity of the solution path, and are able to compute it exactly. More recently, coordinate descent methods have been demonstrated to be significantly faster in high-dimensional situations, and can be applied to fit generalised linear models and others (Friedman et al., 2010). These, however, compute the solution path at a discrete set of λ values. Any of these algorithms are suitable for use in conjunction with Backtracking, but we will focus our discussion on the coordinate descent method, because of the advantages already mentioned.

We are now in a position to introduce the naive version of our Backtracking algorithm applied to the Lasso (Algorithm 1). We will assume that $\mathbf{Y}$ is centred in addition to the design matrix, so no intercept term is necessary.

Algorithm 1 A naive version of Backtracking for the Lasso. The highlighted line is needed for modifications to be presented in Section 2.3.2.2 and should be ignored on a first reading.

1: **Input** Design matrix $\mathbf{X}$ and grid of λ values $\lambda_1 > \cdots > \lambda_L$
2: Set the initial candidate set to contain just the main effects: $C_1 \leftarrow \{\{1\}, \ldots, \{p\}\}$
3: Initialise the index variables for the candidate sets and λ values: $k \leftarrow 1$; $l \leftarrow 1$
4: Initialise the set of interactions generated by the current active set: $I \leftarrow \emptyset$
5: Set the initial vector of residuals to be $\mathbf{Y}$: $\mathbf{R}(1) \leftarrow \mathbf{Y}$
6: **while** $I \subseteq C_k$ **and** $l \leq L$ **and** $\mathbf{R}(l) \neq 0$ **do**
7: Compute $\hat{\boldsymbol{\beta}}$ with design matrix $\mathbf{X}_{C_k}$ at λ_l, and store this in $\mathbf{B}(l)$: $\mathbf{B}(l) \leftarrow \hat{\boldsymbol{\beta}}(\lambda_l, C_k)$
8: Let I be the set of interactions generated from the current active set: $I \leftarrow \mathcal{I}(\mathcal{A}(\mathbf{B}(l)))$
9: Store the current residual in $\mathbf{R}(l)$: $\mathbf{R}(l) \leftarrow \mathbf{Y} - \mathbf{X}_{C_k}\mathbf{B}(l)$
10: Let $K(l)$ be the index of the largest candidate set to have had $\hat{\boldsymbol{\beta}}(\lambda_l, C_k)$ computed so far: $K(l) \leftarrow k$
11: Increment l: $l \leftarrow l + 1$
12: **end while**
13: Let l_k^{add} record the last value of l at which $\hat{\boldsymbol{\beta}}$ was computed: $l_k^{\text{add}} \leftarrow l - 1$
14: Collect the solution path obtained thus far in P_k: $P_k \leftarrow (\mathbf{B}(l) : l \leq l_k^{\text{add}})$
15: **if** $I \nsubseteq C_k$ **then**
16: Spawn a parallel process to continue the solution path P_k using the current design matrix X_{C_k} until either a perfect fit is reached, or the path reaches the end of the grid of λ values
17: Add interactions I to the current set of candidates: $C_{k+1} \leftarrow C_k \cup I$
18: Increment k: $k \leftarrow k + 1$
19: Reset $l \leftarrow 1$ and **go to** line 6
20: **end if**
21: Set T to be the total number of candidate sets: $T \leftarrow k$
22: **return** Completed paths $(P_1, \ldots, P_T)$

Looking at the while loop in Algorithm 1, we see that given a set of candidates C_k, the algorithm decrements λ until either the active set generates interaction terms currently not in C_k, i.e. until $I \nsubseteq C_k$, or until a perfect fit or λ_L is reached. If the former holds, a process is spawned to continue the current solution path, an updated set of candidates, C_{k+1} is formed, and λ is decreased from λ_1 once more. The quantity l_k^{add} records the value of λ at which interaction terms were added to the set of candidates C_k. Termination of the algorithm is guaranteed since $|C_k|$ increases at each iteration, and it cannot exceed $2^p - 1$. Though in practice, termination will typically occur long before $C_k = \mathcal{P}(\{1, \ldots, p\}) \setminus \{\emptyset\}$, for both computational and statistical reasons, we recommend terminating the algorithm if $|C_k| - p$ becomes too large (see Section 2.3.1.1). We note also that the possible interactions to consider can easily be restricted to, say, first-order interactions.

The final output of the algorithm is a collection of solution paths, each one of which corresponds to a different set of candidates. Figures 2.1c–2.1f show steps 3–6 (i.e. $k = 3, \ldots, 6$) of Backtracking applied to the example described in Section 2.2. Note that Figure 2.1a is in fact step 1. Step 2 is not shown as the plot looks identical to that in Figure 2.1a. We see that when $k = 6$, we have a solution path where all the true variable and interaction terms are active before any noise variables enter the coefficient plots.

In the following section we explain how one can choose a final estimator from this collection of paths.

2.3.1.1 Cross-validation

Where the Lasso has one tuning parameter, with Backtracking we have two: λ and k, the rank of the path. When using the Lasso, the tuning parameter used to construct the final estimator is typically chosen by cross-validation.

In many cases we may be performing Backtracking and forcing early termination if C_k gets too large. If the (λ, k) pair with minimal cross-validation score has k less than each of the maximum number of steps reached on each of the folds, one can think of this as a local minimiser of the cross-validation score when the size of C_k is unrestricted. Often, this may in fact be the global minimiser, and in these cases calculating the full collection of solution paths without early termination would result in unnecessary computation. Even when this is not true, since one expects the variance of $\mathbf{X}_{C_k}\hat{\boldsymbol{\beta}}(\lambda, C_k)$ to increase with k, there are statistical reasons one might prefer the restricted minimiser.

In fact, the same reasoning supports terminating solution paths when the active set gets large and so selecting λ to be a possibly local minimiser of the cross-validation score. Since the bulk of the computation in the Lasso solution path occurs when the active set is large, this can result in big computational savings.

In some situations, rather than using the final estimator from the Lasso, it is often better to use the active sets from the Lasso solution paths, and apply a further estimation procedure to subsets of the original design matrix whose columns are given by the variables in the active sets. Sensible candidates for this second estimation procedure include ordinary least squares and a further Lasso fit; these choices giving methods known as (a variant of) the LARS–OLS hybrid (Efron et al., 2004) and the relaxed Lasso (Meinshausen, 2007) respectively. An alternative to this approach is the adaptive Lasso of Zou (2006). All of these methods can be used in conjunction with Backtracking and for our numerical results in Section 2.5.1 we use the LARS–OLS hybrid.

2.3.2 Speeding up computation

2.3.2.1 An improved algorithm

The process of performing mutiple Lasso fits is computationally cumbersome, and an immediate gain in efficiency can be realised by noticing that the final collection of solution paths is in fact a tree of solutions: many of the solution paths computed will share the same initial portions. To discuss this, we first recall that by considering subgradients or simply one-sided directional

derivatives, $\hat{\boldsymbol{\beta}}$ is a solution to (2.1) when the design matrix is $\mathbf{X}_C$ if and only if

$$\tfrac{1}{n}\mathbf{X}_v^T(\mathbf{Y} - \mathbf{X}_C\hat{\boldsymbol{\beta}}) = \lambda \mathrm{sgn}(\hat{\beta}_v) \quad \text{for } \hat{\beta}_v \neq 0 \tag{2.3}$$

$$\tfrac{1}{n}|\mathbf{X}_v^T(\mathbf{Y} - \mathbf{X}_C\hat{\boldsymbol{\beta}})| \leq \lambda \quad \text{for } \hat{\beta}_v = 0. \tag{2.4}$$

Note the $\hat{\mu}\mathbf{X}_v^T\mathbf{1}$ term vanishes as the columns of $\mathbf{X}_C$ are assumed to be centred. These are often referred to as the KKT conditions for the Lasso in the literature.

Write λ_k^{add} for $\lambda_{l_k^{\text{add}}}$ and set $\lambda_{k+1}^{\text{start}} = \lambda_{l_{k+1}^{\text{start}}}$ to be the minimal element of $\{\lambda_1, \ldots, \lambda_k^{\text{add}}\}$ such that the following holds for all $\lambda \geq \lambda_{k+1}^{\text{start}}$:

$$\tfrac{1}{n}\|\mathbf{X}_{C_{k+1}\setminus C_k}^T(\mathbf{Y} - \mathbf{X}_{C_k}\hat{\boldsymbol{\beta}}(\lambda, C_k))\|_\infty \leq \lambda. \tag{2.5}$$

Then crucially, for all $\lambda \geq \lambda_{k+1}^{\text{start}}$,

$$\hat{\boldsymbol{\beta}}_{C_{k+1}\setminus C_k}(\lambda, C_{k+1}) = \mathbf{0} \quad \text{and}$$
$$\hat{\boldsymbol{\beta}}_{C_k}(\lambda, C_{k+1}) = \hat{\boldsymbol{\beta}}(\lambda, C_k).$$

In words, no variables in $C_{k+1} \setminus C_k$ are active before the point $\lambda_{k+1}^{\text{start}}$ on the solution path. Note the existence of $\lambda_{k+1}^{\text{start}}$ is guaranteed provided λ_1 is sufficiently large, since $\hat{\boldsymbol{\beta}}(\lambda, C_{k+1}) = 0$ and $\hat{\boldsymbol{\beta}}(\lambda, C_k) = 0$ for λ sufficiently large. The modifications needed when λ_1 is not large enough are trivial and we do not discuss them here. We can use this knowledge to replace line 19 in Algorithm 1, which sets l to 1 after the set of candidates has changed, with Algorithm 2 below. We see that computation of the solution path need only start again after l_{k+1}^{start} which may in fact be as large as l_k^{add}, the last value of l at which $\hat{\boldsymbol{\beta}}(\lambda_l, C_k)$ was computed.

Algorithm 2 An improvement on line 19 of Algorithm 1.

$l_{k+1}^{\text{start}} \leftarrow 1$
while $\|\mathbf{X}_{C_{k+1}\setminus C_k}^T \mathbf{R}(l_{k+1}^{\text{start}})\|_\infty \leq n\lambda_{l_{k+1}^{\text{start}}}$ **and** $l_{k+1}^{\text{start}} \leq l_k^{\text{add}}$ **do**
  $l_{k+1}^{\text{start}} \leftarrow l_{k+1}^{\text{start}} + 1$
end while
$l \leftarrow l_{k+1}^{\text{start}} + 1$ and **go to** line 6

Notice that the condition to be checked in the while loop involves the multiplication of a $|C_{k+1} \setminus C_k| \times n$ matrix by a vector of length n, and thus has computational complexity $O(|C_{k+1} \setminus C_k|n)$. This computation is very fast, especially compared to the alternative of calculating $\hat{\boldsymbol{\beta}}(\lambda_l, C_{k+1})$. Furthermore, the while loop can, if necessary, be executed in parallel, making the 'Backtracking step' very fast indeed. However, since parallel computing power may well need to be reserved for processing the various jobs assigned to them, in the next section we present another version of the Backtracking algorithm that allows us to bypass most of the calculations in the while loop.

2.3.2.2 The final algorithm

When the current set of candidates changes from C_k to C_{k+1}, Algorithm 2 searches along P_k from λ_1 to λ_k^{add}, checking the validity of each point on the path as a solution $\hat{\boldsymbol{\beta}}(\lambda, C_{k+1})$ with the enlarged candidate set. If the search reaches λ_k^{add}, we would have done a fair few calculations simply to end up, quite literally, back where we started. Motivated by this observation, in Algorithm 3 we present a further improvement on line 19 of Algorithm 1. Note that Algorithm 3 below redefines

l_k^{start} and l_k^{add}, and hence also λ_k^{start} and λ_k^{add}.

Algorithm 3 A further improvement on line 19 of Algorithm 1.

if $\|\mathbf{X}_{C_{k+1}\setminus C_k}^T \mathbf{R}(l_k^{\text{add}})\|_\infty \leq n\lambda_k^{\text{add}}$ **then**
  $l_{k+1}^{\text{start}} \leftarrow l_k^{\text{add}}$
else
  Set l_{k+1}^{start} to be any $l' < l_k^{\text{add}}$ such that
  $\|\mathbf{X}_{C_{k+1}\setminus C_{K(l')}}^T \mathbf{R}(l')\|_\infty \leq n\lambda_{l'}$ **and** $\|\mathbf{X}_{C_{k+1}\setminus C_{K(l'+1)}}^T \mathbf{R}(l'+1)\|_\infty > n\lambda_{l'+1}$,
  (so the KKT conditions hold at l' and consequently $\hat{\boldsymbol{\beta}}_{C_{k+1}\setminus C_{K(l')}}(\lambda_{l'}, C_{k+1}) = 0$, but they are violated at $l'+1$)
end if
$l \leftarrow l_{k+1}^{\text{start}} + 1$ and **go to** line 6

To explain Algorithm 3 in words, we first check whether P_{k+1} can simply be made to extend P_k. If not, we search for *any* point where P_k and P_{k+1} agree but after which they disagree, rather than the first such point. Such a search can be implemented by a bisection method which would terminate in at most $O(\log_2 l_k^{\text{add}})$ steps. Since $l_k^{\text{add}} \leq L$ and L would not usually be more than a few hundred, this modified search is very cheap. Note that when checking the KKT conditions at $\lambda_{l'}$, we need to verify that (2.4) holds for all $v \in C_{k+1} \setminus C_{K(l')}$, where $C_{K(l')}$ is the largest candidate set C to have had $\hat{\boldsymbol{\beta}}(\lambda_{l'}, C)$ computed at that point in the progression of the algorithm (see line 10 of Algorithm 1).

A possible disadvantage of this approach is that the solution paths computed will only be approximate. If we let $\tilde{\boldsymbol{\beta}}(\lambda, C)$, defined for $C = C_1, \ldots, C_T$, give the solution paths obtained by Backtracking with bisection search, then we only know that for $\lambda \leq \lambda_{k+1}^{\text{start}}$ we have $\tilde{\beta}(\lambda, C_{k+1}) = \hat{\boldsymbol{\beta}}(\lambda, C_{k+1})$, the true solution. For $\lambda > \lambda_{k+1}^{\text{start}}$, this need not be the case, as variables in $C_{k+1} \setminus C_k$ could have entered the solution path $\hat{\boldsymbol{\beta}}(\lambda, C_{k+1})$ at an earlier stage, but then left at or before $\lambda_{k+1}^{\text{start}}$. In practice, variables leaving and re-entering the solution path does not happen too often. In fact, one might say that an active variable that is about to leave the active set should be regarded as suspicious, and it makes sense to include it only at a later stage along the path. Furthermore, for the theory we develop in Section 2.6, we lose nothing by using the approximate solutions. For these reasons, we prefer to use the bisection search method, and from now on, we will use Backtracking to mean precisely this variant.

One computational shortcut we have not mentioned yet concerns the fact that when $\lambda_{k+1}^{\text{start}} = \lambda_k^{\text{add}}$, the solution paths P_k and P_{k+1} will still agree beyond $\lambda_{k+1}^{\text{start}}$ and the solution tree will not branch at this point. In this case our algorithm, as it has been presented, will perform some unnecessary computation, though if this redundancy were removed the parallel computational complexity would remain the same. It is straightforward to modify the algorithm so that processes are only spawned at branch points of the solution tree, but the details are rather technical and we do not discuss them here.

2.4 Further applications of Backtracking

Our Backtracking algorithm has been presented in the context of the Lasso for the linear model. However, the real power of the idea is that it can be incorporated into any method that produces a path of increasingly complex sparse solutions by solving a family of convex optimisation problems parametrised by a tuning parameter. For the Backtracking step (Algorithm 3), the KKT conditions

for these optimisation problems provide a way of checking whether a given trial solution is an optimum. As in the case of the Lasso, checking whether the KKT conditions are satisfied typically requires much less computational effort than computing a solution from scratch. Below we briefly sketch some applications of Backtracking to a few of the many possible methods with which it can be used.

2.4.1 Multinomial regression

An example, which we apply to real data in Section 2.5.2, is multinomial regression with a group Lasso (Yuan and Lin, 2006) penalty. Consider n observations of a categorical response that takes J levels, and p associated covariates. Let $\mathbf{Y}$ be the indicator response matrix, with ij^{th} entry equal to 1 if the i^{th} observation takes the j^{th} level, and 0 otherwise. We model

$$\mathbb{P}(Y_{ij} = 1) := \Pi_{ij}(\boldsymbol{\mu}^*, \boldsymbol{\beta}^*; \mathbf{X}_{S^*}) := \frac{\exp\left(\mu_j^* + \left(\mathbf{X}_{S^*}\boldsymbol{\beta}_j^*\right)_i\right)}{\sum_{j'=1}^{J} \exp\left(\mu_{j'}^* + \left(\mathbf{X}_{S^*}\boldsymbol{\beta}_{j'}^*\right)_i\right)}.$$

Here μ^* is a vector of intercept terms and $\boldsymbol{\beta}^*$ is a $|S^*| \times J$ matrix of coefficients; $\boldsymbol{\beta}_j^*$ denotes the j^{th} column of $\boldsymbol{\beta}^*$. This model is over-parametrised, but regularisation still allows us produce estimates of $\boldsymbol{\mu}^*$ and $\boldsymbol{\beta}^*$ and hence also of $\boldsymbol{\Pi}$ (see Friedman et al. (2010)). When our design matrix is $\mathbf{X}_C$, these estimates are given by $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\beta}}) := \underset{\boldsymbol{\mu},\boldsymbol{\beta}}{\arg\min}\, Q(\boldsymbol{\mu}, \boldsymbol{\beta}; \lambda)$ where

$$Q(\boldsymbol{\mu}, \boldsymbol{\beta}; \lambda) := \tfrac{1}{n}\sum_{j=1}^{J} \mathbf{Y}_j^T(\mu_j\mathbf{1} + \mathbf{X}_C\boldsymbol{\beta}_j) - \tfrac{1}{n}\mathbf{1}^T \log\left(\sum_{j=1}^{J} \exp(\mu_j\mathbf{1} + \mathbf{X}_C\boldsymbol{\beta}_j\right) + \lambda \sum_{v \in C} \|(\boldsymbol{\beta}^T)_v\|_2.$$

The functions log and exp are to be understood as applied componentwise and the rows of $\boldsymbol{\beta}$ are indexed by elements of C. To derive the Backtracking step for this situation, we turn to the KKT conditions which characterise the minima of Q:

$$\begin{aligned}
\tfrac{1}{n}\{\mathbf{Y}^T - \boldsymbol{\Pi}^T(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\beta}}; \mathbf{X}_C)\}\mathbf{1} &= \mathbf{0}, \\
\tfrac{1}{n}\{\mathbf{Y}^T - \boldsymbol{\Pi}^T(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\beta}}; \mathbf{X}_C)\}\mathbf{X}_v &= -\lambda \frac{(\hat{\boldsymbol{\beta}}^T)_v}{\|(\hat{\boldsymbol{\beta}}^T)_v\|_2} \quad \text{for } (\hat{\boldsymbol{\beta}}^T)_v \neq \mathbf{0}, \\
\tfrac{1}{n}\|\{\mathbf{Y}^T - \boldsymbol{\Pi}^T(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\beta}}; \mathbf{X}_C)\}\mathbf{X}_v\|_2 &\leq \lambda \quad \text{for } (\hat{\boldsymbol{\beta}}^T)_v = \mathbf{0}.
\end{aligned}$$

Thus, analogously to (2.5), for $D \supsetneq C$, $(\hat{\boldsymbol{\beta}}^T(\lambda, D))_{D\setminus C} = \mathbf{0}$ and $(\hat{\boldsymbol{\beta}}^T(\lambda, D))_C = \hat{\boldsymbol{\beta}}^T(\lambda, C)$ if and only if

$$\max_{v \in D\setminus C} \tfrac{1}{n}\|\{\mathbf{Y}^T - \boldsymbol{\Pi}^T(\hat{\boldsymbol{\mu}}(\lambda, C), \hat{\boldsymbol{\beta}}(\lambda, C); \mathbf{X}_C)\}\mathbf{X}_v\|_2 \leq \lambda.$$

2.4.2 Structural sparsity

Although in our Backtracking algorithm, interaction terms are only added as candidates for selection when all their lower order interactions and main effects are active, this hierarchy in the selection of candidates does not necessarily follow through to the final model: one can have first-order interactions present in the final model without one or more of their main effects, for example. One way to enforce the hierarchy constraint in the final model is to use a base procedure which

obeys the constraint itself. Examples of such base procedures are provided by the Composite Absolute Penalties (CAP) family (Zhao et al., 2009).

Consider the linear regression setup with interactions. For simplicity we only describe Backtracking with first-order interactions. Let C be the candidate set and let $I = C \setminus C_1$ be the (first-order) interaction terms in C. In order to present the penalty, we borrow some notation from Combinatorics. Let $C_1^{(r)}$ denote the set of r-subsets of C_1. For $A \subseteq C_1^{(r)}$ and $r \geq 1$, define

$$\partial_l(A) = \{v \in C_1^{(r-1)} : v \subset u \text{ for some } u \in A\}$$
$$\partial_u(A) = \{v \in C_1^{(r+1)} : v \subset u \text{ for some } u \in A\}$$

These are known as the *lower shadow* and *upper shadow* respectively (Bollobás, 1986).

Our objective function Q is given by

$$Q(\mu, \boldsymbol{\beta}) = \tfrac{1}{2n}\|\mathbf{Y} - \mu\mathbf{1} - \mathbf{X}_C\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}_{C_1 \setminus \partial_l(I)}\|_1 + \lambda \sum_{v \in \partial_l(I)} \|\boldsymbol{\beta}_{\{v\} \cup (\partial_u(\{v\}) \cap I)}\|_\gamma + \lambda\|\boldsymbol{\beta}_I\|_1,$$

where $\gamma > 1$. For example, if $C = \{\{1\}, \ldots, \{4\}, \{1,2\}, \{2,3\}\}$, then omitting the factor of λ, the penalty terms in Q are

$$|\beta_{\{4\}}| + \|(\beta_{\{1\}}, \beta_{\{1,2\}})^T\|_\gamma + \|(\beta_{\{2\}}, \beta_{\{1,2\}}, \beta_{\{2,3\}})^T\|_\gamma + \|(\beta_{\{3\}}, \beta_{\{2,3\}})^T\|_\gamma + |\beta_{\{1,2\}}| + |\beta_{\{2,3\}}|.$$

The form of this penalty forces interactions to enter the active set only after or with their corresponding main effects.

The KKT conditions for this optimisation take a more complicated form than those for the Lasso. Nevertheless, checking they hold for a trial solution is an easier task than computing a solution.

2.4.3 Nonlinear models

If a high-dimensional additive modelling method (Meier et al., 2009; Ravikumar et al., 2009) is used as the base procedure, it is possible to fit nonlinear models with interactions. Here each variable is a collection of basis functions, and to add an interaction between variables, one adds the tensor product of the two collections of basis functions, penalizing the new interaction basis functions appropriately. Structural sparsity approaches can also be used here. The VANISH method of Radchenko and James (2010) uses a CAP-type penalty in nonlinear regression, and this can be used as a base procedure in a similar way to that sketched above.

2.4.4 Introducing more candidates

In our description of the Backtracking algorithm, we only introduce an interaction term when *all* of its lower order interactions and main effects are active. Another possibility, in the spirit of MARS (Friedman, 1991), is to add interaction terms when *any* of their lower order interactions or main effects are active. As at the k^{th} step of Backtracking, there will be roughly kp extra candidates, an approach that can enforce the hierarchical constraint may be necessary to allow main effects to be selected from amongst the more numerous interaction candidates. The key point to note is that if the algorithm is terminated after T steps, we are having to deal with roughly at most Tp variables rather than $O(p^2)$, the latter coming from including all first-order interactions.

Scenario	S_2^*
1	$\emptyset$
2	$\emptyset$
3	$\{\{1,2\},\{3,4\},\{5,6\}\}$
4	$\{\{1,2\},\{1,3\},\ldots,\{1,6\}\}$
5	$\mathcal{I}(\{\{1\},\{2\},\{3\}\}) \cup \mathcal{I}(\{\{4\},\{5\},\{6\}\})$

Table 2.1: Simulation settings.

2.5 Numerical results

2.5.1 Simulations

In this section, we report the results of five numerical studies designed to demonstrate the effectiveness of Backtracking with the Lasso and also highlight some of the drawbacks of using the Lasso with main effects only, when interactions are present. In each of the five scenarios, we generated 200 design matrices with $n = 250$ observations and $p = 1000$ covariates. The rows of the design matrices were sampled independently from $N_p(0, \Sigma)$ distributions. The covariance matrix Σ was chosen to be the identity in all scenarios except scenario 2, where

$$\Sigma_{ij} = 0.75^{-||i-j|-p/2|+p/2}.$$

Thus in this case, the correlation between the components decays exponentially with the distance between them in $\mathbb{Z}/p\mathbb{Z}$.

We created the responses according to the linear model with interactions and set the intercept to 0:

$$\mathbf{Y} = \mathbf{X}_{S^*}\boldsymbol{\beta}^*_{S^*} + \boldsymbol{\varepsilon}, \quad \varepsilon_i \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2). \tag{2.6}$$

The error variance σ^2 was chosen to achieve a signal-to-noise ratio (SNR) of either 2 or 3. We define SNR here by

$$\text{SNR}^2 = \frac{\mathbb{E}\|\mathbf{X}_{S^*}\boldsymbol{\beta}^*_{S^*}\|_2^2}{\mathbb{E}\|\boldsymbol{\varepsilon}\|_2^2}.$$

The set of main effects in S^*, S_1^*, was $\{\{1\},\ldots,\{10\}\}$. The subset of variables involved in interactions was $I_1^* := \{\{1\},\ldots,\{6\}\}$. The set of first-order interactions in S^* chosen in the different scenarios, S_2^*, is displayed in Table 2.1, and we took $S^* = S_1^* \cup S_2^*$ so S^* contained no higher order interactions. In each simulation run, $\boldsymbol{\beta}^*_{S_1^*}$ was fixed and given by

$$(2, -1.5, 1.25, -1, 1, -1, 1, 1, 1, 1)^T.$$

Each component of $\boldsymbol{\beta}^*_{S_2^*}$ was chosen to be $\sqrt{\|\boldsymbol{\beta}^*_{S_1^*}\|_2^2 / |S_1^*|}$. Thus the squared magnitude of the interactions was equal to average of the squared magnitudes of the main effects.

In all of the scenarios, we applied three methods: the Lasso using only the main effects; iterated Lasso fits; and the Lasso with Backtracking. Note that due to the size of p in these examples, the methods for finding interactions in lower-dimensional data discussed in Section 2.1, are computationally impractical here.

For the iterated Lasso fits, we repeated the following process. Given a design matrix, first fit the Lasso. Then apply 5-fold cross-validation to give a λ value and associated active set. Finally add all interactions between variables in this active set to the design matrix, ready for the next iteration.

For computational feasibility, the procedure was terminated when the number of variables in the design matrix exceeded $p + 250 \times 249/2$. As with Backtracking, this method yields a collection of solution paths, each one associated with a different number of iterations of the above process.

Additionally, in scenarios 3–5, we applied the Lasso with all main effects and only the true interactions. This theoretical Oracle approach provided a gold standard against which to test the performance of Backtracking. For all of the procedures mentioned, we used grids of 100 λ values. The LARS–OLS hybrid and cross-validation with squared error loss were used to give the final estimator. For our cross-validation procedure we randomly selected 5 folds each time but repeated this a total of 5 times to reduce the variance of the cross-validation scores. Thus for each λ value we obtained an estimate of the expected prediction error that was an average over the observed prediction errors on 25 (overlapping) validation sets of size $n/5 = 50$. Note that for both Backtracking and the iterated Lasso, this form of cross-validation chose not just a λ value but also a path rank. When using Backtracking, the size of the active set was restricted to 50 and the size of C_k to $p + 50 \times 49/2 = 1225$ (see Section 2.3.1.1). Our restricted minimisers of the cross-validation scores were always very far from these boundaries so it is likely they coincided with the global minimisers.

In scenarios 1 and 2, the results of all the methods were almost indistinguishable. Even in Scenario 2, where high correlations made estimation very challenging, neither Backtracking nor the iterated Lasso fits picked up any false interactions, though all the methods struggled to identify the important main effects.

The results of scenarios 3–5, where the signal contains interactions, are given in Table 2.2. For each scenario, method and SNR level, we report 5 statistics. 'L_2-sq' is the expected squared distance of the signal $\mathbf{f}^*$ and our prediction functions $\hat{\mathbf{f}}$ based on training data $(\mathbf{Y}_{\text{train}}, \mathbf{X}_{\text{train}})$, evaluated at a random independent test observation $\mathbf{x}_{\text{new}}$:

$$\mathbb{E}_{\mathbf{x}_{\text{new}}, \mathbf{Y}_{\text{train}}, \mathbf{X}_{\text{train}}} (\mathbf{f}^*\{\mathbf{x}_{\text{new}}) - \hat{\mathbf{f}}(\mathbf{x}_{\text{new}}; \mathbf{Y}_{\text{train}}, \mathbf{X}_{\text{train}})\}^2.$$

'FP Main' and 'FP Inter' are the numbers of noise main effects and noise interaction terms respectively, incorrectly included in the final active set. 'FN Main' and 'FN Inter' are the numbers of true main effects and interaction terms respectively, incorrectly excluded from the final active set.

For all the statistics presented, lower numbers are to be preferred. However, the higher number of false selections incurred by both Backtracking and the Oracle procedure compared to using the main effects only or iterated Lasso fits, is due to the model selection criterion being the expected prediction error. It should not be taken as an indication that the latter procedures are performing better in these cases.

Backtracking performs best out of the three methods compared here. Note that under all of the settings, iterated Lasso fits incorrectly selects more interaction terms than Backtracking. We see that the more careful way in which Backtracking adds candidate interactions, helps here. Unsurprisingly, fitting the Lasso on just the main effects performs rather poorly in terms of predictive performance. However, it also fails to select important main effects; Backtracking and Iterates have much lower main effect false negatives.

2.5.2 Real data analyses

In this section, we look at the performance of Backtracking using two base procedures, the Lasso for the linear model and the Lasso for multinomial regression, on a regression and a classification

Scenario	Statistic	SNR = 2				SNR = 3			
		Main	Iterate	Back-tracking	Oracle	Main	Iterate	Back-tracking	Oracle
3	L_2-sq	6.946	1.401	1.210	0.825	5.671	0.274	0.272	0.184
	FP Main	3.182	2.434	2.889	3.192	1.914	0.652	0.727	0.788
	FN Main	1.258	0.379	0.237	0.136	0.515	0.045	0.035	0.005
	FP Inter	0.000	0.929	0.449	0.000	0.000	0.273	0.121	0.000
	FN Inter	3.000	0.182	0.141	0.005	3.000	0.030	0.035	0.000
4	L_2-sq	12.046	3.255	2.723	1.682	10.444	0.632	0.406	0.305
	FP Main	2.217	3.884	5.343	7.051	2.581	1.798	2.081	2.212
	FN Main	3.121	0.899	0.606	0.258	1.768	0.111	0.040	0.000
	FP Inter	0.000	2.500	0.768	0.000	0.000	1.768	0.283	0.000
	FN Inter	5.000	0.662	0.510	0.081	5.000	0.076	0.030	0.000
5	L_2-sq	14.122	5.081	4.521	2.144	12.841	1.556	1.170	0.436
	FP Main	3.071	4.747	5.869	8.571	3.429	3.005	3.227	3.768
	FN Main	3.202	1.258	0.980	0.333	2.348	0.253	0.192	0.015
	FP Inter	0.000	3.278	0.869	0.000	0.000	3.051	0.551	0.000
	FN Inter	6.000	1.343	1.227	0.136	6.000	0.394	0.303	0.000

Table 2.2: Simulation results.

data set. As competing methods, we consider simply using the base procedures ('Main'), iterated Lasso fits ('Iterated'), Random Forests (Breiman, 2001), hierNet (Bien et al., 2013) and MARS (Friedman, 1991) (implemented using Hastie et al. (2013)). Note that we do not view the latter two methods as competitors of Backtracking, as they are designed for use on lower dimensional datasets than Backtracking is capable of handling. However, it is still interesting to see how the methods perform on data of dimension that is perhaps approaching the upper end of what is easily manageable for methods such as hierNet and MARS, but at the lower end of what one might used Backtracking on.

Below we describe the datasets used which are both from the UCI machine learning repository (Asuncion and Newman, 2007).

2.5.2.1 Communities and Crime

This dataset available at `http://archive.ics.uci.edu/ml/datasets/Communities+and+Crime+Unnormalized` contains crime statistics for the year 1995 obtained from FBI data, and national census data from 1990, for various towns and communities around the USA. We took violent crimes per capita as our response: violent crime being defined as murder, rape, robbery, or assault. The data set contains two different estimates of the populations of the communities: those from the 1990 census and those from the FBI database in 1995. The latter was used to calculate our desired response using the number of cases of violent crimes. However, in several cases, the FBI population data seemed suspect and we discarded all observations where the maximum of the ratios of the two available population estimates differed by more than 1.25. In addition, we removed all observations that were missing a response and several variables for which the majority of values were missing. This resulted in a dataset with $n = 1903$ observations and $p = 101$ covariates.

2.5.2.2 ISOLET

This data set consists of $p = 617$ features based on the speech waveforms generated from utterances of each letter of the English alphabet. The task is to learn a classifier which can determine the letter spoken based on these features. The dataset is available from `http://archive.ics.uci.edu/`

`ml/datasets/ISOLET`; see Fanty and Cole (1991) for more background on the data. We consider classification on the notoriously challenging E-set consisting of the letters 'B', 'C', 'D', 'E', 'G', 'P', 'T', 'V' and 'Z' (pronounced 'zee'). As there were 150 subjects and each spoke each letter twice, we have $n = 2700$ observations spread equally among 9 classes. The dimension of this data is such that MARS and hierNet could not be applied.

2.5.3 Methods and results

For the Communities and crime data set, we used the Lasso for the linear model as the base regression procedure for Backtracking and Iterates. Since the per capita violent crime response was always non-negative, the positive part of the fitted values was taken. For Main, Backtracking, Iterates and hierNet, we employed 5-fold cross-validation with squared error loss to select tuning parameters. For MARS we used the default settings for pruning the final fits using generalised cross-validation. With Random Forests, we used the default settings on both data sets. For the classification example, penalised multinomial regression was used (see Section 2.4.1) as the base procedure for Backtracking and Iterates, and the deviance was used as the loss function for 5-fold cross-validation. In all of the methods except Random Forests, we only included first-order interactions. When using Backtracking, we also restricted the size of C_k to $p+50\times 49/2 = p+1225$.

To evaluate the procedures, we randomly selected 2/3 for training and the remaining 1/3 was used for testing. This was repeated 200 times for each of the data sets. Note that we have specifically chosen data sets with n large as well as p large. This is to ensure that comparisons between the performances of the methods can be made with more accuracy. For the regression example, out-of-sample squared prediction error was used as a measure of error; for the classification example, we used out-of-sample misclassification error with 0–1 loss. The results are given in Table 2.3.

Random Forests has the lowest prediction error on the regression dataset, with Backtracking not far behind, whilst Backtracking wins in the classification task, and in fact achieves strictly lower misclassification error than all the other methods on 90% of all test samples. Note that a direct comparison with Random Forests is perhaps unfair, as the latter is a black-box procedure whereas Backtracking is aiming for a more interpretable model.

MARS performs very poorly indeed on the regression dataset. The enormous prediction error is caused by the fact that whenever observations corresponding to either New York or Los Angeles were in the test set, MARS predicted their responses to be far larger than they were. However, even with these observations removed, the instability of MARS meant that it was unable to give much better predictions than an intercept-only model.

HierNet performs well on this dataset, though it is worth noting that we had to scale the interactions to have the same ℓ_2-norm as the main effects to get such good results (the default scaling produced error rates worse than that of an intercept-only model). Backtracking does better here. One reason for this is that the because the main effects are reasonably strong in this case, a low amount of penalisation works well. However, because with hierNet, the penalty on the interactions is coupled with the penalty on the main effects, the final model tended to include close to two hundred interaction terms.

The way that Backtracking creates several solution paths with varying numbers of interaction terms means that it is possible to fit main effects and a few interactions using a low penalty without this low penalisation opening the door to many other interaction terms. The iterated

Method	Error	
	Communities and crime	ISOLET
Main	0.414 (6.5×10^{-3})	0.0641 (4.7×10^{-4})
Iterate	0.384 $(5.9 \times 10^{-3}$	0.0641 (4.7×10^{-4})
Backtracking	0.365 (3.7×10^{-3})	0.0563 (4.5×10^{-4})
Random Forest	0.356 (2.4×10^{-3})	0.0837 (6.0×10^{-4})
hierNet	0.373 (4.7×10^{-3})	-
MARS	5580.586 (3.1×10^{3})	-

Table 2.3: Real data analyses results. Average error rates over 200 training–testing splits are given, with standard deviations of the results divided by $\sqrt{200}$ in parentheses.

Lasso approach also has this advantage, but as the number of interactions are increased in discrete stages, it can miss a candidate set with the right number of interactions that may be picked up by the more continuous model building process used by Backtracking. This occurs in a rather extreme way with the ISOLET dataset where, since in the first stage of the iterated Lasso, cross-validation selected far too many variables (> 250), the second and subsequent steps could not be performed. This is why the results are identical to using the main effects alone.

2.6 Theoretical properties

Our goal in this section is to understand under what circumstances Backtracking with the Lasso can arrive at a set of candidates, $\tilde{C}^*$, that contains all of the true interactions, and only a few false interactions. On the event on which this occurs, we can then apply many of the existing results on the Lasso, to show that the solution path $\tilde{\boldsymbol{\beta}}(\lambda, \tilde{C}^*)$ has certain properties. As an example, we give sufficient conditions for the existence of a λ^* such that $\{v : \tilde{\beta}_v(\lambda^*, \tilde{C}^*) \neq 0\}$ equals the true set of variables.

We work with the normal linear model with interactions,

$$\mathbf{Y} = \mu^* \mathbf{1} + \mathbf{X}_{S^*} \boldsymbol{\beta}_{S^*} + \boldsymbol{\varepsilon}, \tag{2.7}$$

where $\varepsilon_i \overset{\text{i.i.d.}}{\sim} N(0, \sigma^2)$, and to ensure identifiability, $\mathbf{X}_{S^*}$ has full column rank. We will assume that $S^* = S_1^* \cup S_2^*$, where S_1^* and S_2^* are main effects and two-way interactions respectively. Further we will assume that the set of interacting main effects,

$$I_1^* := \{v : |v| = 1, v \subseteq u, \text{ some } u \in S_2^*\},$$

satisfies $I_1^* \subseteq S_1^*$.

In order for Backtracking not to add any interactions involving noise variables, to begin with, one pair of interacting signal variables must enter the solution path before any noise variables. Other interacting signal variables need only become active after the interaction between this first pair has become active. Thus we need that there is some ordering of the interacting variables where each variable only requires interactions between those variables earlier in the order to be present before it can become active. Variables early on in the order must have the ability to be selected when there is serious model misspecification as few interaction terms will be available for selection. Variables later in the order only need to have the ability to be selected when the model is approximately correct.

Note that a signal variable having a coefficient large in absolute value does not necessarily ensure that it becomes active before any noise variable. Indeed, in our example in Section 2.2, variable 5 did not enter the solution path at all when only main effects were present, but had the largest coefficient. If we write $\mathbf{f}^*$ for $\mathbf{X}_{S^*}\boldsymbol{\beta}_{S^*}$, and for a set S such that $\mathbf{X}_S$ has full column rank, we define

$$\boldsymbol{\beta}^S := (\mathbf{X}_S^T\mathbf{X}_S)^{-1}\mathbf{X}_S^T\mathbf{f}^*,$$

intuitively what should matter are the sizes of the appropriate coefficients of $\boldsymbol{\beta}^S$ for suitable choices of S. In the next section, we give a sufficient condition based on $\boldsymbol{\beta}^S$ for a variable $v \in S$ to enter the solution path before any variable outside S.

2.6.1 The entry condition

Let $\mathbf{P}^S = \mathbf{X}_S(\mathbf{X}_S^T\mathbf{X}_S)^{-1}\mathbf{X}_S^T$ denote orthogonal projection on to the space spanned by the columns of $\mathbf{X}_S$. Further, for any two candidate sets $S, M \subseteq \mathcal{P}(\{1,\ldots,p\})$, define

$$\boldsymbol{\Sigma}_{S,M} = \tfrac{1}{n}\mathbf{X}_S^T\mathbf{X}_M.$$

Now given a set of candidates, C, let $v \in S \subset C$ and write $M = C \setminus S$. For $\eta > 0$, we shall say that the $\mathrm{Ent}(v, S, C; \eta)$ condition holds if, $\mathbf{X}_S$ has full column rank, and the following holds,

$$\sup_{\boldsymbol{\tau}_S \in \mathbb{R}^{|S|}:\|\boldsymbol{\tau}_S\|_\infty \le 1} \|\boldsymbol{\Sigma}_{M,S}\boldsymbol{\Sigma}_{S,S}^{-1}\boldsymbol{\tau}_S\|_\infty < 1, \tag{2.8}$$

$$|\beta_v^S| > \max_{u \in M}\left\{\frac{\frac{1}{n}\left|\mathbf{X}_u^T(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^*\right| + 2\eta}{1 - \|\boldsymbol{\Sigma}_{S,S}^{-1}\boldsymbol{\Sigma}_{S,\{u\}}\|_1} + \eta\right\}\|(\boldsymbol{\Sigma}_{S,S}^{-1})_v\|_1. \tag{2.9}$$

In Lemma 14 given in the Appendix of this chapter, we show that this condition is sufficient for variable v to enter the active set before any variable in M, when the set of candidates is C and $\|\mathbf{X}_C^T\boldsymbol{\varepsilon}\|_\infty \le \eta$. In addition, we show that v will remain in the active set at least until some variable from M enters the active set.

The condition (2.8) is closely related to irrepresentable conditions (see Meinshausen and Bühlmann (2006), Zhao and Yu (2006), Zou (2006), Bühlmann and van de Geer (2011b), Wainwright (2009), for example), which are used for proving variable selection consistency of the Lasso. Indeed, when S is the set of true nonzero coefficients, it can be shown that the condition,

$$\|\boldsymbol{\Sigma}_{M,S}\boldsymbol{\Sigma}_{S,S}^{-1}\mathrm{sgn}(\boldsymbol{\beta}_S^*)\|_\infty \le 1, \tag{2.10}$$

is essentially necessary for variable selection consistency of the Lasso. If we require this to hold for all possible sign vectors $\mathrm{sgn}(\boldsymbol{\beta}_S^*)$, we arrive at (2.8).

The second part of the entry condition (2.9) asserts that coefficient v of the regression of $\mathbf{f}^*$ on $\mathbf{X}_S$ must exceed a certain quantity that we now examine in more detail. The $\frac{1}{n}\mathbf{X}_u^T(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^*$ term is the sample covariance between $\mathbf{X}_u$, which is one of the columns of $\mathbf{X}_M$, and the residual from regressing $\mathbf{f}^*$ on $\mathbf{X}_S$. Note that the more of S^* that S contains, the closer this will be to 0.

To understand the $\|(\boldsymbol{\Sigma}_{S,S}^{-1})_v\|_1$ term, without loss of generality take v as $\{1\}$ and write $\mathbf{b} = \boldsymbol{\Sigma}_{S\setminus\{v\},\{v\}}$ and $\mathbf{D} = \boldsymbol{\Sigma}_{S\setminus\{v\},S\setminus\{v\}}$. For any square matrix $\boldsymbol{\Sigma}$, let $c_{\min}(\boldsymbol{\Sigma})$ denote its minimal

eigenvalue. Using the formula for the inverse of a block matrix and writing s for $|S|$, we have

$$\begin{aligned}\|(\mathbf{\Sigma}_{S,S}^{-1})_v\|_1 &= \left\| \begin{pmatrix} 1 + \mathbf{b}^T(\mathbf{D} - \mathbf{b}\mathbf{b}^T)^{-1}\mathbf{b} \\ -(\mathbf{D} - \mathbf{b}\mathbf{b}^T)^{-1}\mathbf{b} \end{pmatrix} \right\|_1 \\ &\leq 1 + \frac{\|\mathbf{b}\|_2^2 + \sqrt{s-1}\|\mathbf{b}\|_2}{c_{\min}(\mathbf{\Sigma}_{S,S})}.\end{aligned}$$

In the final line we have used the Cauchy–Schwarz inequality and the fact that if $\mathbf{w}^*$ is a unit eigenvector of $\mathbf{D} - \mathbf{b}\mathbf{b}^T$ with minimal eigenvalue, then

$$c_{\min}(\mathbf{D} - \mathbf{b}\mathbf{b}^T) = \left\| \mathbf{\Sigma}_{S,S} \begin{pmatrix} -\mathbf{b}^T\mathbf{w}^* \\ \mathbf{w}^* \end{pmatrix} \right\|_2 \geq c_{\min}(\mathbf{\Sigma}_{S,S})\sqrt{1 + |\mathbf{b}^T\mathbf{w}^*|^2} \geq c_{\min}(\mathbf{\Sigma}_{S,S}).$$

Thus when variable v is not too correlated with the other variables in S, and so $\|\mathbf{b}\|_2$ is small, $\|(\mathbf{\Sigma}_{S,S}^{-1})_v\|_1$ will not be too large. Even when this is not the case, we still have the bound

$$\|(\mathbf{\Sigma}_{S,S}^{-1})_v\|_1 \leq \frac{\sqrt{s}}{c_{\min}(\mathbf{\Sigma}_{S,S})}.$$

Turning now to the denominator, $\|\mathbf{\Sigma}_{S,S}^{-1}\mathbf{\Sigma}_{S,\{u\}}\|_1$ is the ℓ_1-norm of the coefficients from regression of $\mathbf{X}_u$ on $\mathbf{X}_S$, and the maximum of this quantity over $u \in M$ gives the left-hand side of (2.8). Thus when u is highly correlated with many of the variables in S, $\|\mathbf{\Sigma}_{S,S}^{-1}\mathbf{\Sigma}_{S,\{u\}}\|_1$ will be large. On the other hand, in this case one would expect $\|(\mathbf{I} - \mathbf{P}^S)\mathbf{X}_u\|_2$ to be small, and so to some extent the numerator and denominator compensate for each other.

2.6.2 Statement of results

Without loss of generality assume $I_1^* = \{\{1\}, \ldots, \{|I_1^*|\}\}$. Our formal assumption corresponding to the discussion at the beginning of Section 2.6 is the following.

> **The entry order condition.** There is some $I_1^* \subseteq \tilde{S}_1^* \subseteq S_1^*$, and some ordering of the variables in I_1^*, which without loss of generality we take to simply be $1, \ldots, |I_1^*|$, such that for each $\{j\} \in I_1^*$, we have,
>
> $$\begin{gathered}\text{For all } A : \mathcal{I}(\{\{1\}, \ldots, \{j-1\}\}) \subseteq A \subseteq \mathcal{I}(\tilde{S}_1^*) \\ \text{Ent}(\{j\}, \tilde{S}_1^* \cup (A \cap S_2^*), C_1 \cup A; \eta) \text{ holds.}\end{gathered}$$
>
> where
>
> $$\eta = \eta(t; n, p, |\tilde{S}_1^*|, \sigma) = \sigma\sqrt{\frac{t^2 + 2\log(p + \frac{1}{2}|\tilde{S}_1^*|^2)}{n}}.$$

First we discuss the implications for variable $\{1\}$. Let $\tilde{S}^* = \tilde{S}_1^* \cup S_2^*$ The condition ensures that whenever the candidate set is enlarged from C_1 to also include any subset of $\mathcal{I}(\tilde{S}_1^*)$, variable $\{1\}$ enters the active set before any variable outside $\tilde{S}^*$, and moreover, it remains in the active set at least until a variable outside $\tilde{S}^*$ enters.

For $j > 2$, we see that the enlarged candidate sets for which we require the entry conditions to hold, are fewer in number. Variable $\{|I_1^*|\}$ only requires the entry condition to hold for candidate

sets that at least include $\mathcal{I}(\{\{1\},\ldots,\{|I_1^*|-1\}\})$ and thus include almost all of S^*. What this means is that we require some 'strong' interacting variables, for which when $\mathbf{f}^*$ is regressed onto a variety of sets of variables containing them (some of which contain only a few of the true interaction variables), always have large coefficients. Given the existence of such strong variables, other interacting variables need only have large coefficients when $\mathbf{f}^*$ is regressed onto sets containing them that also include many true interaction terms. Note that the equivalent result for the success of the strategy that simply adds interactions between selected main effects would essentially require all main effect involved in interactions to satisfy the conditions imposed on the variables $\{1\}$ and $\{2\}$ here. Going back to the example in Section 2.2, variable 5 has $|\beta^S_{\{5\}}| \approx 0$ for all $S \subseteq \{\{1\},\ldots,\{6\}\}$, but $|\beta^S_{\{5\}}| > 0$ once $\{1,2\} \in S$ or $\{3,4\} \in S$.

The reason we use the sets $\tilde{S}_1^*$ and $\tilde{S}^*$ rather than their larger counterparts, S_1^* and S^*, is that there may be some very weak signals in $S_1^* \setminus I_1^*$. We do not want to require that the interacting variables remain in the active set all the way until these weak variables are selected, as the entry conditions would dictate.

We are now in a position to state our main theorem. Although the Backtracking algorithm was presented for a base path algorithm that computed solutions at only discrete values, for the following theorem, we need to imagine an idealised algorithm which computes the entire path of solutions. Explicitly, we require that our algorithm outputs a collection of paths

$$\{\tilde{\boldsymbol{\beta}}(\lambda, C_k) : \lambda \in [0,\infty], 1 \le k \le T\}$$

for which there exists a λ_k^{start} sequence with $\lambda_1^{\text{start}} = \infty$, that satisfies $\tilde{\boldsymbol{\beta}}(\lambda, C_k) = \hat{\boldsymbol{\beta}}(\lambda, C_k)$ for all $\lambda \le \lambda_k^{\text{start}}$, and for $2 \le k \le T$,

$$\mathcal{A}(\tilde{\boldsymbol{\beta}}(\lambda_k^{\text{start}}, C_k)) = \mathcal{A}(\tilde{\boldsymbol{\beta}}(\lambda_k^{\text{start}}, C_{k-1})).$$

In addition, we will assume that we only allow first-order interactions in the Backtracking algorithm.

Theorem 12. *Assume the entry order condition holds and let $\tilde{C}^* = C_1 \cup \mathcal{I}(\tilde{S}_1^*)$. With probability at least $1 - \exp(-t^2/2)$, there exists a k^* such that $\tilde{C}^* \supseteq C_{k^*} \supseteq S^*$.*

Theorem 12 gives sufficient conditions for Backtracking to produce a set of candidates that includes S^*, but no interactions among variables in $C_1 \setminus S^*$. Once we have such a set of candidates, we are essentially in the familiar 'Lasso with the linear model world', and we do not need to worry about interactions. The one caveat is that the path $\tilde{\boldsymbol{\beta}}(\cdot, C_{k*})$ need only coincide with $\hat{\boldsymbol{\beta}}(\cdot, C_{k^*})$ after $\lambda_{k^*}^{\text{start}}$. If this subtlety is taken into account, many of the theorems concerning the Lasso with the linear model can be applied. As an example, we give the following corollary.

Corollary 13. *Assume the entry order condition holds. Writing $N = \tilde{C}^* \setminus S^*$, further assume*

$$\|\boldsymbol{\Sigma}_{N,S^*}\boldsymbol{\Sigma}^{-1}_{S^*,S^*}\text{sgn}(\boldsymbol{\beta}^*_{S^*})\|_\infty < 1;$$

and that for all $v \in S^*$,

$$|\beta_v^*| > \frac{\eta\left|\text{sgn}(\boldsymbol{\beta}^*_{S^*})^T(\boldsymbol{\Sigma}^{-1}_{S^*,S^*})_v\right|}{1 - \|\boldsymbol{\Sigma}_{N,S^*}\boldsymbol{\Sigma}^{-1}_{S^*,S^*}\text{sgn}(\boldsymbol{\beta}^*_{S^*})\|_\infty} + \xi,$$

where

$$\xi = \xi(t; n, |S^*|, \sigma, c_{\min}(\mathbf{\Sigma}_{S^*,S^*})) = \sigma\sqrt{\frac{t^2 + 2\log(|S^*|)}{nc_{\min}(\mathbf{\Sigma}_{S^*,S^*})}}.$$

Then with probability at least $1 - 3\exp(-t^2/2)$*, there exist* k^* *and* λ^* *such that*

$$\mathcal{A}(\tilde{\boldsymbol{\beta}}(\lambda^*, C_{k^*})) = S^*.$$

Note that if we were to simply apply the Lasso to the set of candidates $C^{\text{all}} := C_1 \cup \mathcal{I}(C_1)$ (i.e. all possible main effects and their first-order interactions), we would require an irrepresentable condition of the form

$$\|\mathbf{\Sigma}_{N^{\text{all}},S^*}\mathbf{\Sigma}_{S^*,S^*}^{-1}\text{sgn}(\boldsymbol{\beta}^*_{S^*})\|_\infty < 1,$$

where $N^{\text{all}} = C^{\text{all}} \setminus S^*$. Thus we would need $O(p^2)$ inequalities to hold, rather than our $O(p)$. Of course, we had to introduce many additional assumptions to reach this stage and no set of assumptions is uniformly stronger or weaker than the other. However, our proposed method is computationally feasible.

2.7 Discussion

While several methods now exist for fitting interactions in moderate-dimensional situations where $p < 1000$, the problem of fitting interactions when the data is of truly high dimension has received comparatively little attention.

Typically, the search for interactions must be restricted by first fitting a model using only main effects, and then including interactions between those selected main effects, as well as the original main effects, as candidates in a final fit. This approach has the drawbacks that important main effects may not be selected in the initial stage as they require certain interactions to be present in order for them to be useful for prediction; and the initial model may contain too many main effects when, without the relevant interactions, the model selection procedure cannot find a good sparse approximation to the true model.

The Backtracking method proposed in this chapter allows interactions to be added in a more natural gradual fashion, so there is a better chance of having a model which contains the right interactions. The method is computationally efficient, and our numerical results demonstrate its effectiveness for both variable selection and prediction.

From a theoretical point of view we have shown that when used with the Lasso, rather than requiring all main effects involved in interactions to be highly correlated with the signal, Backtracking only needs there to exist some ordering of these variables where those early on in the order are important for predicting the response by themselves. Variables later in the order only need to be helpful for predicting the response when interactions between variables early on in the order are present.

Though here, we have largely focussed on Backtracking used with the Lasso, the method is very general and can be used with many procedures that involve sparsity-inducing penalty functions. These methods tend to be some of the most useful for dealing with high-dimensional data, as they can produce stable, interpretable models. Combined with Backtracking, the methods become much more flexible, and it would be very interesting to explore to what extent using non-linear base procedures could yield interpretable models with predictive power comparable to black-box

procedures such as Random Forests (Breiman, 2001). In addition, we believe integrating Backtracking with some of the penalty-based methods for fitting interactions to moderate-dimensional data, will prove to be a fruitful direction for future research.

2.8 Appendix

In this section, after presenting a lemma on the entry condition (Section 2.6.1), we prove Theorem 12 and Corollary 13. The proofs of Lemma 14 below, and Corollary 13 use many ideas from Wainwright (2009) and Bühlmann and van de Geer (2011b).

Lemma 14. *Let $S \subseteq C$ be such that X_S has full column rank and let $M = C \setminus S$. On the event*

$$\Omega_{C,\eta} := \{\tfrac{1}{n}\|\mathbf{X}_C^T\boldsymbol{\varepsilon}\|_\infty \leq \eta\},$$

the following hold:

(i) If

$$\lambda > \max_{u \in M}\left\{\frac{\frac{1}{n}|\mathbf{X}_u^T(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^*| + 2\eta}{1 - \|\boldsymbol{\Sigma}_{S,S}^{-1}\boldsymbol{\Sigma}_{S,\{u\}}\|_1}\right\}, \tag{2.11}$$

then the Lasso solution is unique and $\hat{\boldsymbol{\beta}}_M(\lambda, C) = \mathbf{0}$.

(ii) If λ is such that for some Lasso solution $\hat{\boldsymbol{\beta}}_M(\lambda, C) = \mathbf{0}$, and for $v \in S$,

$$|\beta_v^S| > \|(\boldsymbol{\Sigma}_{S,S}^{-1})_v\|_1(\lambda + \eta),$$

then for all Lasso solutions, $\hat{\beta}_v(\lambda, C) \neq 0$.

(iii) Let

$$\lambda^{\text{ent}} = \sup\{\lambda : \lambda \geq 0 \text{ and for some Lasso solution } \hat{\boldsymbol{\beta}}_M(\lambda, C) \neq \mathbf{0}\},$$

where we take $\sup\emptyset = 0$. If for $v \in S$,

$$|\beta_v^S| > \max_{u \in M}\left\{\frac{\frac{1}{n}|\mathbf{X}_u^T(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^*| + 2\eta}{1 - \|\boldsymbol{\Sigma}_{S,S}^{-1}\boldsymbol{\Sigma}_{S,\{u\}}\|_1} + \eta\right\}\|(\boldsymbol{\Sigma}_{S,S}^{-1})_v\|_1,$$

there exists a $\lambda > \lambda^{\text{ent}}$ such that the solution $\hat{\boldsymbol{\beta}}(\lambda, C)$ is unique, and for all $\lambda' \in (\lambda^{\text{ent}}, \lambda]$ and all Lasso solutions $\hat{\boldsymbol{\beta}}(\lambda', C)$, we have $\hat{\beta}_v(\lambda', C) \neq 0$.

Proof. We begin by proving (i). Suppressing the dependence of $\hat{\boldsymbol{\beta}}$ on λ and C, we can write the KKT conditions ((2.3), (2.4)) as

$$\frac{1}{n}\mathbf{X}_C^T(\mathbf{Y} - \mathbf{X}_C\hat{\boldsymbol{\beta}}) = \lambda\hat{\boldsymbol{\tau}},$$

where $\hat{\boldsymbol{\tau}}$ is an element of the subdifferential $\partial\|\hat{\boldsymbol{\beta}}\|_1$ and thus satisfies

$$\|\hat{\boldsymbol{\tau}}\|_\infty \leq 1, \tag{2.12}$$

$$\hat{\beta}_v \neq 0 \Rightarrow \hat{\tau}_v = \text{sgn}(\hat{\beta}_v). \tag{2.13}$$

By decomposing $\mathbf{Y}$ as $\mathbf{P}^S\mathbf{f}^* + (\mathbf{I}-\mathbf{P}^S)\mathbf{f}^* + \varepsilon$, $\mathbf{X}_C$ as $(\mathbf{X}_S\,\mathbf{X}_M)$, and noting that $\mathbf{X}_S^T(\mathbf{I}-\mathbf{P}^S) = \mathbf{0}$,

we can rewrite the KKT conditions in the following way:

$$\tfrac{1}{n}\mathbf{X}_S^T(\mathbf{P}^S\mathbf{f}^* - \mathbf{X}_S\hat{\boldsymbol{\beta}}_S) + \tfrac{1}{n}\mathbf{X}_S^T\boldsymbol{\varepsilon} - \boldsymbol{\Sigma}_{S,M}\hat{\boldsymbol{\beta}}_M = \lambda\hat{\boldsymbol{\tau}}_S, \tag{2.14}$$

$$\tfrac{1}{n}\mathbf{X}_M^T(\mathbf{P}^S\mathbf{f}^* - \mathbf{X}_S\hat{\boldsymbol{\beta}}_S) + \tfrac{1}{n}\mathbf{X}_M^T\{(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^* + \boldsymbol{\varepsilon}\} - \boldsymbol{\Sigma}_{M,M}\hat{\boldsymbol{\beta}}_M = \lambda\hat{\boldsymbol{\tau}}_M. \tag{2.15}$$

Now let $\breve{\boldsymbol{\beta}}_S$ be a solution to the restricted Lasso problem,

$$(\hat{\mu}, \breve{\boldsymbol{\beta}}_S) = \underset{\mu,\boldsymbol{\beta}_S}{\arg\min}\ \left\{\tfrac{1}{2n}\|\mathbf{Y} - \mu\mathbf{1} - \mathbf{X}_S\boldsymbol{\beta}_S\|^2 + \lambda\|\boldsymbol{\beta}_S\|_1\right\}.$$

The KKT conditions give that $\breve{\boldsymbol{\beta}}_S$ satisfies

$$\frac{1}{n}\mathbf{X}_S^T(\mathbf{Y} - \mathbf{X}_S\breve{\boldsymbol{\beta}}_S) = \lambda\breve{\boldsymbol{\tau}}_S, \tag{2.16}$$

where $\breve{\boldsymbol{\tau}}_S \in \partial\|\breve{\boldsymbol{\beta}}_S\|_1$. We now claim that

$$(\hat{\boldsymbol{\beta}}_S, \hat{\boldsymbol{\beta}}_M) = (\breve{\boldsymbol{\beta}}_S, \mathbf{0}) \tag{2.17}$$

$$(\hat{\boldsymbol{\tau}}_S, \hat{\boldsymbol{\tau}}_M) = \left(\breve{\boldsymbol{\tau}}_S,\ \boldsymbol{\Sigma}_{M,S}\boldsymbol{\Sigma}_{S,S}^{-1}(\breve{\boldsymbol{\tau}}_S - \tfrac{1}{n}\lambda^{-1}\mathbf{X}_S^T\boldsymbol{\varepsilon}) + \tfrac{1}{n}\lambda^{-1}\mathbf{X}_M^T\{(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^* + \boldsymbol{\varepsilon}\}\right) \tag{2.18}$$

is the unique solution to (2.14), (2.15), (2.12) and (2.13). Indeed, as $\breve{\boldsymbol{\beta}}_S$ solves the reduced Lasso problem, we must have that (2.14) and (2.13) are satisfied. Multiplying (2.14) by $\mathbf{X}_S\boldsymbol{\Sigma}_{S,S}^{-1}$, setting $\hat{\boldsymbol{\beta}}_M = \mathbf{0}$ and rearranging gives us that

$$\mathbf{P}^S\mathbf{f}^* - \mathbf{X}_S\hat{\boldsymbol{\beta}}_S = X_S\boldsymbol{\Sigma}_{S,S}^{-1}(\lambda\hat{\boldsymbol{\tau}}_S - \tfrac{1}{n}\mathbf{X}_S^T\boldsymbol{\varepsilon}), \tag{2.19}$$

and substituting this into (2.15) shows that our choice of $\hat{\boldsymbol{\tau}}_M$ satisfies (2.15). It remains to check that we have $\|\hat{\boldsymbol{\tau}}_M\|_\infty \leq 1$. In fact, we shall show that $\|\hat{\boldsymbol{\tau}}_M\|_\infty < 1$. Since we are on Ω_C and $\|\breve{\boldsymbol{\tau}}_S\|_\infty \leq 1$, for $u \in M$ we have

$$\begin{aligned}
\lambda|\hat{\boldsymbol{\tau}}_u| &\leq \|\boldsymbol{\Sigma}_{S,S}^{-1}\boldsymbol{\Sigma}_{S,\{u\}}\|_1\left(\lambda\|\breve{\boldsymbol{\tau}}_S\|_\infty + \|\tfrac{1}{n}X_S^T\boldsymbol{\varepsilon}\|_\infty\right) + \tfrac{1}{n}\left|\mathbf{X}_u{}^T(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^*\right| + \tfrac{1}{n}\left|\mathbf{X}_u{}^T\boldsymbol{\varepsilon}\right| \\
&< \lambda\|\boldsymbol{\Sigma}_{S,S}^{-1}\boldsymbol{\Sigma}_{S,\{u\}}\|_1 + \tfrac{1}{n}\left|\mathbf{X}_u{}^T(\mathbf{I}-\mathbf{P}^S)\mathbf{f}^*\right| + 2\eta \\
&< \lambda,
\end{aligned}$$

where the final inequality follows from (2.11). We have shown that there exists a solution, $\hat{\boldsymbol{\beta}}$, to the Lasso optimisation problem with $\hat{\boldsymbol{\beta}}_M = 0$. The uniqueness of this solution follows from noting that $\|\hat{\boldsymbol{\tau}}_M\|_\infty < 1$, X_S has full column rank and appealing to Lemma 1 of Wainwright (2009).

For (ii), note that from (2.14), provided $\hat{\boldsymbol{\beta}}_M = 0$, we have that

$$\hat{\boldsymbol{\beta}}_S = \boldsymbol{\beta}^S - \boldsymbol{\Sigma}_{S,S}^{-1}(\lambda\hat{\boldsymbol{\tau}}_S - \tfrac{1}{n}\mathbf{X}_S^T\boldsymbol{\varepsilon}).$$

But by assumption

$$|\beta_v^S| > \|(\boldsymbol{\Sigma}_{S,S}^{-1})_v\|_1(\lambda+\eta) \geq \left|(\boldsymbol{\Sigma}_{S,S}^{-1})_v^T(\lambda\hat{\boldsymbol{\tau}}_S - \tfrac{1}{n}\mathbf{X}_S^T\boldsymbol{\varepsilon})\right|,$$

whence $\hat{\beta}_v \neq 0$.

(iii) follows easily from (i) and (ii). □

Proof of Theorem 12. In all that follows, we work on the event $\Omega_1 \cap \Omega_{\tilde{C}^*,\eta}$ where $\Omega_{\tilde{C}^*,\eta}$ is defined as in Lemma 14 and

$$\Omega_1^c = \{\mathbf{Y} = \mu\mathbf{1} + \mathbf{X}_A\boldsymbol{\beta}_A, \text{ some } \mu, \boldsymbol{\beta}, A \in \mathcal{P}(C_1) : |A| < |S^*|\}.$$

Clearly as $|S^*| < n$, $\mathbb{P}(\Omega_1) = 1$. Note that trivially,

$$\Omega_{\tilde{C}^*,\eta} = \bigcap_{A:A\subseteq\tilde{C}^*} \Omega_{A,\eta},$$

so Lemma 14 can be used in conjunction with the entry order condition to deduce that certain variables enter the active set before certain sets of noise variables. Using standard bounds for the tails of Gaussian random variables and the union bound, it is easy to show that $\mathbb{P}(\Omega_1 \cap \Omega_{\tilde{C}^*,\eta}) \geq 1 - \exp(-t^2/2)$.

Let $C_{k^{\max}}$ be the largest member of $\{C_1, \ldots, C_T\}$ satisfying $C_{k^{\max}} \subseteq \tilde{C}^*$. Such a $C_{k^{\max}}$ exists since $C_1 \subseteq \tilde{C}^*$.

Now suppose that for $k \leq k^{\max}$, $C_k \not\supseteq S^*$. We shall show that then $k+1 \leq T$ and $C_{k+1} \subseteq \tilde{C}^*$, thus showing that we may take $k^* = k^{\max}$. Take $j^{\max}$ such that

$$\mathcal{I}(\{\{1\}, \ldots, \{j^{\max} - 1\}\}) \subseteq C_k,$$

with $j^{\max}$ maximal. Since $\mathcal{I}(\{\{1\}\}) = \emptyset$, such a $j^{\max}$ exists. Let $A = C_k \setminus C_1$. Note that

$$\mathcal{I}(\{\{1\}, \ldots, \{j^{\max} - 1\}\}) \subseteq A \subseteq \tilde{C}^* \setminus C_1 = \mathcal{I}(\tilde{S}_1^*).$$

Thus by the entry order condition, we know that for all $j \leq j^{\max}$, the $\text{Ent}(\{j\}, C_k \setminus \tilde{S}^*, C_k, \eta)$ condition holds. Let

$$\lambda^{\text{ent}} = \sup\{\lambda : \lambda \geq 0 \text{ and for some Lasso solution } \hat{\boldsymbol{\beta}}_{C_k\setminus\tilde{S}^*}(\lambda, C_k) \neq 0\},$$

where we take the supremum to be 0 if the set is empty. By Lemma 14, part (iii), we know that for all $j \leq j^{\max}$, there exists a $\lambda_j > \lambda^{\text{ent}}$ such that $\{j\} \in \mathcal{A}(\hat{\boldsymbol{\beta}}(\lambda, C_k))$ for all $\lambda \in (\lambda^{\text{ent}}, \lambda_j]$, and moreover, we know that the Lasso solution at λ_j is unique. Note that as $\mathcal{A}(\hat{\boldsymbol{\beta}}(\lambda_j, C_k) \subsetneq S^*$, the fact that we are on Ω_1 means we do not have a perfect fit at λ_j, i.e. $\|Y - \mu\mathbf{1} - X_{C_k}\hat{\boldsymbol{\beta}}(\lambda_j, C_k)\|_2 > 0$. Let $\lambda^{\text{all}} = \min_j \lambda_j$. Then

$$\mathcal{A}(\hat{\boldsymbol{\beta}}(\lambda^{\text{all}}, C_k)) \supseteq \{\{1\}, \ldots, \{j^{\max}\}\} \quad \text{and}$$
$$\hat{\boldsymbol{\beta}}_{C_k\setminus S^*}(\lambda, C_k) = 0 \quad \text{for all } \lambda \geq \lambda^{\text{all}}$$

That is, λ^{all} is a point on the solution path at which variables $\{1\}, \ldots, \{j^{\max}\}$ are in the active set, and before which no variable from $C_k \setminus S^*$ is active.

Now it remains to understand what this means for the approximate solution paths, $\tilde{\boldsymbol{\beta}}$, computed by Backtracking. For the case $k = 1$, we have $\tilde{\boldsymbol{\beta}}(\cdot, C_1) = \hat{\boldsymbol{\beta}}(\cdot, C_1)$, and so we can conclude that $k + 1 = 2 \leq T$, and $C_2 \subseteq \tilde{C}^*$.

For the case $k > 1$, suppose first (for contradiction) that $\lambda_k^{\text{start}} \leq \lambda^{\text{all}}$. Note that

$$\mathcal{A}(\hat{\boldsymbol{\beta}}(\lambda_k^{\text{start}}, C_k)) = \mathcal{A}(\tilde{\boldsymbol{\beta}}(\lambda_k^{\text{start}}, C_{k-1})).$$

Now we must have that

$$\mathcal{A}(\tilde{\boldsymbol{\beta}}(\lambda_k^{\text{start}}, C_{k-1})) \subseteq \tilde{S}^*, \tag{2.20}$$

$$\mathcal{A}(\tilde{\boldsymbol{\beta}}(\lambda_k^{\text{start}}, C_{k-1})) \not\supseteq \{\{1\}, \ldots, \{j^{\max}\}\} \tag{2.21}$$

as otherwise, by the design of our Backtracking algorithm, either $C_k \not\subseteq \tilde{C}^*$ or $C_k \supseteq \mathcal{I}(\{\{1\}, \ldots, \{j^{\max}\}\})$. By Lemma 14, part (ii), we know that for each $j \leq j^{\max}$, if for some $\lambda \leq \lambda_j$, $\hat{\boldsymbol{\beta}}_{C_k \setminus \tilde{S}^*}(\lambda, C_k) = 0$, then $\hat{\boldsymbol{\beta}}_{\{j\}}(\lambda, C_k) \neq 0$. But since $\lambda_k^{\text{start}} \leq \lambda^{\text{all}}$, by (2.20) λ_k^{start} is such a λ, which then contradicts (2.21).

Thus $\lambda_k^{\text{start}} > \lambda^{\text{all}}$, so we can conclude that $k+1 \leq T$ and that $C_{k+1} \subseteq \tilde{C}^*$. □

Proof of Corollary 13. Let Ω_1 and $\Omega_{\tilde{C}^*,\eta}$ be defined as in Lemma 14. Also define the events

$$\Omega_2 = \{\tfrac{1}{n}\|\mathbf{X}_N^T(\mathbf{I} - \mathbf{P}^{S^*})\boldsymbol{\varepsilon}\|_\infty \leq \eta\},$$
$$\Omega_3 = \{\tfrac{1}{n}\|\boldsymbol{\Sigma}_{S^*,S^*}^{-1}\mathbf{X}_{S^*}^T\boldsymbol{\varepsilon}\|_\infty \leq \xi\}$$

In all that follows, we work on the event $\Omega_1 \cap \Omega_2 \cap \Omega_3 \cap \Omega_{\tilde{C}^*,\eta}$. As $\mathbf{I} - \mathbf{P}^{S^*}$ is a projection,

$$\mathbb{P}(\tfrac{1}{n}|\mathbf{X}_v{}^T(\mathbf{I} - \mathbf{P}^{S^*})\boldsymbol{\varepsilon}| \leq \eta) \geq \mathbb{P}(\tfrac{1}{n}|\mathbf{X}_v{}^T\boldsymbol{\varepsilon}| \leq \eta).$$

Further, $\frac{1}{n}\boldsymbol{\Sigma}_{S^*,S^*}^{-1}\mathbf{X}_{S^*}^T\boldsymbol{\varepsilon} \sim N_{|S^*|}(0, \frac{1}{n}\boldsymbol{\Sigma}^2\boldsymbol{\Sigma}_{S^*,S^*}^{-1})$. Thus

$$\mathbb{P}(\Omega_3) \geq |S^*|\mathbb{P}(|Z| \leq \xi)$$

where $Z \sim N(0, \boldsymbol{\Sigma}^2/(nc_{\min}(\boldsymbol{\Sigma}_{S^*,S^*})))$. Note that

$$\mathbb{P}(\Omega_1 \cap \Omega_2 \cap \Omega_3 \cap \Omega_{\tilde{C}^*,\eta}) \geq 1 - \mathbb{P}(\Omega_{\tilde{C}^*,\eta}^c) - \mathbb{P}(\Omega_2^c) - \mathbb{P}(\Omega_3^c).$$

Using this, it is straightforward to show that $\mathbb{P}(\Omega_1 \cap \Omega_2 \cap \Omega_3 \cap \Omega_{\tilde{C}^*,\eta}) \geq 1 - 3\exp(-t^2/2)$.

Since we are on $\Omega_1 \cap \Omega_{\tilde{C}^*,\eta}$, we can assume the existence of a k^* from Theorem 12. We now follow the proof of Lemma 14 taking $S = S^*$ and $M = C_{k^*} \setminus S^* \subseteq N$. The KKT conditions become

$$\boldsymbol{\Sigma}_{S^*,S^*}(\boldsymbol{\beta}_{S^*}^* - \hat{\boldsymbol{\beta}}_{S^*}) + \tfrac{1}{n}\mathbf{X}_{S^*}^T\boldsymbol{\varepsilon} - \boldsymbol{\Sigma}_{S^*,M}\hat{\boldsymbol{\beta}}_M = \lambda\hat{\boldsymbol{\tau}}_{S^*}, \tag{2.22}$$

$$\boldsymbol{\Sigma}_{M,S^*}(\boldsymbol{\beta}_{S^*}^* - \hat{\boldsymbol{\beta}}_{S^*}) + \tfrac{1}{n}\mathbf{X}_M^T\boldsymbol{\varepsilon} - \boldsymbol{\Sigma}_{M,M}\hat{\boldsymbol{\beta}}_M = \lambda\hat{\boldsymbol{\tau}}_M, \tag{2.23}$$

with $\hat{\boldsymbol{\tau}}$ also satisfying (2.12) and (2.13) as before. Now let λ be such that

$$\frac{\eta}{1 - \|\boldsymbol{\Sigma}_{M,S^*}\boldsymbol{\Sigma}_{S^*,S^*}^{-1}\text{sgn}(\boldsymbol{\beta}_{S^*}^*)\|_\infty} < \lambda < \min_{v \in S^*}\left\{\left|\text{sgn}(\boldsymbol{\beta}_{S^*}^*)^T(\boldsymbol{\Sigma}_{S^*,S^*}^{-1})_v\right|^{-1}(|\beta_v^*| - \xi)\right\}.$$

It is straightforward to check that

$$(\hat{\boldsymbol{\beta}}_{S^*}, \hat{\boldsymbol{\beta}}_M) = (\boldsymbol{\beta}_{S^*}^* - \lambda\boldsymbol{\Sigma}_{S^*,S^*}^{-1}\text{sgn}(\boldsymbol{\beta}_{S^*}^*) + \tfrac{1}{n}\boldsymbol{\Sigma}_{S^*,S^*}^{-1}\mathbf{X}_{S^*}^T\boldsymbol{\varepsilon},\, \mathbf{0})$$
$$(\hat{\boldsymbol{\tau}}_{S^*}, \hat{\boldsymbol{\tau}}_M) = \left(\text{sgn}(\boldsymbol{\beta}_{S^*}^*),\, \boldsymbol{\Sigma}_{M,S^*}\boldsymbol{\Sigma}_{S^*,S^*}^{-1}\text{sgn}(\boldsymbol{\beta}_{S^*}^*) + \tfrac{1}{n}\lambda^{-1}\mathbf{X}_M^T(\mathbf{I} - \mathbf{P}^{S^*})\boldsymbol{\varepsilon}\right)$$

is the unique solution to (2.22), (2.23), (2.12) and (2.13). The only danger is that we may have $\lambda > \lambda_{k^*}^{\text{start}}$. However, we know that $\hat{\boldsymbol{\beta}}_M(\lambda_{k^*}^{\text{start}}, C_{k^*}) = 0$. It is easy to check that in this case we

still have $\mathrm{sgn}(\hat{\boldsymbol{\beta}}_{S^*}(\lambda_{k^*}^{\mathrm{start}}, C_{k^*})) = \mathrm{sgn}(\boldsymbol{\beta}^*_{S^*})$, and thus we may take $\lambda^* = \min\{\lambda, \lambda_{k^*}^{\mathrm{start}}\}$. □

Chapter 3

Random Intersection Trees

3.1 Introduction

In this chapter we return to the problem of detecting interactions that was the subject of the previous chapter, but this time in the setting of classification with high-dimensional binary predictors. We suppose we have data that can be written in the form (Y_i, X_i) for observations $i = 1, \ldots, n$; Y_i is the class label and $X_i \subseteq \{1, \ldots, p\}$ is the set of active predictors for observations i (out of a total of p predictors). An important example of this type of problem is that of text classification, where then X_i is the set of frequently appearing words (in a suitable sense) for document i, and Y_i indicates whether the document belongs to a certain class. In this case, the dimension p can be of the order of several thousand or more. More generally, if data with continuous predictors are available, they can be converted to binary format by choosing various split-points, and then reporting whether or not each variable exceeds each of these thresholds.

Our aim here is to develop methodology that can discover important interaction terms in the data without requiring that any of their lower order interactions are also informative. This is in contrast to the previous chapter where we relied on all corresponding lower order interactions being present when higher order interactions were important. To state the problem under consideration here more precisely, we are interested in finding subsets $S \subseteq \{1, \ldots, p\}$ of all predictor variables that occur more often for observations in a class of interest than for other observations. We will use the terms 'leaf nodes', 'rules', 'patterns' and 'interactions' interchangeably to describe such subsets S. In general, when p is large, finding interactions of this sort without restricting our search as in the previous chapter, can be computationally infeasible. Here however, we hope to exploit the sparsity of the predictors to overcome these difficulties.

For simplicity, suppose there are only two classes, the set of labels being $\{0, 1\}$. The case with more than two classes can be dealt with using one-versus-one, or one-versus-all strategies. Given a pair of thresholds, $0 \leq \theta_0 < \theta_1 \leq 1$, our goal is to find all sets S (or as many as possible), for which

$$\mathbb{P}_n(S \subseteq X | Y = 1) \geq \theta_1 \quad \text{and} \quad \mathbb{P}_n(S \subseteq X | Y = 0) \leq \theta_0. \tag{3.1}$$

Here and throughout this chapter, we use the subscript n to indicate that the probabilities are empirical probabilities. For example, for $c \in \{0, 1\}$,

$$\mathbb{P}_n(S \subseteq X | Y = c) := \frac{1}{|C_c|} \sum_{i \in C_c} \mathbb{1}_{\{S \subseteq X_i\}},$$

where we have denoted the set of observations in class c by C_c. Of course, one would also be interested in sets S which satisfy a version of (3.1) with classes 1 and 0 interchanged, but we will only consider (3.1) for simplicity.

The interaction terms uncovered can be used in various ways. For example, they can be built into tree-based methods, or form new features in linear or logistic regression models. The interactions may also be of interest in their own right, as they can characterise distinctions between classes in a simple and interpretable way. These potentially high order interactions that our method aims to target would be very difficult to discover using existing methods, as we now explain.

A pure brute force search examines each potential interaction S of a given size to check whether it fulfils (3.1). Restricting the order of interactions to size s, the computational complexity scales as p^s, rendering problems with even moderate values of p infeasible.

Instead of searching through every possible interaction, tree-based methods build up interactions incrementally. A typical tree classifier such as CART (Breiman et al., 1984) works by building a decision tree greedily from root node to the leaves; see also Loh and Shih (1997). The feature space is recursively partitioned based on the variable whose presence or absence best distinguishes the classes. The myopic nature of this strategy makes it a computationally feasible approach, even for very large problems. The downside is that it produces rather unstable results: small changes in the data can lead to very different partitions being produced at the leaf nodes. Moreover, because of the incremental way in which interactions are constructed, the success of this strategy in recovering an important interaction S rests on at least some of its lower order interactions being informative for distinguishing the classes.

Approaches based on tree ensembles can somewhat alleviate the problem of tree instability; Random Forests (Breiman, 2001) is a prominent example. Here the data with which the decision trees are constructed is sampled with replacement from the original data. Further randomness is introduced by randomising over the subset of variables considered for each split in the construction of the trees. While the results of Random Forests are very complex and hard to interpret, one can examine what are known as variable importance measures. These aim to quantify the marginal or pairwise importance of predictor variables (Strobl et al., 2008). Though such measures can be useful, checking through all possible high order interactions is too cumbersome, and so these may fail to be highlighted.

More recently, there has been interest in algorithms that start from deep splits or leaf nodes in trees and then try to build a simpler model out of many thousands of these leaves by regularisation and dimension reduction. Examples include Rule Ensembles (Friedman and Popescu, 2008), Node Harvest (Meinshausen, 2010) and the general framework of Decision Lists (Marchand and Sokolova, 2006; Rivest, 1987). Though these methods have been demonstrated to improve on Random Forests in some situations, they nevertheless crucially rely on a good initial basis of leaf nodes. These bases are usually generated by tree ensemble methods and so, if the base trees miss some important splits, they would also be absent in the results of these derivative algorithms.

A complementary approach has developed in data mining under the name of frequent itemset search, starting with the Apriori algorithm (Agrawal et al., 1994), which has since then developed into many improved and more specialised forms. The starting point for these was 'market basket analysis', where the shopping behaviour of customers is analysed and the goal is to identify baskets that are often bought together. Many algorithms have been proposed that aim to improve on Apriori in terms of memory requirements and speed, such as the FP-growth (Han et al., 2000) and H-mine (Pei et al., 2001) algorithms. While generally very successful, all these methods are only

computationally feasible in large-scale settings if among the itemsets of low size, there are many that are infrequent, and so using the principle that subsets of frequent itemsets are also frequent, the search space can be greatly reduced. However, if small itemsets all have roughly the same frequency, these methods cannot greatly improve over a brute force search.

We now give a simple example where tree-based approaches and those based on the Apriori algorithm will struggle. Let $Z = (Z_1, \ldots, Z_p) \in \{0,1\}^p$ be a random variable with p independent components each having a Bernoulli(1/2)-distribution. We take X to be the set of active entries $\{k : Z_k = 1\}$. Suppose the response $Y \in \{0,1\}$ is determined by an interaction between the first two variables such that $Y = \mathbb{1}_{\{Z_1+Z_2\neq 1\}}$. Then none of the variables have a marginal effect as Y is independent of Z_k for all $k = 1, \ldots, p$. In this case, when using trees or the Apriori algorithm, one would have to search among $O(p^2)$ potential interactions to find the interaction pattern $\{1,2\}$.

Here we look at a new way to discover interactions, which we call Random Intersection Trees. Rather than searching through potential interactions directly, our method works by looking for collections of observations whose common active variables together form informative interactions. We present a basic version of the Random Intersection Trees algorithm in the following section. This approach allows for computationally feasible discovery of interactions in settings where most existing procedures would perform poorly. Bounds on the complexity of our algorithm are given in Section 3.3. For example, our results yield that in the scenario discussed in the previous paragraph, the order of computational complexity of our method is at most $o(p^\kappa)$ for any $\kappa > 1$. In Section 3.4, we propose some modifications of our basic method to reduce its computational cost, based on min-wise hash schemes. Some numerical examples are given in Section 3.5. We conclude with a brief discussion in Section 3.6, and all technical proofs are collected in the appendix of this chapter.

3.2 Random Intersection Trees

Our method searches for important interactions by looking at intersections of randomly chosen observations from class 1. We start with the full set of variables as an interaction and then iteratively prune away variables to make the interaction smaller. At each iteration, we just keep variables in the interaction that are present in a new randomly chosen observation of class 1. All variables in the interaction that are not present in the chosen observation are removed. Then we repeat with a new randomly chosen observation until an interaction of the desired size emerges. If a pattern S has high prevalence in class 1, i.e. $\mathbb{P}_n(X = S|Y = 1)$ is large, it will be included in the observations chosen with high probability. Thus, provided the overall process is repeated often enough, S is likely to be retained in some of the final intersections. On the other hand, elements in S^c, the complement of S in $\{1, \ldots, p\}$, are unlikely to be present in all the observations being intersected. Thus of those intersections which contain S, there is a good chance that at least one of them is exactly S. Arranging the procedure in a tree-type search makes performing the intersections more computationally efficient; details are given in the following section. One would then consider each of these intersections as possible solutions of (3.1), checking whether their prevalence among class 0 is below θ_0.

It may at first seem strange that in the above, class 0 plays a part in the procedure only at the very end. One might expect that many candidate interactions could be generated that have high prevalence in both classes 1 and 0 and thus would not be useful for distinguishing between classes. In Section 3.4, we do present an improved version of our algorithm that makes use of class 0 at an earlier stage. However, in the sparse setting we are considering here, interactions with high

prevalence in either class would typically be rather few in number. Thus even if all interactions with high prevalence in class 1, and not necessarily low prevalence in class 0, were generated by the procedure outlined above, this would be a manageable number of candidate sets. Note that the assumptions that allow this to happen certainly do not trivialise the problem: even if, given all solutions to the first equation in (3.1), it is easy to uncover those interactions that additionally satisfy the second equation, the first part of the task is still very challenging.

To describe the details of our algorithm, we first define some terms associated with trees that will be needed later. Recall that a tree is a pair (N, E) of nodes and edges forming a connected acyclic (undirected) graph. We will always assume (with no loss of generality) that $N = \{1, \ldots, |N|\}$. A *rooted tree* is the directed acyclic graph obtained from a tree by designating one node as root and directing all edges away from this root.

Let α and β be two nodes in a rooted tree, with β not the root node. If $(\alpha, \beta) \in E$, β is said to be the *child* of α, and α, the *parent* of β. We will denote by $\mathrm{ch}(\alpha)$, the set of children of a node α. Since we are only considering rooted trees here as opposed to general directed graphs, we will differ with convention slightly and will use $\mathrm{pa}(\beta)$ to mean the unique parent of β. Thus here, $\mathrm{pa}(\beta)$ is a node itself, whereas $\mathrm{ch}(\alpha)$ is a set of nodes.

If $\alpha \neq \beta$ lies on the unique path from the root to β, we say α is an *ancestor* of β, and β is a *descendant* of α. We denote the sets of all ancestors and descendants of α by $\mathrm{an}(\alpha)$ and $\mathrm{de}(\alpha)$ respectively. The *depth* of α, denoted $\mathrm{depth}(\alpha)$, is the number of ancestors of α: $\mathrm{depth}(\alpha) = |\mathrm{an}(\alpha)|$. In particular, the depth of the root node is 0. The *depth* (also known as the *height*) of a rooted tree is the length of the longest path, or equivalently, the greatest number of ancestors of any particular node. By *level* d of the tree, we will mean the set of nodes with depth d.

We will say an indexing of the nodes is *chronological* if, for every parent and child pair, larger indices are assigned to the child than the parent. In particular, the root node will be 1. Note that both depth-first and breadth-first indexing methods are chronological in this way.

Algorithm 4 A basic version of Random Intersection Trees

for tree $t = 1$ **to** T **do**

Let t be a rooted tree of depth D, with each node j in levels $0, \ldots, D-1$ having B_j children, where the B_j are i.i.d. with a pre-specified distribution. Denote by J the total number of nodes in the tree, and index the nodes chronologically. For each of the nodes $j = 1, \ldots, J$, let $i(j)$ be an independently and uniformly chosen index in the set of class 1 observations $\{i : Y_i = 1\}$.

Set $S_1 = X_{i(1)}$.
for node $j = 2$ **to** J **do**
Set $S_j = X_{i(j)} \cap S_{\mathrm{pa}(j)}$.
end for

Denote the collection of resulting sets from all nodes at depth d, for $d = 1, \ldots, D$, by $L_{d,t} = \{S_j : \mathrm{depth}(j) = d\}$.
end for
return candidate set of interactions $L_D := \bigcup_{t=1}^{T} L_{D,t}$.

Algorithm 1 describes a basic version of the Random Intersection Trees procedure. The reason for allowing random choices of children is for the proof of Theorem 1, where we can randomly choose the number of children to be in $\{b, b+1\}$ for a suitable integer value b. Although we have allowed the number of children of each non-leaf node in the trees to be random, in practice we

would take this as a fixed number.

Looking at the innermost for-loop, we see that each node in each tree is associated with a randomly drawn observation from class 1. For every tree, we visit each non-root node in turn, and compute the intersection of the observation assigned to it, and all those assigned to its ancestors. Because of the way the nodes are indexed, parents are always visited before their children, and this intersection can simply be computed as $S_j = X_{i(j)} \cap S_{\mathrm{pa}(j)}$. This is crucial to reducing the computational complexity of the procedure, as we shall see in the next section.

Each of the sets assigned to the leaf nodes of each of the trees yields a collection of potential candidate interactions, L_D. One could then proceed to test these as potential solutions to (3.1); we present a more efficient approach in Section 3.4, where we build this testing step into the construction of the trees.

An illustration of this improved algorithm applied to the Tic-Tac-Toe data discussed in Section 3.5 is given in Figure 3.1. Observations here correspond to winning endgame positions, coded such that the data is binary. Class labels record which player (black or white) won the game, and the goal is to infer the interactions (corresponding to positions of a few counters) that lead to a win for each player. In this example, the root node contains a randomly drawn final win-state for black (class 1). This corresponds to S_1 in our algorithm. For each other node j, we draw a new random observation $i(j)$ from all class 1 observations. The randomly chosen additional black-win state $X_{i(j)}$ is shown along the edge from its parent node. The new intersection, S_j, is the intersection of the interaction in the parent node and the new set $X_{i(j)}$; it is shown in the corresponding node. The early stopping added in the improved algorithm also allows it to run until the algorithm has terminated in all nodes. Thus no prior specification of the tree depth will be necessary in practice, as will be shown in Section 3.4.

3.3 Computational complexity

How many trees do we have to compute to have a very high probability of finding an interesting interaction S that fulfils (3.1)? And what is the required size of these trees? If the interaction is not associated with a main effect, most approaches like trees and association rules would require of order $p^{|S|}$ searches. In this section, we show that in many settings, Random Intersection Trees improves on this complexity. We consider a single interaction S of size $s := |S|$, and examine the computational cost for returning S as one of the candidate interactions, with a given probability. We will see that this depends critically on three factors:

- **Prevalence** $\theta_1 := \mathbb{P}_n(S \subseteq X | Y = 1)$ of the interaction pattern. If the pattern S in question appears frequently in class 1, the search is more efficient.

- **Sparsity** $\delta_k := \mathbb{P}_n(k \in X | Y = 1)$ of the predictor variables $k = 1, \ldots, p$. If δ_k is very low for many k (and sparsity of predictors consequently high), computation of the intersections is much cheaper, and so overall computational cost is greatly reduced. Indeed, for a fixed tree t, consider a node j with depth $d < D$. We have that

$$\mathbb{E}(|S_j|) = \sum_{k=1}^{p} \delta_k^{d+1}.$$

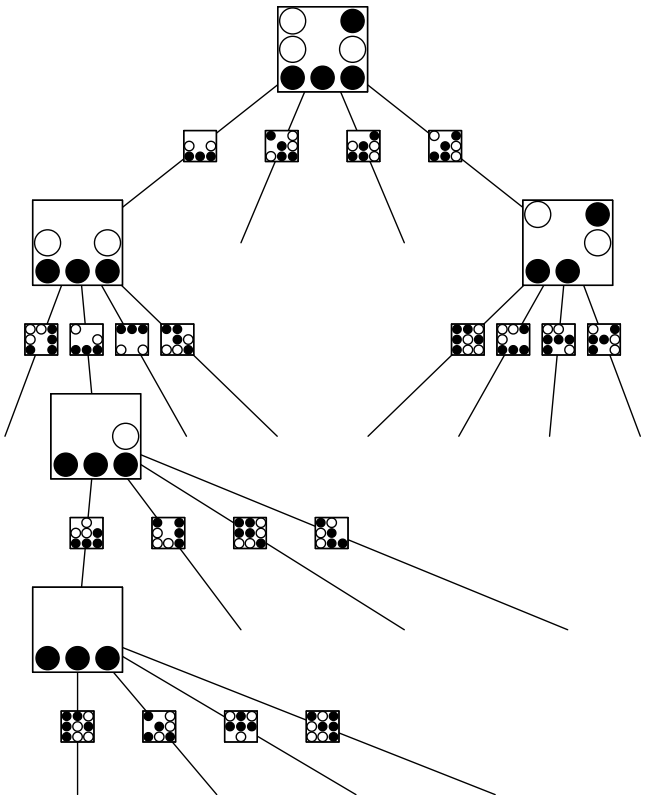


Figure 3.1: An intersection tree for the Tic-Tac-Toe game dataset. Given winning positions of the black player, we intersect them randomly to produce the interactions (corresponding to positions of black or white stones) that are responsible for wins. Starting with a randomly chosen class 1 (black wins) observation at the root node, $B = 4$ randomly chosen class 1 observations are intersected with the pattern. These randomly chosen observations are shown along the edges and the resulting intersections S_j as the nodes in the next layer of the tree. Nodes are only shown if the corresponding patterns S_j have an estimated prevalence among class 0 below a set threshold; the branching of the tree terminates for all other nodes. The algorithm continues until all resulting S_j corresponding to the leaf nodes have prevalence among class 0 exceeding the threshold. Here, one of the winning states for black is filtered out after three intersections.

Thus, for $j' \in \mathrm{ch}(j)$, computation of $S_{j'}$ requires on average at most

$$O\left(\log(p)\sum_{k=1}^{p}\delta_k^{d+1}\right)$$

operations. This is because in order to compute the intersection, one can check whether each member of S_j is in $X_{i(j')}$, and each such check is $O(\log(p))$ if the sets X_i are ordered so a binary search can be used. If we compare this to the $O(p)$ computations required to calculate each of the S_j if no tree structure were used, we see that large efficiency gains are possible when $d \geq 1$ if many variables are sparse. For intersections with the root node, the tree structure offers no advantage, and in practice, branching the tree only after level 1 (so the root node has only one child), is more efficient, though this modification does not improve the order of complexity.

- **Independence of** S: Define $\nu := \max_{k\in S^c}\mathbb{P}_n(k \in X | S \subseteq X, Y = 1)$. If ν is low, less computational effort is required to recover S. Note that if, for some $k \in S^c$, $\mathbb{P}_n(k \in X | S \subseteq X) = 1$, interest would centre on $S \cup \{k\}$ rather than S itself. Indeed, if S satisfied (3.1), so would $S \cup \{k\}$. In general, if ν is large, the search will tend to find sets containing S, though not necessarily S itself.

With the assumptions that $\theta_1 > 0$ and $\nu < 1$, we can give a bound on the computational complexity of the basic version of Random Intersection Trees introduced in the previous section.

Let us define

$$C(T, D, F_B)$$

to be the expected number of computations required to perform all the intersections in the algorithm when T trees of depth D are created and the distribution of the branching factors B_j is F_B.

Theorem 15. *Given $\eta, \epsilon \in (0, 1]$, there exist choices of T, D and F_B such that the set L_D returned by Algorithm 4 contains S with probability at least $1 - \eta$, and*

$$C(T, D, F_B) = O\left[\log(1/\eta)\frac{\log^2(p)}{\epsilon}\left\{p + \sum_{k:\,(1+\epsilon)\delta_k > \theta_1} p^{\frac{\log\{(1+\epsilon)\delta_k/\theta_1\}}{\log(1/\nu)}}\right\}\right]. \tag{3.2}$$

As a function of the number of variables p, there is a contribution of $p\log^2(p)$ and an additional contribution in the brackets that depends on the sparsity δ_k of each variable. Sparse variables do not contribute to this sum, which can be $O(1)$ if sparsity among variables is high enough. This would yield a computational complexity with order bounded above by $o(p^\kappa)$ for any $\kappa > 1$, compared to the corresponding complexity of p^s for a brute force search. In most interesting settings, however, we would not achieve a nearly linear scaling in complexity, but would hope to still be faster than a brute force search.

Before discussing the result further, we comment briefly on the values of T, D and the distribution of the B_j, that yield (3.2). From the proof, it follows that there exist choices of T and D

giving (3.2) that satisfy

$$T \leq \frac{(1+2\epsilon)\log(1/\eta)}{2\epsilon\theta_1},$$
$$D \leq \frac{\log\{p(1+2\epsilon)\}}{\log(1/\nu)}.$$

The random number B_j used in the proof takes just one of two consecutive integers (essentially to avoid the discretisation effect when being restricted to integers), and $\mathbb{E}(B_j) \leq (1+\epsilon)/\theta_1$. Though the optimal choices of parameters for the theorem depend on the unknown ν and the minimising ϵ, which will in turn depend on ν, the functional relationships given above still provide rough qualitative guidelines for good choices for these parameters in practice.

Using the values of T, D and B_j necessary to guarantee that with high probability the set S is in the set L_D, we can also obtain a bound on the expected number of candidate interaction sets in L_D. This will in turn bound the expected number of 'false positives' returned. The expected number of sets returned is bounded by

$$\mathbb{E}(|L_D|) \leq T\mathbb{E}(B_j)^D \ \leq \frac{\log(1/\eta)}{\epsilon}\left\{\frac{(1+2\epsilon)p}{\nu}\right\}^{\frac{\log(1+\epsilon)/\theta_1}{\log(1/\nu)}}$$

The value of ϵ can be chosen to minimise the bound above, but its value here and in the computational complexity bound of Theorem 15 have to be the same, as they are linked to the choice of the branching factor used when building the trees. We see that in many situations, we can expect the bound above to be very much lower than the $O(p^s)$ sets a complete list of s-way interactions would contain. Note that if s were known, the relevant quantity to consider would be

$$\mathbb{E}(|\{S' \in L_D : |S'| = s\}|),$$

which is likely to be much less than $\mathbb{E}(|L_D|)$. Even if s were unknown, one would only be interested in the expected number of non-empty sets in L_D, a quantity which may well also be substantially lower than the derived bound on $\mathbb{E}(|L_D|)$.

The influence of sparsity on computational complexity. It is interesting to make the influence of the sparsity of individual variables, δ_k, on the overall computational complexity, more explicit. We have the following corollary to Theorem 15.

Corollary 16. *Define β by $\nu = \theta_1^\beta$. Suppose that $\gamma, \alpha^\star, \alpha_\star$ are such that $\alpha^\star > \alpha_\star$, and*

$$\begin{aligned} \delta_k &\leq \theta_1^{1-\alpha^\star} && \text{for all } k \in \{1,\ldots,p\} \\ \delta_k &> \theta_1^{1-\alpha_\star} && \text{for at most } p^\gamma \text{ variables.} \end{aligned}$$

Given $\eta \in (0,1]$, there exist choices of T, D and F_B such that the set L_D returned by Algorithm 4 contains S with probability at least $1-\eta$, and

$$C(T,D,F_B) = o\big(p^\kappa\big) \qquad \text{for any } \kappa > \max\left\{\frac{\alpha^\star}{\beta} + \gamma,\ \left[\frac{\alpha_\star}{\beta}\right]_+ + 1\right\}. \tag{3.3}$$

The implication of Corollary 16 is most apparent if we take $\gamma = 1$ as we can then set $\alpha_\star = 0$.

In this case,

$$\alpha^\star = 1 - \frac{\log(\max_k \delta_k)}{\log(\theta_1)}.$$

We can then bound the computational complexity by

$$o\big(p^\kappa\big) \qquad \text{for any } \kappa > 1 + \frac{\log(1/\theta_1) - \log(1/\max_k \delta_k)}{\log(1/\nu)}. \tag{3.4}$$

The fraction on the right-hand side is a function of the prevalence of the pattern S, θ_1, the maximum sparsity of the variables, and the maximum sparsity of the variables in S^c, conditional on the presence of S. As long as this fraction is less than 1, the computational complexity is guaranteed to be better than a brute force search with the knowledge that $s = 2$, and the relative advantage grows for larger sizes of the pattern.

Independent noise variables. To gain further insight, we consider the special case where variables in S^c are independent of S (conditional on being in class 1), in the sense that for all $k \in S^c$,

$$\mathbb{P}_n(k \in X | S \subseteq X, Y = 1) = \mathbb{P}_n(k \in X | Y = 1) = \delta_k. \tag{3.5}$$

Corollary 17. *Assume* (3.5) *and that* $\delta_k < 1$ *for all* k. *Given* $\eta \in (0, 1]$, *there exist choices of* T, D *and* F_B *such that the set* L_D *returned by Algorithm 4 contains* S *with probability at least* $1 - \eta$, *and*

$$C(T, D, F_B) = o(p^\kappa) \qquad \textit{for any } \kappa > \frac{\log(1/\theta_1)}{\log(1/\max_k \delta_k)}. \tag{3.6}$$

We see that the computational complexity is approximately linear in p if the prevalence of the pattern S is as high as the prevalence of the least sparse predictor variables. This is the case in the example mentioned in the introduction, where $\theta = \delta_k = 1/2$.

We can also consider the situation where in addition to the independence (3.5), all variables have the same sparsity δ. If the prevalence θ_1 of S is only as high as that of a random occurrence of two independent predictor variables, we get $\kappa > 2$ and the computational complexity is approximately quadratic in p. In this case, the algorithm would not yield a computational advantage over brute force search if looking for patterns of size 2. This is to be expected since *every* pattern S of size 2 would have the same prevalence in this scenario, and so there is nothing special about a pattern S of size 2 with prevalence δ^2, and in general no hope of beating the complexity p^s of a brute force search. However, the bound in (3.6) is independent of s. Thus provided the prevalence, θ_1, drops more slowly than the rate δ^s, at which every pattern of size S would occur randomly among independent predictor variables, our results show that Random Intersection Trees is still to be preferred over a brute force search.

3.4 Early stopping using min-wise hashing

While Algorithm 1 is computationally attractive, the following observation suggests that further improvements are possible. Suppose that, for a particular tree, we have just computed the intersection S_j corresponding to a node j at depth $d < D$. If

$$\mathbb{P}_n(S_j \subseteq X | Y = 0) > \theta_0,$$

then since for all $j' \in \mathrm{de}(j)$, $S_{j'} \subseteq S_j$, we also have

$$\mathbb{P}_n(S_{j'} \subseteq X | Y = 0) > \theta_0.$$

Thus no intersection sets corresponding to descendants of j have any hope of yielding solutions to (3.1), and so all further associated computations are wasted.

In view of this, one option would be to compute the quantity $\mathbb{P}_n(S_j \subseteq X | Y = 0)$ at each node j as the algorithm progresses, and if this exceeds the threshold θ_0, not visit any descendants j' of j for computation of $S_{j'}$. This could be prohibitively costly, though, as it would require a pass over all observations in class 0, for each node of each tree. One could work with a subsample of the observations, but if θ_0 is low, the subsample size may need to be fairly large in order to estimate the probabilities to a sufficient degree of accuracy.

Instead, we propose a fast approximation, using some ideas based on min-wise hashing (Broder et al., 1998; Cohen et al., 2001; Datar and Muthukrishnan, 2002) applied to the columns of the data-matrix. We describe the scheme by leaving aside the conditioning on $Y = 0$, which can be added at the end by restricting to observations in class 0. Consider taking a random permutation σ of all observations $\{1, \ldots, n\}$. Let $m_\sigma(k)$ be the minimal value ι such that variable k is active in observation $\sigma(\iota)$:

$$m_\sigma(k) := \min\{\iota' : k \in X_{\sigma(\iota')}\}.$$

It is well known (Broder et al., 1998) that the probability that $m_\sigma(k)$ and $m_\sigma(k')$ agree for two variables k, k' under a random permutation σ is identical to the Jaccard-index for the two sets $I_k = \{i : k \in X_i\}$ and $I_{k'} = \{i : k' \in X_i\}$, that is

$$\mathbb{P}_\sigma(m_\sigma(k) = m_\sigma(k')) \;=\; \frac{|I_k \cap I_{k'}|}{|I_k \cup I_{k'}|}.$$

Here the subscript σ indicates that the probability is with respect to a random permutation σ of the observations. A min-wise hash scheme is typically used to estimate the Jaccard-index by approximating the probability on the left-hand side of the equation above.

Now,

$$\begin{aligned}\mathbb{P}_n(S \subseteq X) &= \mathbb{P}_n(k \in X \text{ for all } k \in S) \\ &= \mathbb{P}_n(k \in X \text{ for all } k \in S \,|\, \exists\, k' \in S \text{ such that } k' \in X) \\ &\quad \times \mathbb{P}_n(\exists k \in S \text{ such that } k \in X).\end{aligned}$$

Let us denote the first and second terms on the right-hand side by $\pi_1(S)$ and $\pi_2(S)$ respectively. Note that $\pi_1(S)$ is equal to the probability that all variables $k \in S$ have the same min-wise hash value $m_\sigma(k)$:

$$\pi_1(S) = \mathbb{P}_\sigma(\exists\, \iota : m_\sigma(k) = \iota \text{ for all } k \in S). \tag{3.7}$$

Turning now to $\pi_2(S)$, observe that

$$\mathbb{E}_\sigma(\min_{k \in S} m_\sigma(k)) = \frac{n+1}{\pi_2(S)n + 1}, \tag{3.8}$$

and so

$$\pi_2(S) = \frac{n+1}{n}\left\{\frac{1}{\mathbb{E}_\sigma(\min_{k \in S} m_\sigma(k))} - \frac{1}{n+1}\right\}. \tag{3.9}$$

A derivation of (3.8) is given in the appendix of this chapter.

Equations (3.7) and (3.9) provide the basis for an estimator of $\mathbb{P}_n(S \subseteq X)$. First we generate L random permutations of $\{1, \ldots, n\}$: $\sigma_1, \ldots, \sigma_L$. We then use these to create an $L \times p$ matrix M whose entries are given by

$$M_{lk} = m_{\sigma_l}(k).$$

Now we estimate $\pi_1(S)$ and $\pi_2(S)$ by their respective finite-sample approximations, $\hat{\pi}_1(S)$ and $\hat{\pi}_2(S)$:

$$\hat{\pi}_1(L; S, M) := \tfrac{1}{L} \sum_{l=1}^{L} \mathbb{1}_{\{M_{lk} = M_{lk'} \text{ for all } k,k' \in S\}}$$

$$\hat{\pi}_2(L; S, M) := \frac{n+1}{n} \left\{ \frac{1}{\frac{1}{L}\sum_{l=1}^{L} \min_{k \in S} M_{lk}} - \frac{1}{n+1} \right\}.$$

Finally, we estimate $\mathbb{P}_n(S \subseteq X)$ by

$$\hat{\mathbb{P}}_n(L; S, M) := \hat{\pi}_1(L; S, M) \cdot \hat{\pi}_2(L; S, M). \tag{3.10}$$

To our knowledge, this use of min-wise hashing techniques, and in particular the estimator $\hat{\pi}_2(L; S, M)$, is new. The estimator enjoys reduced variance compared to that which would be obtained using subsampling, as the following theorem shows.

Theorem 18. *For $\hat{\mathbb{P}}_n(L; S, M)$, $\pi_1(S)$ and $\pi_2(S)$ defined as in (3.10), (3.7), and (3.9) respectively, as $L \to \infty$, we have*

$$\sqrt{L}(\hat{\mathbb{P}}_n(L; S, M) - \mathbb{P}_n(S \subseteq X)) \xrightarrow{d} N(0, \pi_2(S)^2 \pi_1(S)(1 - \pi_1(S)\pi_2(S))(1 + \epsilon(n))), \tag{3.11}$$

where

$$\epsilon(n) = \frac{1}{n} \frac{n^{-1} - \pi_2^2 - 2\pi_2 n^{-1}}{\pi_2(\pi_2 + 2n^{-1})(1 + n^{-1})} = O(n^{-1}). \tag{3.12}$$

A derivation is given in the Appendix of this chapter. If we tried to estimate $\pi_1\pi_2$ by evaluating the prevalence of S on a subset of the data of size L, the corresponding estimator multiplied by $\sqrt{L}$ would have variance

$$\pi_2(S)\pi_1(S)(1 - \pi_1(S)\pi_2(S)) + o_n(1),$$

where $o_n(1) \to 0$ as $n \to \infty$. Comparing this variance to the variance of the normal distribution in (3.11), we see that a factor of $\pi_2(S)$ is gained: matching the accuracy of the the min-wise hash scheme with subsampling would require roughly $1/\pi_2(S)$ times as many samples. By using min-wise hashing, choosing $L = 100$ typically delivers a reasonable approximation as long as we just want to resolve values at $\theta_0 = 0.01$ and above.

An improved version of Algorithm 1, building in the ideas discussed above, is given in Algorithm 2 below. Note that $\hat{\mathbb{P}}_n(S_{\mathrm{pa}(j)}, M)$ need only be computed once for every j with the same parent.

Early stopping decreases the computational cost of the algorithm as many nodes in the trees generated may not need to have their associated intersections calculated. In addition, the set of candidate intersections L_D will be smaller but the chance of it containing interesting intersections would not decrease by much. These gains comes at a small price, since the min-wise hash matrix M must be computed, and the computational effort going into this will in turn determine the quality of the approximation in (3.10). We have previously shown the complexity bounds in the absence

Algorithm 5 Random Intersection Trees with early stopping

Compute the $L \times p$ min-wise hash matrix M, using only class 0 observations.
for tree $t = 1$ **to** T **do**
  Let t be a rooted tree of depth D, with each node j in levels $0, \ldots, D-1$ having B_j children, where the B_j are i.i.d. with a pre-specified distribution. Denote by J the total number of nodes in the tree, and index the nodes chronologically. For each of the nodes $j = 1, \ldots, J$, let $i(j)$ be an independently and uniformly chosen index in the set of class 1 observations $\{i : Y_i = 1\}$.

  Set $S_1 = X_{i(1)}$.
  for node $j = 2$ **to** J **do**
    if $\hat{\mathbb{P}}_n(S_{\mathrm{pa}(j)}, M) \leq \theta_0$ **then**
      Set $S_j = X_{i(j)} \cap S_{\mathrm{pa}(j)}$.
    end if
  end for

  Denote the collection of resulting sets of all nodes at depth d, for $d = 1, \ldots, D$, by $L_{d,t} = \{S_j : \mathrm{depth}(j) = d\}$.
end for
return $L_D := \bigcup_{t=1}^{T} L_{D,t}$.

of early stopping and thus avoided the difficulty of making this trade-off explicit. We will use the improved version of Random Intersection Trees with early stopping in all the practical examples to follow, taking small values of L in the range of a (few) hundred permutations.

The depth D of the tree is still given explicitly in Algorithm 2. An interesting modification creates the tree recursively. Starting with the root node, B children are added to all leaf nodes of the tree in which the early stopping criterion has not been triggered yet. When the algorithm terminates, all intersections in the leaf nodes of the final tree are collected.

3.5 Numerical Examples

In this section, we give two numerical examples to provide further insight into the performance of our method. The first is about learning the winning combinations for the well-known game Tic-Tac-Toe. This example serves to illustrate how Random Intersection Trees can succeed in finding interesting interactions when other methods fail. The second example concerns text classification. Specifically, we want to find simple characterisations (using only a few words, or word-stems in this case) for classes within a large corpus in a large-scale text analysis application.

3.5.1 Tic-Tac-Toe endgame prediction

The Tic-Tac-Toe endgame dataset (Aha et al., 1991; Matheus and Rendell, 1989) contains all possible winning end states of the game Tic-Tac-Toe, along with which player (white or black) has won for each of these. There are just under 1000 possible such end states, and our goal is to learn the rules that determine which player wins from a randomly chosen subset of these. We use half of the observations for training, and the other half for testing.

There are 9 variables in the original dataset which can take the values 'black', 'white' or 'blank'. These can trivially be transformed into a set of twice as many binary variables where the first block of variables encodes presence of black and the second block encodes presence of white.

Two properties of this dataset that make it particularly interesting for us here are:

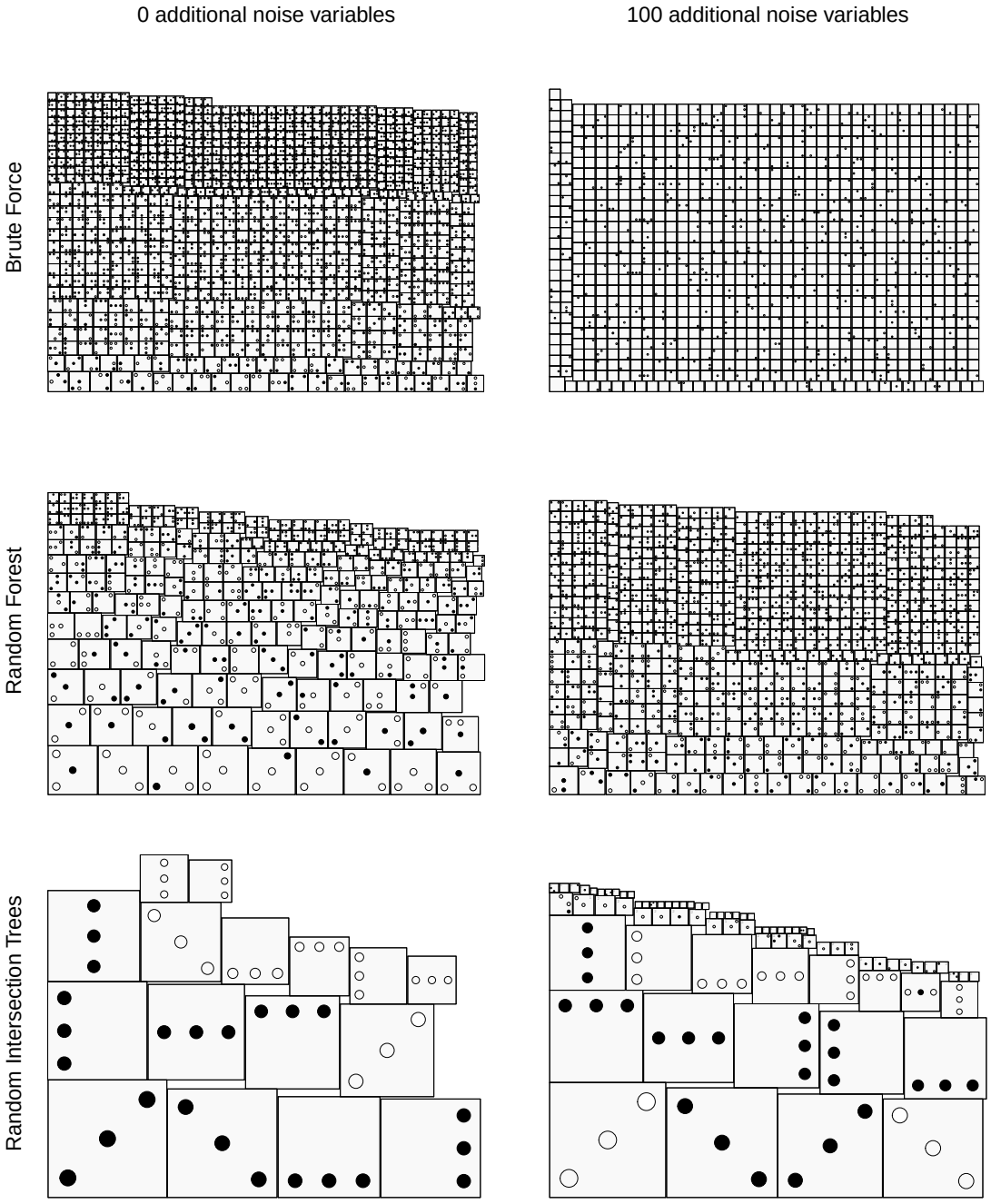


Figure 3.2: Left panel: patterns that are returned by Random Intersection Trees (bottom row), Random Forests of depth 3 (middle row) and brute force search among all interactions of size 3 (top row) for the Tic-Tac-Toe data. Each pattern is scaled to make the area proportional to the empirical frequency with which each pattern is found by these search algorithms. Right panel: the same results in the case when 100 noise variables are added. Note that Random Intersection Trees were not constrained to find interactions of depth 3. In the case with noise variables, some of the patterns with the very smallest areas also contained a small number of noise variables, which are not shown. Just counting three- to five-way interactions, there are more than 10^8 potential interactions when 100 noise variables are added.

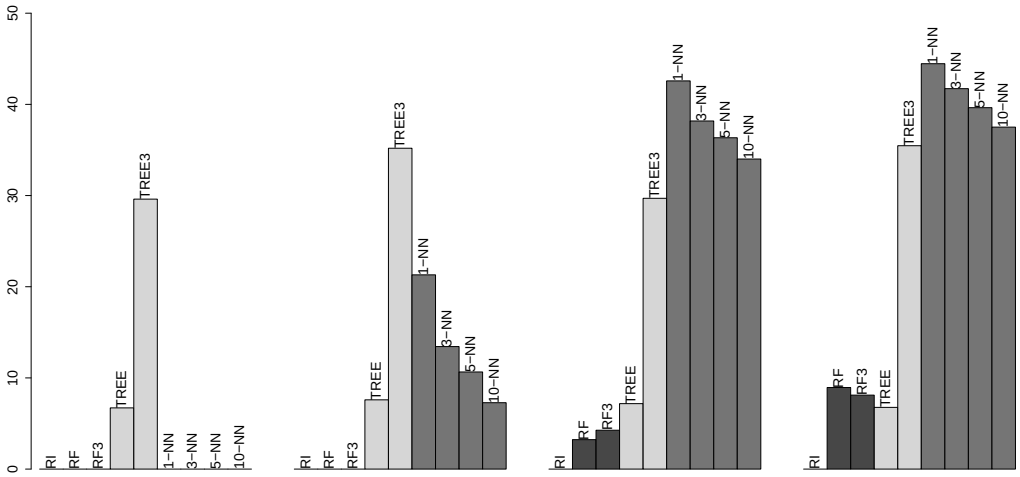


Figure 3.3: From left to right: the misclassification rate (in %) on Tic-Tac-Toe data for 0, 60, 300 and 400 added noise variables. Each classifier is tuned to have equal misclassification rate in both classes. The simple classifier based on Random Intersection Trees (RI) has a misclassification rate of 0% in all cases, as the winning patterns are sampled very frequently (see Figure 3.2). Random Forests (RF) and Random Forests limited to depth 3 trees (RF3) are competitive but the misclassification rate increases sharply when many noise variables are added.

- The presence of interactions is obvious by the nature of the game.

- There are only very weak marginal effects. Knowing that the upper right corner is occupied by a black stone is only very weakly informative about the winner of the game. Greedy searches by trees fail in the presence of many added noise variables and linear models do not work well at all.

We apply Random Intersection Trees to finding patterns that indicate a black win (class 1), and also patterns that indicate a white win (class 0). We use the early stopping modifications proposed in Section 3.4, and create two min-wise hash tables from the available observations in each of the classes, taking $L = 200$. Figure 3.1 shows how the individual Intersection Trees are constructed and illustrates the use of the early stopping rule. We emphasise that we do not need to specify or know that the winning states are functions of only three variables. We let each tree run until all its branches terminate, and collect all resulting leaves.

Figure 3.2 illustrates the importance sampling effect of Random Intersection Trees when using only the training data, and adding a varying number of noise variables. When adding 100 noise variables, all 16 winning final combinations are among the 40 most frequently chosen patterns. All winning states are chosen hundreds of millions times more often than a random sampling of interactions would pick them.

As discussed in Section 3.1, the interactions or rules that are found could be entered into any existing aggregation method, such as Rule Ensembles (Friedman and Popescu, 2008) or Decision Lists (Marchand and Sokolova, 2006; Rivest, 1987). Here, we consider an even simpler aggregation method by selecting all patterns during 1000 iterations of Random Intersection Trees (with $B = 5$ samples as branching factor in each tree) that were selected by at least two trees. For each selected pattern, we compute the (empirical) class distributions conditional on the presence and absence of

the pattern, using the training sample. That is, for each selected pattern S, we compute

$$\mathbb{P}_n(Y = 1|X \subseteq S) \text{ and } \mathbb{P}_n(Y = 1|X \nsubseteq S).$$

Then, given an observation from the test set, we classify according to the average of the log-odds of being in class 1 calculated from each of the conditional probabilities above.

Figure 3.3 shows the misclassification rates under situations with different numbers of added noise variables. The simple prediction based on Random Intersection Trees achieves perfect classification even when 400 noise variables are added. Neither k-NN nor CART (Breiman et al., 1984), either restricted to trees of depth 3 (TREE3) or depth chosen by cross-validation (TREE), are as successful, giving misclassification rates between 5% and 40%. Interestingly, trees of depth 3 perform much worse than deeper trees. The winning patterns are not identified in a pure form but only after some other variables have been factored in first. This also means that it is very hard to read the winning states of the trees, unlike the patterns obtained by our method. Random Forests also maintain a 0% misclassification rate up until about a hundred added noise variables but start to degrade in performance when further noise variables are added. It is easy to identify the noise variables from a variable importance plot (Strobl et al., 2008). However, within the signal variables the patterns are not easy to see since each variable is approximately equally important for determining the winner (with the slight exception of the middle field in the 3×3 board which is more important than the other fields) and the nature of the interactions is thus not obvious from analysing a Random Forest fit.

3.5.2 Reuters RCV1 text classification

The Reuters RCV1 text data contain the *tf–idf* (term frequency–inverse document frequency) weighted presence of 47,148 word-stems in each document; for details on the collection and processing of the original data, see Lewis et al. (2004). Each document is assigned possibly more than one topic. Here we are interested in whether Random Intersection Trees is able to give a quick and accurate summary of each topic. For each topic, we seek sets of word-stems, S, whose simultaneous presence is indicative of a document falling within that topic.

To evaluate the performance of Random Intersection Trees, we divide the documents into a training and test set with the first batch of 23,149 documents as training and the following 30,000 documents as test documents. We compare our procedure to an approach based on Random Forests and a simple linear method.

Random Forests and classification trees can be very time- and memory-intensive to apply on a dataset of the scale we consider here. In order to be able to compute Random Forests, we only consider word-stems if they appear in at least 100 documents in the training data. This leaves 2484 word-stems as predictor variables. We also only consider topics that contain at least 200 documents. To simplify the problem further, we consider a binary version of the predictor variables for all methods, using a 1 or 0 to represent whether each tf–idf value is positive or not.

Let C be the set of topics in our modified dataset. Let $Y \subseteq C$ indicate the topics that a given document belongs to. Consider a topic or class $c \in C$. Our goal is to find patterns S that maximise

$$\mathbb{P}_n(c \in Y|S \subseteq X), \tag{3.13}$$

whilst also maintaining that the prevalence of S among all observations be bounded away from 0.

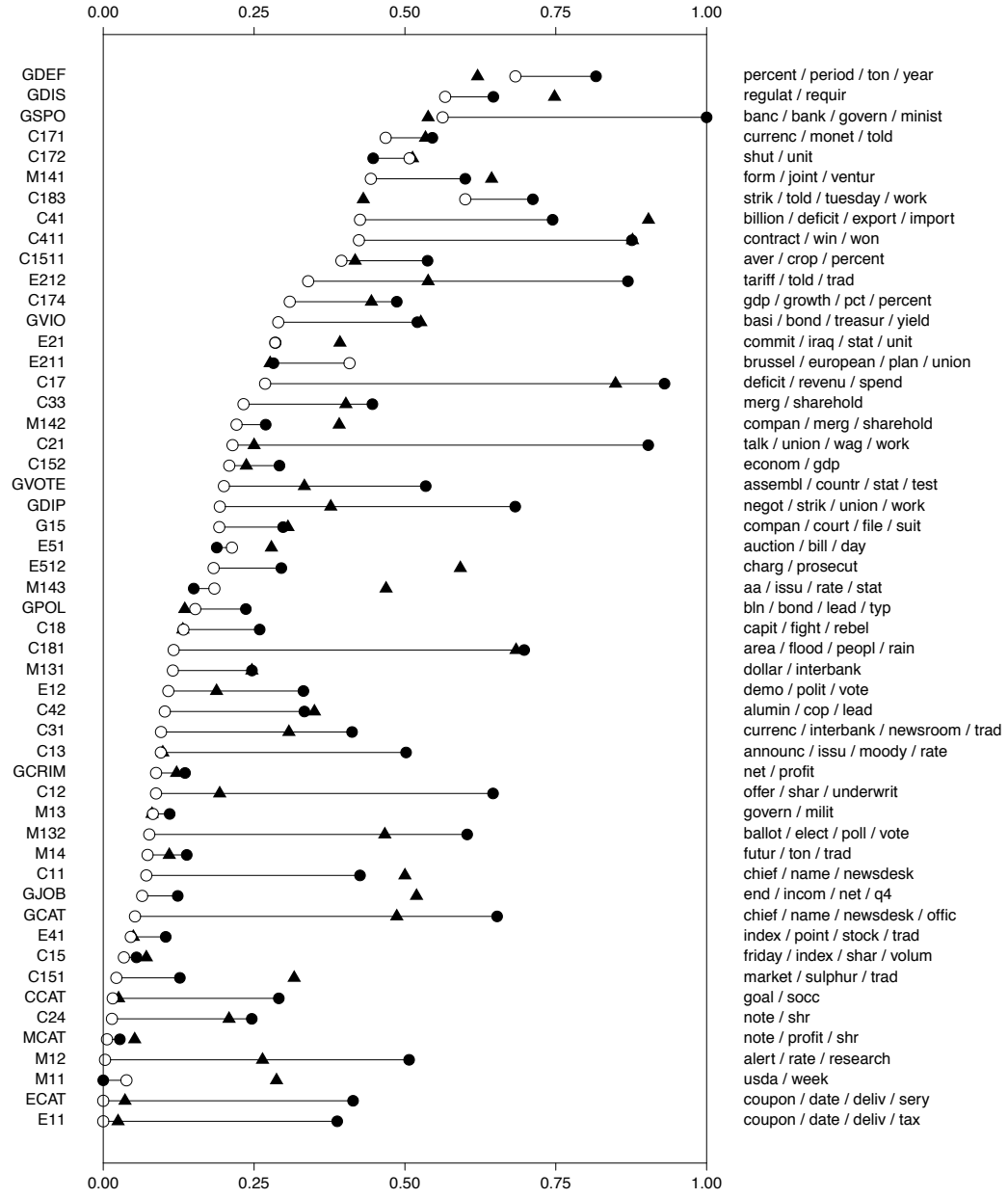


Figure 3.4: The misclassification rate $\mathbb{P}_n(c \notin Y | S \subseteq X)$ on the test data for a pattern S chosen with a tree ensemble node generation mechanism (black circle), Random Intersections (white circle), and a linear method (black triangle) for topics $c \in C$ in the Reuters RCV1 text classification data. The topics are shown on the left and the word combinations chosen by Random Intersection Trees on the right.

Specifically, we shall require that

$$\mathbb{P}_n(S \subseteq X) \geq p_c/10 \text{ where } p_c = \mathbb{P}_n(c \in Y). \tag{3.14}$$

To see how this can be cast within the framework set in (3.1), note that if $S^\star$ maximises (3.13) and $S^{\star\star}$ satisfies

$$\mathbb{P}_n(S^{\star\star} \subseteq X | Y \in c) \geq \mathbb{P}_n(S^\star \subseteq X | Y \in c) \text{ and} \tag{3.15}$$

$$\mathbb{P}_n(S^{\star\star} \subseteq X | Y \notin c) \leq \mathbb{P}_n(S^\star \subseteq X | Y \notin c), \tag{3.16}$$

then

$$\begin{aligned}
\mathbb{P}_n(c \in Y | S^\star \subseteq X) &= \frac{\mathbb{P}_n(S^\star \subseteq X | c \in Y)\mathbb{P}_n(c \in Y)}{\mathbb{P}_n(S^\star \subseteq X | c \in Y)\mathbb{P}_n(c \in Y) + \mathbb{P}_n(S^\star \subseteq X | c \notin Y)\mathbb{P}_n(c \notin Y)} \\
&\leq \frac{\mathbb{P}_n(S^{\star\star} \subseteq X | c \in Y)\mathbb{P}_n(c \in Y)}{\mathbb{P}_n(S^{\star\star} \subseteq X | c \in Y)\mathbb{P}_n(c \in Y) + \mathbb{P}_n(S^{\star\star} \subseteq X | c \notin Y)\mathbb{P}_n(c \notin Y)} \\
&= \mathbb{P}_n(c \in Y | S^{\star\star} \subseteq X),
\end{aligned}$$

whence $S^{\star\star}$ also maximises (3.13) by optimality of $S^\star$. Thus treating those documents belonging to topic c as class 1, and all others as class 0, by solving (3.1) with θ_0 and θ_1 chosen appropriately, we can obtain all solutions to (3.13).

In view of this, we use each of the methods to search for patterns S that have high prevalence for a given topic c. We then remove all patterns that do not satisfy (3.14) on the test data. Then, from the remaining patterns, we select the one that maximises (3.13) on training data. Below, we describe specific implementation details of each of the methods under consideration.

Random Intersection Trees. We create the min-wise hash table for the prevalence among all samples once, using 200 permutations with associated min-wise hash values for each word-stem. Then 1000 iterations of the tree search are performed with a cut-off value $\theta_0 = (3/20)p_c$ and all remaining patterns S with a length less than or equal to 4 are retained.

Random Forests. For a tree-based procedure, one approach is to fit classification trees on subsampled data and adding randomness in the variable selection as in Random Forests (Breiman, 2001) and then looking among all created leaf nodes for the most suitable node among all nodes created.

We generate 100 trees as in the Random Forests method: each is fit to subsampled training data using CART algorithm restricted to depth 4, and further randomness is injected by only permitting variables to be selected from a random subset of those available, for each tree. This takes on average between 90% to 110% of the computational time of a non-optimised pure `R` (R Development Core Team, 2005) implementation of Random Intersection Trees for these data. Note that this is when using the Fortran version of (Breiman, 2001) for the Random Forests node generation; we expect a significant speedup if Fortran or C code were used for Random Intersection Trees. We are currently working on such a version and plan to make it available soon. Furthermore, Random Forests would scale much worse if many more word-stems were included as variables.

Linear models. For linear models, we fit a sparse model with at most ℓ predictors (with $\ell \leq 4$), using a logistic model with an ℓ_1-penalty (Friedman et al., 2010; Tibshirani, 1996). We constrain

the regression coefficients to be positive since we are only looking for positive associations in the two previously discussed approaches, and want to keep the same interpretability for the linear model. For each value of $\ell \leq 4$, we take S_ℓ to be the set of variables with a positive regression coefficient. We select the largest value of ℓ such that the fraction of documents attaining the maximal value is at least $p_c/10$ and select the associated pattern S_ℓ. (An alternative approach would be to retain the documents with the highest predicted value when using a sparse regression fit. This approach gave very similar results.)

After screening the candidate patterns returned by each of the methods using (3.14) on all of the topics $c \in C$, we evaluate the misclassification rate $\mathbb{P}_n(c \notin Y | S \subseteq X)$ on the test data. The results for all of the topics are shown in Figure 3.4. The rules found with Random Intersection Trees have a smaller loss than those found with Random Forests in all but 5 of the topics. For those topics where Random Forests performs better, the difference in loss is typically small. Linear models achieve a smaller loss than Random Forests among most of the topics, but only have a smaller loss than Random Intersection Trees in 6 topics, performing worse in all remaining 46 topics.

3.6 Discussion

We have proposed Random Intersection Trees as an efficient way of finding interesting interactions. In contrast to more established algorithms, the patterns are not built up incrementally by adding variables to create interactions of greater and greater size. Instead we start from the full interaction $S = \{1, \ldots, p\}$ and remove more and more variables from this set by taking intersections with randomly chosen observations. Arranging the search in a tree increases efficiency by exploiting sparsity in the data. For the basic version of our method (Algorithm 1), we were able to derive a bound on the computational complexity. The bound depends on (a) the prevalence or frequency with which the pattern S appears among observations in class 1, and (b) the overall sparsity of the data, with higher sparsity making it easier to detect the interaction using a given computational budget. In the best case, we can achieve an almost linear complexity bound as a function of p; more generally our complexity bound typically has a smaller exponent than that for a brute force search. Further improvements can be made by using min-wise hashing techniques to terminate parts of the search (i.e. branches of the Intersection Tree) that have no chance of leading to interesting interactions. Numerical examples illustrate the improved interaction detection power of Random Intersection Trees over other tree-based methods and linear models.

There are many diverse ways in which interactions that solve (3.1) can be used in further analysis. The interactions may be of interest in their own right as shown in both numerical examples. One can also simply use the search to make sure that a dataset is unlikely to have strong interactions that could otherwise have been missed. If the aim is to build a classifier, they can be added to a linear model, or built into classifiers based on tree ensembles. For the latter approach one could consider, for example, averaging predictions in a linear way or averaging log-odds as in Random Ferns (Bosch et al., 2007). We believe developments along these lines will prove to be fruitful directions for future research. We also plan to generalise the idea to categorical and continuous predictor variables.

3.7 Appendix

Proof of Theorem 15. Fix a tree $t \in \{1, \ldots, T\}$ and suppose this has node set $N = \{1, \ldots, J\}$ indexed chronologically (see Section 3.2). For $d \in \{1, \ldots, D\}$, define

$$N_d = \{j \in N : \text{depth}(j) = d \text{ and } S_j \supseteq S\},$$
$$W_d = |N_d|.$$

Let E be the event that S is contained in S_1, the random sample selected for the root node of tree t. Further, let $G_d(u) = \mathbb{E}(u^{W_d}|E)$, the probability generating function of W_d conditional on the event E.

We make a few simple observations from the theory of branching processes. Firstly, for $d \leq D-1$, $G_{d+1} = G_d \circ G$ where $G := G_1$. To see this, first note that

$$W_{d+1} = \sum_{j \in N_d} \sum_{j' \in \text{ch}(j)} \mathbb{1}_{\{S \subseteq X_{i(j')}\}}.$$

Now conditional on E, the random variables $\sum_{j' \in \text{ch}(j)} \mathbb{1}_{\{S \subseteq X_{i(j')}\}}$ for $j \in N_d$, are independent of N_d. Moreover, they are independent of each other and have identical distributions equal to that of

$$\sum_{j' \in \text{ch}(1)} \mathbb{1}_{\{S \subseteq X_{i(j')}\}} = W_1.$$

This entails

$$\mathbb{E}(u^{W_{d+1}}|W_d = w, E) = \{\mathbb{E}(u^{W_1}|E)\}^w = \{G(u)\}^w.$$

Thus

$$G_{d+1}(u) = \mathbb{E}(\mathbb{E}(u^{W_{d+1}}|W_d, E)|E) = \mathbb{E}(\{G(u)\}^{W_d}|E) = G_d(G(u)),$$

as claimed.

From this we can conclude that if G has a fixed point q, then this must be a fixed point for all G_d. Since each G_d is non-decreasing on $(0, 1]$, we have that for all $d \in \mathbb{N}$, if $q' \leq q$ and $q' \in (0, 1]$, then $G_d(q') \leq q$. The relevance of these remarks will become clear from the following: for an $S' \in L_{D,t}$, we have

$$\begin{aligned} G_D(\mathbb{P}(S' \supsetneq S|S' \supseteq S)) &= \sum_{\ell=0}^{\infty} \mathbb{P}(W_D = \ell|E)\mathbb{P}(S' \supsetneq S|S' \supseteq S)^\ell \\ &= \sum_{\ell=0}^{\infty} \mathbb{P}(\{W_D = \ell\} \cap \{S \notin L_{D,t}\}|E) \\ &= \mathbb{P}(S \notin L_{D,t}|E). \end{aligned}$$

Thus if we can ensure that $\mathbb{P}(S' \supsetneq S|S' \supseteq S)$ is at most q, then the final probability in the above display will also be at most q. The rest of the proof proceeds with the following steps:

1. Find conditions on F_B, the distribution of the B_j, such that there exists a fixed point of G, q.
2. Find conditions on the tree depth D such that $\mathbb{P}(S' \supsetneq S|S' \supseteq S) \leq q$.
3. Given q establish conditions on T such that the overall probability of recovering S is at least

$1-\eta$.

4. Given F_B, D and T, compute the expected computational cost of the algorithm.

Step 1: Let the distribution of the B_j be such that

$$B_j = \begin{cases} b & \text{with probability } 1-\alpha, \\ b+1 & \text{with probability } \alpha. \end{cases}$$

Now given a $q \in (0,1]$, we shall pick $b \in \mathbb{Z}_+$ and $\alpha \in [0,1)$ to satisfy $G(q) = q$. To this end, observe that

$$\begin{aligned} G(q) &= (1-\alpha)(1-\theta_1(1-q))^b + \alpha(1-\theta_1(1-q))^{b+1} \\ &= [1-\{(\alpha+b)-\lfloor \alpha+b \rfloor\}\theta_1(1-q)]\{1-\theta_1(1-q)\}^{\lfloor \alpha+b \rfloor}. \end{aligned}$$

From the last displayed equation, we see that $G(q)$ varies with $\alpha+b$ continuously. Furthermore, when $\alpha+b=0$, $G(q)=1$, and by making $\alpha+b$ large, we can make $G(q)$ arbitrarily close to 0. Thus by the intermediate value theorem, for any $q \in (0,1]$, $\alpha+b$ can be chosen such that $G(q)=q$.

We now bound $\alpha+b$ from above in terms of q for use later in creating a bound on the complexity of the algorithm. We have

$$\begin{aligned} b+\alpha &= \frac{\log(q) - \log(1-\alpha\theta_1(1-q))}{\log(1-\theta_1(1-q))} + \alpha \\ &\leq \frac{-\log(q) + \log(1-\alpha\theta_1(1-q))}{\theta_1(1-q)} + \alpha \\ &\leq \frac{-\log(q)}{\theta_1(1-q)} \\ &\leq \frac{1+(1-q)/(2q)}{\theta_1}. \end{aligned} \tag{3.17}$$

In the final line, we used the inequality

$$\log(z) \geq (z-1) - \frac{(z-1)^2}{2z}, \quad 0 < z \leq 1.$$

Step 2: We now bound $\mathbb{P}(S' \supsetneq S | S' \supseteq S)$ from above in terms of D. The set S' is the intersection of $D+1$ observations selected independently of one another. In order for some $k \in S^c$ to be contained in S', it must have been present in all these $D+1$ observations. Thus by the union bound we have

$$\mathbb{P}(S' \supsetneq S | S' \supseteq S) \;\leq\; \sum_{k \in S^c} \mathbb{P}(k \in S' | S' \supseteq S) \;\leq\; p\nu^{D+1},$$

the rightmost inequality following from (A2).

To ensure this is at most q, we take

$$D = \left\lceil \frac{\log(p/q)}{\log(1/\nu)} \right\rceil - 1, \tag{3.18}$$

so

$$D \leq \frac{\log(p/q)}{\log(1/\nu)}. \tag{3.19}$$

Step 3: Turning now to the probability of recovering S, we have

$$\mathbb{P}(S \in L_D) = 1 - [1 - \{1 - \mathbb{P}(S \notin L_{D,t}|E)\}\theta_1]^T.$$

Given the choices of α and b (3.17), and D (3.18), we have that $\mathbb{P}(S \notin L_{D,t}|E) \leq q$. Thus taking T to be at least

$$\frac{-\log(\eta)}{(1-q)\theta_1} \geq \frac{-\log(\eta)}{\log\{1-(1-q)\theta_1\}} \tag{3.20}$$

guarantees recovery of S with probability at least $1-\eta$.

Step 4: To bound the complexity of the algorithm, observe that $\mathbb{E}(B_j) = b + \alpha$, so

$$\begin{aligned} C(T, D, F_B) \leq & \log(p) T \sum_{k=1}^{p} [(b+\alpha)\delta_k + \cdots + \{(b+\alpha)\delta_k\}^D] \\ & \leq \log(p) TD \left[p + \sum_{k:(b+\alpha)\delta_k > 1} \left\{ \left((b+\alpha)\delta_k\right)^D - 1 \right\} \right]. \end{aligned} \tag{3.21}$$

Substituting equations (3.17), (3.19) and (3.20) into the complexity bound (3.21), and writing $\epsilon = (1-q)/(2q)$ gives a bound for the computational complexity of

$$\log(p) \frac{\log(1/\eta)}{\theta_1} \frac{1+2\epsilon}{2\epsilon} \frac{\log\{p(1+2\epsilon)\}}{\log(1/\nu)} \left[p + \sum_{k:(1+\epsilon)\delta_k > \theta_1} \left\{ \left(p(1+2\epsilon)\right)^{\frac{\log\{(1+\epsilon)\delta_k/\theta_1\}}{\log(1/\nu)}} - 1 \right\} \right]. \tag{3.22}$$

Given that ϵ is bounded above, removing constant factors not depending on p, we get that the order of the computational complexity is bounded above by

$$\log(1/\eta) \frac{\log^2(p)}{\epsilon} \left\{ p + \sum_{k:(1+\epsilon)\delta_k > \theta_1} \left(p^{\frac{\log\{(1+\epsilon)\delta_k/\theta_1\}}{\log(1/\nu)}} - 1 \right) \right\}. \quad \square$$

Proof of Corollary 16. Note that

$$\sum_{k:(1+\epsilon)\delta_k > \theta_1} p^{\frac{\log((1+\epsilon)\delta_k/\theta_1)}{\log(1/\nu)}}$$

is bounded by

$$(1+\epsilon)^{\frac{\log(p)}{\log(1/\nu)}} \left(p^{\gamma} \cdot p^{\alpha^\star/\beta} \mathbb{1}_{\{\alpha^\star/\beta > 0\}} \; + \; p \cdot p^{\alpha_\star/\beta} \mathbb{1}_{\{\alpha_\star/\beta > 0\}} \right)$$

The result then follows from substituting into (3.22) and taking $\epsilon \propto 1/\log(p)$. $\square$

Derivation of (3.8)**.** Writing $r = n\pi_2(S)$, we have

$$\begin{aligned} \binom{n}{r} \mathbb{E}_\sigma(\min_{k \in S} m_\sigma(k)) &= \sum_{\ell=1}^{n-r+1} \ell \binom{n-\ell}{r-1} \\ &= \sum_{\ell=1}^{n-r+1} \left\{ (\ell-1) \binom{n-(\ell-1)}{r} - \ell \binom{n-\ell}{r} \right\} + \sum_{\ell=1}^{n-r+1} \binom{n-\ell+1}{r}. \end{aligned}$$

The first two terms sum to zero leaving only the final term. Thus

$$\begin{aligned}\binom{n}{r}\mathbb{E}_\sigma(\min_{k\in S} m_\sigma(k)) &= \sum_{\ell=1}^{n-r+1}\left\{\binom{n-\ell+2}{r+1}-\binom{n-\ell+1}{r+1}\right\}\\ &= \binom{n+1}{r+1},\end{aligned} \tag{3.23}$$

whence

$$\mathbb{E}_\sigma(\min_{k\in S} m_\sigma(k)) = \frac{n+1}{r+1}. \quad \square \tag{3.24}$$

Proof of Theorem 18. Writing

$$\tilde{\pi}_2^{-1}(L;S,M) := \tfrac{1}{L}\sum_{l=1}^{L}\min_{k\in S} M_{lk}$$

and suppressing dependence on S and M, we have

$$\begin{aligned}\hat{\pi}_1\hat{\pi}_2 - \pi_1\pi_2 &= \frac{(n+1-\tilde{\pi}_2^{-1})\hat{\pi}_1}{n\tilde{\pi}_2^{-1}} - \pi_1\pi_2\\ &= \frac{n+1-\tilde{\pi}_2^{-1}}{n\tilde{\pi}_2^{-1}}\left\{(\hat{\pi}_1-\pi_1) - \pi_1\frac{n\pi_2+1}{n+1-\tilde{\pi}_2^{-1}}\left(\tilde{\pi}_2^{-1}-\frac{n+1}{n\pi_2+1}\right)\right\}.\end{aligned} \tag{3.25}$$

Consider $L\to\infty$. By the weak law of large numbers and the continuous mapping theorem, we have

$$\begin{aligned}\frac{n+1-\tilde{\pi}_2^{-1}(L)}{n\tilde{\pi}_2^{-1}(L)} &\xrightarrow{p} \pi_2 \quad \text{and}\\ \frac{n\pi_2+1}{n+1-\tilde{\pi}_2^{-1}(L)} &\xrightarrow{p} \frac{(\pi_2+n^{-1})^2}{\pi_2(1+n^{-1})}.\end{aligned}$$

By the central limit theorem, Slutsky's lemma and Lemma 19,

$$\begin{aligned}A_L := \sqrt{L}(\hat{\pi}_1(L)-\pi_1) &\xrightarrow{d} N(0,\pi_1(1-\pi_1)) \quad \text{and}\\ B_L := -\pi_1\frac{n\pi_2+1}{n+1-\tilde{\pi}_2^{-1}(L)} \times \sqrt{L}\left(\tilde{\pi}_2^{-1}(L)-\frac{n+1}{n\pi_2+1}\right) &\xrightarrow{d} N(0,\pi_1^2(1-\pi_2)(1+\epsilon(n))),\end{aligned}$$

with $\epsilon(n)$ defined as in (3.12).

Define $I_S := \{i : S\subseteq X\}$ and let $k\in S$. Now observe that

$$\{\exists \iota' : m_\sigma(k)=\iota' \text{ for all } k'\in S\} = \{\sigma^{-1}(m_\sigma(k))\in I_S\} \text{ and } \{\min_{k\in S} m_\sigma(k)=\iota\}$$

are independent: in words, the distribution of $\min_{k\in S} m_\sigma(k)$ conditional on the fact that an observation index in I_S was permuted to a lower value than any in $I_k\setminus I_S$ is the same as its unconditional distribution. This implies the independence of $\hat{\pi}_1$ and $\tilde{\pi}_2^{-1}$ and thence also that of A_L and B_L. Thus we have that for all $u_1,u_2\in\mathbb{R}$,

$$\mathbb{E}(e^{i(u_1A_L+u_2B_L)}) = \mathbb{E}(e^{iu_1A_L})\mathbb{E}(e^{iu_2B_L}) \to \exp[\tfrac{1}{2}u_1^2\pi_1(1-\pi_1) + \tfrac{1}{2}u_2^2\{\pi_1^2(1-\pi_2)(1+\epsilon(n))\}].$$

pointwise as $L \to \infty$. Returning to (3.25), by Lévy's continuity theorem we have

$$\sqrt{L}\{\hat{\pi}_1(L)\hat{\pi}_2(L) - \pi_1\pi_2\} \xrightarrow{d} N(0, \pi_2^2\pi_1(1-\pi_1\pi_2)(1+\epsilon(n))). \quad \square$$

Lemma 19. *Let $r = n\pi_2(S)$ and suppose $n \geq r+2$. Then*

$$\mathrm{Var}_\sigma(\min_{k\in S} m_\sigma(k)) = \frac{r(n+1)(n-r)}{(r+1)^2(r+2)}.$$

Proof. We have,

$$\begin{aligned}
\binom{n}{r}\mathbb{E}_\sigma\{(\min_{k\in S} m_\sigma(k))^2\} &= \sum_{\ell=1}^{n-r+1} \ell^2 \binom{n-\ell}{r-1} \\
&= \sum_{\ell=1}^{n-r+1} \left\{(\ell-1)^2\binom{n-(\ell-1)}{r} - \ell^2\binom{n-\ell}{r}\right\} \\
&\quad + \sum_{\ell=1}^{n-r+1} \left\{2(\ell-1)\binom{n-(\ell-1)}{r} + \binom{n-\ell+1}{r}\right\} \\
&= 2\binom{n+1}{r+2} + \binom{n+1}{r+1},
\end{aligned}$$

where in the last line we used (3.23) and (3.24). Simplifying and using (3.8) then gives the result. $\square$

Chapter 4

Min-wise hashing for large-scale regression and classification with sparse data

4.1 Introduction

The modern field of high-dimensional statistics has now developed a powerful range of methods to deal with datasets where the number of variables p may greatly exceed the number of variables n. The prototypical example of microarray data, where p may be in the tens of thousands but n is typically not more than a few hundred, has motivated much of this development. Yet not all modern datasets come in this sort of shape and size. The emerging area of 'large-scale data' or the more vaguely defined 'Big Data' is a response to the increasing prevalence of computationally challenging datasets as arise in text analysis or web-scale prediction tasks, to give two examples. Here both n and p can run into the millions or more, particularly if interactions are considered.

In these 'large p, large n' regression scenarios, one can imagine situations where ordinary least squares (OLS) has competitive performance for prediction, but the sheer size of the data renders it computationally infeasible. Thus it makes sense, in this setting, to consider approximations to OLS that can be obtained at a lower computational cost.

Some effort has gone into studying approximations to least squares estimation by using low-rank matrix decompositions including CUR-type decompositions (Drineas et al., 2006, 2011). Here, however, as in the previous chapter, we are interested in developing methodology that can take advantage of sparsity in the design matrix, a property that is often present in large-scale data. This is not to be confused with signal sparsity, a common assumption in the high-dimensional context. Indeed, when the design matrix is sparse, having only a few variables that contribute to the response would make the expected response values of all observations with no non-zero entries for the important variables exactly the same; one expects that such a property would not be possessed by many datasets. However, similarly to the way in which many high-dimensional techniques exploit sparsity to improve statistical efficiency, here we aim to use sparsity in the data to yield both computational and statistical improvements.

The approach we take uses dimensionality reduction: we first map each of the p-dimensional observations to an L-dimensional space, where we would typically choose $L \ll p$, and then perform

regression on the reduced design matrix thus created. Provided L is small enough, this final regression will be computationally feasible. Our dimensionality reduction step is based on a min-wise hashing scheme (see Section 3.4), and is a modification of b-bit min-wise hashing (Li and König, 2011). The latter method can be applied to compress binary design matrices and has been used with SVM-type classifiers (see Li et al. (2011); Yu et al. (2012)). Despite its promising empirical results, the theoretical properties of the predictions obtained following a reduction by b-bit min-wise hashing have not been thoroughly explored for linear or logistic regression. Our variant, which we call '**m**in-wise hash **r**andom-**s**ign mapping' (MRS mapping), is more analytically tractable than b-bit min-wise hashing, and has the additional benefit that it can be used on design matrices where some predictors are continuous. In addition, our scheme allows the construction of variable importance measures that can allow one to assess the influence of the individual variables on the predictions.

We describe the MRS mapping algorithm in Section 4.2. In Section 4.3 we study the performance of linear and logistic regression using the reduced design matrix created. We show, in particular, that if the original data are well-approximated by a linear model with coefficient vector $\boldsymbol{\beta}^*$, the expected mean-squared prediction error is bounded by a small constant times $\sqrt{q/n}\,\|\boldsymbol{\beta}^*\|_2$, where q is the maximal number of active variables for each observation.

Perhaps more surprisingly, we show in Section 4.4 that despite the information loss through MRS mapping, a main effects model in the reduced design matrix can also approximate an interaction model in the original data. This does not require a modification of the procedure, though one typically needs a larger dimensionality L of the mapping, to reduce the error in the approximation. Variable importance measures and other extensions are discussed in Section 4.5, after which some numerical studies are presented in Section 4.6. We conclude with a discussion in Section 4.7, and all proofs are collected in the appendix of this chapter.

4.2 MRS mapping and dimension reduction

In this section, we present our MRS mapping methodology for dimension reduction. Given a sparse design matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$, the aim is to map this to a compressed matrix $\mathbf{S} \in \mathbb{R}^{n\times L}$, in a way that is computationally efficient and such that linear combinations of variables in $\mathbf{X}$ can be well-approximated by linear combinations of columns of $\mathbf{S}$. Section 4.2.2 describes the mapping to $\mathbf{S}$. The construction may seem rather bizarre at first sight; indeed, our initial motivation for developing MRS mapping was to emulate the equally mysterious b-bit min-wise hashing scheme (Li and König, 2011) in an analytically tractable manner. We explain the connection between the two techniques in Section 4.2.3. In Section 4.2.4 we briefly discuss two other dimension reduction schemes that one might consider applying to $\mathbf{X}$, namely principal components and random projections. We begin by establishing some notation.

4.2.1 Notation

Given a matrix $\mathbf{U}$, we will write $\mathbf{u}_i$ and $\mathbf{U}_j$ for the i^{th} row and j^{th} column respectively, where both are to be regarded as column vectors. The ij^{th} entry will be denoted U_{ij}. For a set of indices of columns of $\mathbf{U}$, A, we will write $\mathbf{U}_A$ for the submatrix consisting of the columns of $\mathbf{U}$ indexed by

A. A vector of 1's will be denoted $\mathbf{1}$. We also define the following matrix norms

$$\|\mathbf{U}\|_\infty := \max_{i,j} |U_{ij}|$$
$$\|\mathbf{U}\|_F := \left(\sum_{i,j} U_{ij}^2 \right)^{1/2}.$$

When the parentheses following probability and expectation signs, $\mathbb{P}$ and $\mathbb{E}$, enclose multiple potential sources of randomness, we will sometimes add subscripts to indicate what is being considered as random. For example, if U and V are random variables, we may write $\mathbb{E}_U(U|V)$ for the conditional expectation of U given V, and $\mathbb{E}_{U,V}(U+V)$ for the expected value of $U+V$.

4.2.2 Construction of S

Here we describe how the l^{th} column of $\mathbf{S}$, $\mathbf{S}_l$, is generated. We will create L columns in total, the entire collection of columns forming a set of i.i.d. random vectors. First let $\mathbf{\Psi} \in \{-1, 1\}^{p \times L}$ be a matrix consisting of i.i.d. random signs, each chosen with probability $1/2$. There are three steps to the construction:

Step 1: Generate a random permutation of the set $\{1, \ldots, p\}$, π_l, and permute the columns of $\mathbf{X}$ according to this permutation.

Step 2: Search along each row of the permuted design matrix (in order of increasing column index) and record in the vector $\mathbf{H}_l \in \mathbb{N}^n$, the indices of the variables (indexed as in the original design matrix) with the first non-zero value.

Step 3: Form $\mathbf{S}_l \in \mathbb{R}^n$ with i^{th} component given by $\mathbf{\Psi}_{H_{il}l} X_{iH_{il}}$.

This construction is illustrated for a toy example in Table 4.1.

$\mathbf{X} = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{pmatrix} \cdot & 7 & \cdot & 9 \\ \cdot & \cdot & 1 & 4 \\ 1 & \cdot & 2 & \cdot \\ \cdot & 6 & 1 & \cdot \\ 8 & 5 & \cdot & \cdot \end{pmatrix} \end{array} \overset{\pi_l = 2314}{\mapsto} \begin{array}{c} \begin{matrix} 3 & 1 & 2 & 4 \end{matrix} \\ \begin{pmatrix} \cdot & \cdot & \mathbf{7} & 9 \\ \mathbf{1} & \cdot & \cdot & 4 \\ \mathbf{2} & 1 & \cdot & \cdot \\ \mathbf{1} & \cdot & 6 & \cdot \\ \cdot & \mathbf{8} & 5 & \cdot \end{pmatrix} \end{array}$	$\mathbf{H}_l = \begin{pmatrix} 2 \\ 3 \\ 3 \\ 3 \\ 1 \end{pmatrix}$	$\mathbf{\Psi}_l = \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \Rightarrow \mathbf{S}_l = \begin{pmatrix} -7 \\ -1 \\ -2 \\ -1 \\ 8 \end{pmatrix}$
Step 1: non-zero indices whose variable indices will appear in $\mathbf{H}_l$ in Step 2 are in bold.	*Step 2.*	*Step 3.*

Table 4.1: Steps 1–3 applied to a toy example. Dots represent zeroes.

Let $\mathbf{z}_i = \{k : X_{ik} \neq 0\}$ be the set of variable indices whose entries entries have non-zero values for the i^{th} observation. Performing the steps above for all $1 \leq l \leq L$, we get $n \times L$ matrices $\mathbf{H}$, and $\mathbf{S}$ given by

$$H_{il} = \arg\min_{k \in \mathbf{z}_i} \pi_l(k), \tag{4.1}$$
$$S_{il} = \Psi_{H_{il}l} X_{iH_{il}}. \tag{4.2}$$

Suppose $\mathbf{Y}$ is a vector of responses associated with $\mathbf{X}$. Having created matrix $\mathbf{S}$, we now regress $\mathbf{Y}$ on $\mathbf{S}$, rather than the original $\mathbf{X}$. For example, we may perform a linear regression,

$$(\hat{\alpha}, \hat{\mathbf{b}}) = \underset{(\alpha,\mathbf{b})\in\mathbb{R}\times D_\lambda}{\arg\min} \|\mathbf{Y} - \alpha\mathbf{1} - \mathbf{S}\mathbf{b}\|_2^2,$$

where $D_\lambda = \mathbb{R}^L$ for OLS or, $D_\lambda = \{\mathbf{w} \in \mathbb{R}^L : \|\mathbf{w}\|_2 \leq \lambda\}$ for ridge regression and a given value of the tuning parameter λ.

The estimator $\hat{\mathbf{b}}$ can then be used for predicting the expected responses of the existing observations based on $\mathbf{S}\hat{\mathbf{b}}$. For new observations, a corresponding new $\mathbf{S}$ matrix must be created. To create the $\mathbf{S}$ matrix for new observations we have to use the same permutations and sign-matrix $\mathbf{\Psi}$ as for the original data.

Note that one would not necessarily follow the above steps when implementing the MRS mapping algorithm. In practice, one would not store the entire matrix of signs nor all the random permutations. In an implementation, hash functions (Carter and Wegman, 1979) would be used to create the matrix $\mathbf{S}$ deterministically, though it is beyond the scope of this work to go into the details; see Li et al. (2012b) for more information. With this approach, $\mathbf{S}$ would be created row-by-row, and only a single observation from $\mathbf{X}$ would need to be kept in memory at any one time. Furthermore, many rows could be created in parallel. Other ideas such as one-permutation hashing (Li et al., 2012a) can also be used to speed up the pre-processing step. For the theoretical analysis in Section 4.3.1 and following, we will assume that all random inputs in the construction of $\mathbf{S}$ are truly random.

4.2.3 Connection to b-bit min-wise hashing

Instead of creating the matrix $\mathbf{H}$, b-bit min-wise hashing (Li and König, 2011) calculates a matrix $\mathbf{M} \in \mathbb{N}^{n\times L}$ with entries given by

$$M_{il} = \min_{k\in\mathbf{z}_i} \pi_l(k). \tag{4.3}$$

Using this, a matrix $\tilde{\mathbf{M}} \in \mathbb{N}^{n\times L}$ is created whose entries contain the numeric values of the lowest b bits of the entries in $\mathbf{M}$ when written out in binary form (i.e. the values of the remainders when dividing each entry by 2^b). Finally, each column of $\tilde{\mathbf{M}}$ is expanded into a block of 2^b columns, where each column codes for each of the 2^b possible values. This creates a matrix $\mathbf{T} \in \{0,1\}^{n\times 2^b L}$ with which one can perform regression. Note that no information about the values of the entries in $\mathbf{X}$ is used other than whether or not they are zero. Thus for design matrices with real-valued entries, some form of quantisation must be performed first.

When $\mathbf{X}$ is binary, MRS mapping is perhaps most closely related to b-bit min-wise hashing with $b = 1$, since the last bit of the value M_{il} can be considered to be a close approximation to a random sign entry. Indeed, in the generation of each column of the final compressed design matrix, both MRS mapping and 1-bit min-wise hashing divide the observations into two groups with membership determined by random signs for MRS mapping and by the parity of the entries in $\mathbf{M}$ for 1-bit min-wise hashing. If for a particular l, two observations, i and i', have $M_{il} = M_{i'l}$, or equivalently $H_{il} = H_{i'l}$, they will be in the same group under both schemes; if not, their assignments to the two groups will be completely random with MRS-mapping, and approximately random for 1-bit min-wise hashing.

Provided one includes an unpenalised intercept term in the regression using $\mathbf{T}$, the fitted values

from regression on $\mathbf{T}$ will be the same as when the binary entries in $\mathbf{T}$ are transformed such that all zeroes take the value -1. Thus predictions from MRS mapping should be fairly close to those of 1-bit min-wise hashing.

One could attempt to mimic b-bit min-wise hashing for $b > 1$ by taking b random sign-assignments along with each permutation. The expansion used to code for the different sequences of b signs that can be taken can be thought of as adding interactions of all order between each block of b columns. However, we do not investigate this further here.

We do not make the claim that MRS mapping performs better when $\mathbf{X}$ is binary. Rather, the random signs used in the former method help to expose the mechanism at work in both schemes that make them successful under certain circumstances. In addition, using random signs allows us to deal with continuous predictors, and makes it easier to compute the variable importance measures to be described in Section 4.5.2.

4.2.4 Principal components and random projections

Performing principal component analysis (Jolliffe, 1986) (PCA) and retaining only the fist L components is a popular and effective form of dimension reduction. One drawback in the large-scale data setting is that computing the principal components can be computationally demanding. On the other hand, the generation of $\mathbf{S}$ with MRS mapping has complexity of order L times the number of non-zero entries in $\mathbf{X}$, and thus is almost the best one can hope for.

The method of random projections, motivated by the celebrated Johnson–Lindenstrauss lemma (Johnson and Lindenstrauss, 1984), offers dimension reduction at a similar computational cost to MRS mapping. In this scheme, $\mathbf{X}$ is mapped to $\mathbf{XA}$, where $\mathbf{A}$ is a $p \times L$ matrix with random entries typically chosen to be i.i.d. Gaussians. We compare MRS mapping and random projections from a theoretical perspective in Section 4.3.2 and empirically in Section 4.6.2.

One advantage that MRS mapping has over both PCA and random projections, is that interactions between the original predictors in $\mathbf{X}$ can be captured when only performing regression using the main effects in $\mathbf{S}$, as shown in Section 4.4. In contrast, a regression using main effects in the reduced design matrices obtained through PCA or random projections must necessarily fit a model no more complex than a main effects model in the original data.

4.3 Main effect models

In this section, we present results that bound the expected prediction error when performing regression on the reduced design matrix $\mathbf{S}$ in the contexts of the linear and logistic regression models. Here we assume that the mean response depends on $\mathbf{X}\boldsymbol{\beta}^*$ where $\boldsymbol{\beta}^* \in \mathbb{R}^p$ is a vector of coefficients. Models containing interactions will be treated in Section 4.4.

The first step in obtaining these results is to construct a vector $\mathbf{b}^* \in \mathbb{R}^L$ such that $\mathbf{Sb}^*$ is close to $\mathbf{X}\boldsymbol{\beta}^*$, on average. We address this in the next section.

4.3.1 Approximation error

Assume that the number of non-zero entries in each row of $\mathbf{X} \in \mathbb{R}^{n \times p}$ is $q \geq 1$. The simple extension to unequal row sparsity will be dealt with below.

Theorem 20. *Let* $\mathbf{b}^* \in \mathbb{R}^L$ *be defined by*

$$b_l^* := \frac{q}{L} \sum_{k=1}^{p} \beta_k^* \Psi_{kl} w_{\pi_l(k)},$$

where $\mathbf{w} \in \mathbb{R}^p$ *is a vector of weights. Then there exists a choice of weight vector, such that* $\mathbf{b}^*$ *has the following properties.*

(i) The approximation is unbiased: $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\mathbf{S}\mathbf{b}^*) = \mathbf{X}\boldsymbol{\beta}^*$.

(ii) $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2) \leq (2 - q/p)q\|\boldsymbol{\beta}^*\|_2^2/L$.

(iii) If $\|\mathbf{X}\|_\infty \leq 1$, *then* $\frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{X}\boldsymbol{\beta}^*\|_2^2) \leq \mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2)$.

(iv) In general,

$$\frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{X}\boldsymbol{\beta}^*\|_2^2) \leq \frac{2 - q/p}{Ln}\left(\|\mathbf{X}\|_F^2\|\boldsymbol{\beta}^*\|_2^2 + q\sum_{k=1}^{p}\|\mathbf{X}_k\|_2^2\beta_k^{*2}\right).$$

The theorem above shows that there exists a vector $\mathbf{b}^*$ that furnishes two bounds on the approximation error of $\mathbf{X}\boldsymbol{\beta}^*$. The version in (iii) holds when $\|\mathbf{X}\|_\infty \leq 1$. The more general bound in (iv) is often tighter depending on the distribution of the entries in $\mathbf{X}$, though a little more complicated. In the results on prediction to follow, we will use the simpler bound (iii) assuming that $\|\mathbf{X}\|_\infty \leq 1$. However, in all of these results, we could equally well use (iv) to give bounds that are valid when the predictor variables are not necessarily bounded. For example, in Theorem 21 we can achieve this by replacing $\sqrt{q}\|\boldsymbol{\beta}^*\|_2$ by $\left(\|\mathbf{X}\|_F^2\|\boldsymbol{\beta}^*\|_2^2 + q\sum_{k=1}^{p}\|\mathbf{X}_k\|_2^2\beta_k^{*2}\right)^{1/2}$.

One might initially think of simply approximating $\mathbf{X}\boldsymbol{\beta}^*$ by its projection on to $\mathbf{S}$. However, it is not clear how to obtain a bound on the approximation error of the type in (iii) or (iv) directly. On the other hand, the relatively simple form of $\mathbf{b}^*$ gives a bound on the approximation error through relatively elementary calculations.

Another useful property of $\mathbf{b}^*$, aside from the approximation accuracy it delivers, is given in (ii): on average, $\|\mathbf{b}^*\|_2^2$ is small when L is large. In addition, the fact that the components of $\mathbf{b}^*$ are i.i.d. entails that $\|\mathbf{b}^*\|_2^2$ concentrates around its expected value with high probability (see Lemma 28 in the Appendix of this chapter). This property will be useful when studying the application of ridge regression to $\mathbf{S}$.

We finally comment on the issue of unequal row sparsity; a simple solution to the problem is as follows. Let q_i be the number of non-zero entries in $\mathbf{x}_i$. Now form an augmented design matrix $\tilde{\mathbf{X}}$ by adding $\max_i q_i - \min_i q_i$ 'dummy' variables to $\mathbf{X}$, all of whose non-zero values are equal to a nominal value ξ and are arranged in such a way that $|\{k : \tilde{\mathbf{x}}_i \neq 0\}| = \max_i q_i$. Here ξ is to be thought of as an arbitrarily small non-zero value so that the indices of the dummy variables can be chosen as entries in the matrix $\mathbf{H}$. The value of ξ is set to zero after $\mathbf{S}$ has been created, so if H_{il} is the index of a dummy variable, then entry S_{il} will vanish. In this way, we can and will assume that there are exactly q non-zero entries in each row, where $q = \max_i q_i$.

In practice, one does not usually need to consider $\tilde{\mathbf{X}}$ as regressing on the original $\mathbf{S}$ tends to produce a vector of coefficients $\hat{\mathbf{b}}$ that adapts reasonably well to unequal row sparsity, as long as the variation in row sparsity across observations is moderate.

4.3.2 Linear regression models

Assume we have the following approximately linear model:

$$\mathbf{Y} = \alpha^* \mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\gamma} + \boldsymbol{\varepsilon}. \tag{4.4}$$

Here α^* is the intercept and $\mathbf{X}$ is a sparse $n \times p$ design matrix with entries in $[-1, 1]$, of which q are non-zero in each row. See the comments after Theorem 20 for how to obtain results without the restriction $\|\mathbf{X}\|_\infty \leq 1$. We assume that the random noise $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ satisfies $\mathbb{E}_{\boldsymbol{\varepsilon}}(\varepsilon_i) = 0$, $\mathbb{E}_{\boldsymbol{\varepsilon}}(\varepsilon_i^2) = \sigma^2$ and $\mathrm{Cov}_{\boldsymbol{\varepsilon}}(\varepsilon_i, \varepsilon_j) = 0$. The vector $\boldsymbol{\gamma} \in \mathbb{R}^n$ represents deterministic structural error. We do not require that $\boldsymbol{\gamma}$ be orthogonal to the column space of $\mathbf{X}$. Our bounds on prediction error involve $\|\boldsymbol{\gamma}\|_2^2$ and $\|\boldsymbol{\beta}^*\|_2$, so if a $\boldsymbol{\beta}^*$ is available with low ℓ_2-norm which satisfies (4.4) at the expense of a small increase in $\|\boldsymbol{\gamma}\|_2^2$, it is to be preferred. However, without loss of generality, we will demand that $\boldsymbol{\gamma}^T \mathbf{1} = 0$.

Our results here give bounds on a mean-squared prediction error (MSPE) of the form

$$\mathrm{MSPE}((\hat{\alpha}, \hat{\mathbf{b}})) := \mathbb{E}_{\boldsymbol{\varepsilon}, \boldsymbol{\pi}, \boldsymbol{\Psi}} \Big(\|\alpha^* \mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}}\|_2^2 \Big) / n. \tag{4.5}$$

Thus we consider a denoising-type or in-sample error: the error on the data used to fit the regression coefficients. Bounds on the prediction error at new observations would require conditions on the distribution of observations and we have avoided making any such assumptions for the results here.

4.3.2.1 Ordinary least squares

Perhaps the simplest way to estimate the linear model is to apply a least squares estimator,

$$(\hat{\alpha}, \hat{\mathbf{b}}) := \operatorname*{arg\,min}_{(\alpha, \mathbf{b}) \in \mathbb{R} \times \mathbb{R}^L} \|\mathbf{Y} - \alpha \mathbf{1} - \mathbf{S}\mathbf{b}\|_2^2, \tag{4.6}$$

to the matrix $\mathbf{S}$, assuming that $L \in \mathbb{N}$ is smaller than the number of samples n. We have the following theorem.

Theorem 21. *Let $(\hat{\alpha}, \hat{\mathbf{b}})$ be the least squares estimator (4.6) and let $L^* := \sqrt{(2 - q/p)qn}\|\boldsymbol{\beta}^*\|_2/\sigma$. We have*

$$\mathrm{MSPE}((\hat{\alpha}, \hat{\mathbf{b}})) \;\leq\; 2\sqrt{2 - \frac{q}{p}} \max\left\{ \frac{L}{L^*}, \frac{L^*}{L} \right\} \sigma \sqrt{\frac{q}{n}} \|\boldsymbol{\beta}^*\|_2 \;+\; \frac{\|\boldsymbol{\gamma}\|_2^2}{n} + \frac{\sigma^2}{n}.$$

Considering the case where $\boldsymbol{\gamma} = \mathbf{0}$, the result for an optimal choice of $L \approx L^*$ implies that the MSPE is of order $\sigma\sqrt{q/n}\|\boldsymbol{\beta}^*\|_2$. The slow rate in n, which stems from a balance between the approximation error of order $q\|\boldsymbol{\beta}^*\|_2^2/L$ and an estimation error of order $\sigma^2 L/n$, seems unavoidable if we do not make stronger conditions on the design. Indeed, essentially the same error rate is obtained in Theorem 21 of Maillard and Munos (2012) for OLS following dimension reduction by random projections, though there an extra $\log(n)$ factor is incurred. Furthermore the optimal number of projections in that case is of the same order as that of L^* for MRS mapping here.

To better understand the implications of Theorem 21, it is helpful to assume $\alpha^* = 0$ and fix the size of the signal so that $\|\mathbf{X}\boldsymbol{\beta}^*\|_2^2/n = 1$, and look at a few special cases. Consider the random design setting with independent predictors with roughly equal sparsity and which sum to roughly zero. In this case, $\|\boldsymbol{\beta}^*\|_2$ will be of order $\sqrt{p/q}$ and the MSPE will vanish if $p/n \to 0$.

The method becomes more attractive if the signal is associated with directions of $\mathbf{X}$ with larger variance. For example, suppose the variables can be partitioned into B blocks of variables $I_1, \ldots, I_B$

such that within the blocks, predictors are independent but the contributions $\mathbf{s}_b = \mathbf{X}_{I_b}\boldsymbol{\beta}^*_{I_b}$ to the signal of each block $b = 1, \ldots, B$ have positive correlations of at least a constant $\rho > 0$. Suppose further that the ratios of the ℓ_2-norms of the $\mathbf{s}_b$ are bounded and all predictors have roughly equal sparsity. The signals $\|\mathbf{s}_b\|_2^2/n$ then scale as $1/(B^2\rho)$ and the ℓ_2-norm $\|\boldsymbol{\beta}^*\|_2$ will scale as $\sqrt{p/(\rho Bq)}$, making the predictions converge to the true signal as long as $(p/B)/n \to 0$. We see that the overall number of blocks is not relevant to the success of the scheme, and all that matters is the average number of variables within a block, p/B.

We can also consider sparsity in the signal: if the number of important variables is fixed, the ℓ_2-norm of the optimal regression vector will remain a constant if we add additional variables, whilst q and p increase. All that is required for consistency is $q/n \to 0$.

An interesting scenario is one of increasing variable sparseness. In many applications, the more predictor variables are added the sparser they tend to become. In text analysis, the first block of predictor variables might encode the presence of individual words. The next block might code for bigrams and the following, higher order N-grams. With this design, predictor variables in each successive block become sparser than the previous. It is then interesting to consider how much the MSPE can increase if we add a block with many sparse variables which contain no additional signal contribution. The result above indicates that the MSPE only increases as $\sqrt{q}$ since the norm of $\|\boldsymbol{\beta}^*\|_2$ would be constant in this case. Adding a block of several million (sparse) bigrams might thus have the same statistical effect as adding several thousand (denser) unigrams (individual words).

If we assume $n = O(q)$, which is all that would be required to keep the prediction error bounded asymptotically, then in this situation, we see that $L^* = O(q)$. This could be a substantial reduction over the original dimension of the data, p, and would result in a corresponding large reduction in the computational cost of regression. Indeed, ridge regression or the LAR algorithm (Efron et al., 2004) applied to $\mathbf{X}$ would have complexity $O(q^2p)$, and one would expect that the Lasso (Tibshirani, 1996) would have similar computational cost. In contrast OLS applied to $\mathbf{S}$ would only require $O(q^3)$ operations, an improvement of q/p.

The discussion above considered an optimal choice of $L \approx L^*$. Even if we cannot afford to work with the optimal dimension L^* for computational reasons, the bound will still be useful for smaller values of L. The guarantee on prediction accuracy afforded by MRS mapping could not be obtained if, for example, a random subset of the predictors were chosen and remaining ones discarded, or SIS (Fan and Lv, 2008) was used; the latter would require much stronger conditions on the design and signal to be met for it to work well.

4.3.2.2 Ridge regression

Instead of using a least-squares estimator on the transformed data matrix $\mathbf{S}$ we can also apply ridge regression (Hoerl and Kennard, 1970). Here we will not require $L \leq n - 1$ and so a higher-dimensional MRS mapping can be used to create $\mathbf{S}$. This will be especially useful when fitting interaction models where a larger choice of L is needed (see Section 4.4).

For a given $\eta > 0$, the regression coefficients are found by

$$(\hat{\alpha}, \hat{\mathbf{b}}^\eta) := \underset{(\alpha,\mathbf{b})\in\mathbb{R}\times\mathbb{R}^L}{\arg\min} \|\mathbf{Y} - \hat{\alpha}\mathbf{1} - \mathbf{S}\mathbf{b}\|_2^2 \text{ such that } \|\mathbf{b}\|_2^2 \leq (1+\eta)\frac{(2-q/p)q\|\boldsymbol{\beta}^*\|_2^2}{L}. \tag{4.7}$$

The theorem below gives a bound on the MSPE of $(\hat{\alpha}, \hat{\mathbf{b}}^\eta)$.

Theorem 22. *Let L^* be defined as in Theorem 21. We have*

$$\text{MSPE}((\hat{\alpha}, \hat{\mathbf{b}}^\eta)) \;\le\; \sqrt{(2-q/p)q}\,\|\boldsymbol{\beta}^*\|_2 \left(\frac{2\sigma\sqrt{1+\eta}+(L^*/L)}{\sqrt{n}}\right)+\rho\frac{\|\mathbf{X}\boldsymbol{\beta}^* - \overline{\mathbf{X}\boldsymbol{\beta}^*}\mathbf{1}+\boldsymbol{\gamma}\|_2^2}{n}+\frac{\|\boldsymbol{\gamma}\|_2^2}{n}+\frac{\sigma^2}{n},$$

where

$$\rho := \exp\left(-\frac{L\eta^2}{18(2-q/p)q\{18(2-q/p)+\eta\}}\right), \tag{4.8}$$

and

$$\overline{\mathbf{X}\boldsymbol{\beta}^*} := \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i^T\boldsymbol{\beta}^*.$$

The ridge regression result above is very similar to that for OLS with the leading term of $\sigma\|\boldsymbol{\beta}^*\|_2\sqrt{q/n}$ being identical in both settings. The main difference is the following: achieving a good prediction error with OLS hinges on a careful choice of L. In contrast, with ridge regression, L can (and should) be chosen very large, from a purely statistical point of view. However, the constraint on the ℓ_2-norm of $\hat{\mathbf{b}}$ needs to be chosen carefully with ridge regression. The tuning parameter thus simply appears in a different place.

4.3.3 Logistic regression

We give an analogous result to Theorem 22 for classification problems under logistic loss. Let $\mathbf{X} \in [-1,1]^{n\times p}$ be the design matrix of predictor variables and let $\mathbf{Y} \in \{0,1\}^n$ be an associated vector of class labels. We assume the model

$$Y_i \sim \text{Bernoulli}(p_i); \qquad \log\left(\frac{p_i}{1-p_i}\right) = \mathbf{x}_i^T\boldsymbol{\beta}^* + \gamma_i, \tag{4.9}$$

with the Y_i independent for $1 \le i \le n$. The vector $\boldsymbol{\gamma} \in \mathbb{R}^n$ represents structural error. We have omitted the separate intercept term for simplicity, and we do not require any orthogonality conditions on $\boldsymbol{\gamma}$ or $\mathbf{X}$.

Here we consider a linear classifier constructed by ℓ_2-constrained logistic regression. One can obtain a similar result for unconstrained logistic regression based on Lemma 6.6 of Bühlmann and van de Geer (2011a), but we do not pursue this further here. Define

$$\hat{\mathbf{b}}^\eta = \arg\min_{\mathbf{b}} \frac{1}{n}\sum_{i=1}^n \left[-Y_i\mathbf{s}_i^T\mathbf{b} + \log\{1+\exp(\mathbf{s}_i^T\mathbf{b})\}\right] \;\text{ such that }\; \|\mathbf{b}\|_2^2 \le (1+\eta)\frac{(2-q/p)q\|\boldsymbol{\beta}^*\|_2^2}{L}. \tag{4.10}$$

Let $\mathcal{E}(\hat{\mathbf{b}}^\eta)$ denote the excess risk of $\hat{\mathbf{b}}^\eta$ under logistic loss, so

$$\mathcal{E}(\hat{\mathbf{b}}^\eta) = \frac{1}{n}\sum_{i=1}^n \left[-p_i\mathbf{s}_i^T\hat{\mathbf{b}}^\eta + \log\{1+\exp(\mathbf{s}_i^T\hat{\mathbf{b}}^\eta)\}\right] - \frac{1}{n}\sum_{i=1}^n \left[-p_i\mathbf{x}_i^T\boldsymbol{\beta}^* + \log\{1+\exp(\mathbf{x}_i^T\boldsymbol{\beta}^*+\gamma_i)\}\right]. \tag{4.11}$$

We can now state the analogous result to Theorem 22.

Theorem 23. *Define $\tilde{p} \in \mathbb{R}$ by*

$$\tilde{p} := \frac{1}{n}\sum_{i=1}^n p_i(1-p_i) \le \frac{1}{2}. \tag{4.12}$$

Then we have

$$\mathbb{E}_{\mathbf{Y},\boldsymbol{\pi},\boldsymbol{\Psi}}\{\mathcal{E}(\hat{\mathbf{b}}^{\eta})\} \leq \sqrt{(2-q/p)q}\,\|\boldsymbol{\beta}^*\|_2 \left(\frac{\sqrt{(1+\eta)\tilde{p}} + L^*/(4L)}{\sqrt{n}}\right) + \log(2)\rho,$$

where ρ and L^* are defined as in (4.8) *and Theorem 21 respectively.*

The result illustrates that the usefulness of MRS mapping is not limited to regression problems. In fact, most applications of b-bit min-wise hashing are classification problems (Li and König, 2011) and our analysis of MRS mapping here gives a theoretical explanation for its performance in these cases.

4.4 Interaction models

One of the compelling aspects of regression and classification with MRS mapping is the fact that a particular form of interactions between variables can be fitted. This does not require any change in the procedure other than a possible increase in L, the dimension of the MRS mapping. To be clear, in order to capture interactions with MRS mapping, just as in the main effects case, we create a reduced matrix $\mathbf{S}$ and then fit a main effects model to $\mathbf{S}$. The dimension of the compressed data, L, can still be substantially smaller than the $O(p^2)$ number of coefficients that would need to be estimated if the interactions were modelled in the conventional way, and so the resulting computational advantage can be very large.

Note that in situations where the number of original predictors, p, may be manageable, including interactions explicitly can quickly become computationally infeasible. For example, if we start with, 10^5 variables, the two-way interactions number more than a billion. For larger values of p, even methods such as Random Forest (Breiman, 2001) or Rule Ensembles (Friedman and Popescu, 2008) would suffer similar computational problems.

We now describe the type of interaction model that can be fitted with MRS mapping. Let $\mathbf{f}^* \in \mathbb{R}^n$ be given by

$$f_i^* = \sum_{k=1}^{p} X_{ik}\theta_k^{*,(1)} + \sum_{k,k_1=1}^{p} X_{ik}\mathbb{1}_{\{X_{ik_1}=0\}}\Theta_{k,k_1}^{*,(2)}, \quad i = 1,\ldots,n, \tag{4.13}$$

where $\boldsymbol{\theta}^{*,(1)} \in \mathbb{R}^p$ is a vector of coefficients for the main effects terms, and $\boldsymbol{\Theta}^{*,(2)} \in \mathbb{R}^{p\times p}$ is a matrix of coefficients for interactions whose diagonal entries are zero. Throughout this section we will assume that $\|\mathbf{X}\|_\infty \leq 1$. Note that if $\mathbf{X}$ were a binary matrix, then (4.13) parametrises (in fact over-parametrises) all linear combinations of bivariate functions of predictors; that is all possible two-way interactions are included in the model.

In general, the interaction model includes the tensor product of the set of original variables with the columns of an $n \times p$ matrix with ik^{th} entry $\mathbb{1}_{\{X_{ik}=0\}}$. The value zero is thus given a special status and the model seems particularly appropriate in the sparse design setting we are considering here.

Now let $\boldsymbol{\Theta}^*$ collect together $\boldsymbol{\theta}^{*,(1)}$ and $\boldsymbol{\Theta}^{*,(2)}$ so that we may define

$$\ell(\boldsymbol{\Theta}^*) := \|\boldsymbol{\theta}^{*,(1)}\|_2 + \left(2(2-q/p)q \sum_{k,k_1,k_2} \left|\Theta_{kk_1}^{*,(2)}\Theta_{kk_2}^{*,(2)}\right|\right)^{1/2}. \tag{4.14}$$

We will show that the results for main effects models with MRS mapping can be transferred to interaction models if we replace $\|\boldsymbol{\beta}^*\|_2$ with $\ell(\boldsymbol{\Theta}^*)$. We proceed exactly as before, first bounding the approximation error and then using this to control the prediction error.

4.4.1 Approximation error

As in the main effects case, we assume that the number of non-zero entries in each row of $\mathbf{X} \in [-1,1]^{n\times p}$ is $q \geq 1$; see the comments in Section 4.3.1 for how imbalanced row sparsity can be dealt with. Furthermore, for technical reasons, we assume here that $p \geq 3$.

Theorem 24. *Let* $\mathbf{b}^* \in \mathbb{R}^L$ *be defined by* $\mathbf{b}^* = \mathbf{b}^{*,(1)} + \mathbf{b}^{*,(2)}$, *where*

$$b_l{}^{*,(1)} := \frac{q}{L}\sum_{k=1}^{p} \Psi_{kl}\theta_k^{*,(1)} W_{1\pi_l(k)},$$

$$b_l{}^{*,(2)} := \frac{pq}{L}\sum_{k=1}^{p}\Psi_{kl}\sum_{k_1=1}^{p}\Theta_{kk_1}^{*,(2)}\mathbb{1}_{\{\pi_l(k_1)<\pi_l(k)\}}W_{2\pi_l(k)},$$

and $\mathbf{W}$ *is a* $2\times p$ *matrix with each row a vector of weights. Then there exists a choice of weight matrix such that* $\mathbf{b}^*$ *has the following properties:*

(i) $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\mathbf{S}\mathbf{b}^*) = \mathbf{f}^*$;

(ii) $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2) \leq (2-q/p)q\ell^2(\boldsymbol{\Theta}^*)/L$;

(iii) $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{f}^*\|_2^2)/n \leq \mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2)$.

The bound on the approximation error in (iii) is most suited to situations where there are a fixed number of interaction terms, so

$$\sum_{k,k_1,k_2}\left|\Theta_{kk_1}^{*,(2)}\Theta_{kk_2}^{*,(2)}\right| = O(1). \tag{4.15}$$

Then we see that the contribution of the interaction terms to the bound on the approximation error is of order q^2. On the other hand, if we are considering a growing number of many small interaction terms, much tighter bounds than that given by (iii) can be obtained. The results for interaction models corresponding to Theorems 21, 22 and 23 now follow.

4.4.2 Linear regression models

Assume the model (4.4) and define the MSPE by (4.5) but in both cases with $\mathbf{X}\boldsymbol{\beta}^*$ now replaced by $\mathbf{f}^*$ defined in (4.13). Note that where before, the structural error term $\boldsymbol{\gamma}$ necessarily included two-way interaction terms if they were present, here the deterministic error need only include three-way and higher order interactions.

4.4.2.1 Ordinary least squares

Theorem 25. *Let* $(\hat{\alpha}, \hat{\mathbf{b}})$) *be the least squares estimator* (4.6) *and let* $L^* := \sqrt{(2-q/p)qn}\,\ell(\boldsymbol{\Theta}^*)/\sigma$. *We have*

$$\text{MSPE}((\hat{\alpha},\hat{\mathbf{b}})) \;\leq\; 2\sqrt{2-\frac{q}{p}}\max\left\{\frac{L}{L^*},\frac{L^*}{L}\right\}\sigma\sqrt{\frac{q}{n}}\ell(\boldsymbol{\Theta}^*) \;+\; \frac{\|\boldsymbol{\gamma}\|_2^2}{n} + \frac{\sigma^2}{n}.$$

To interpret the result, consider a situation where there are a fixed number of interaction and main effects of fixed size, so in particular (4.15) holds. If n, q and p increase by collecting new data and adding uninformative variables, then in order for the MSPE to vanish asymptotically, we require $q^2/n \to 0$. Compare this to the corresponding requirement of OLS applied to $\mathbf{X}$, that $p^2/n \to 0$. Particularly in situations of increasing variable sparseness, as discussed in Section 4.3.2.1, this can amount to a large statistical advantage.

The computational gains can be equally great. In the situation considered here, the optimal dimension $L^* = O(q\sqrt{n})$. If, for example, $n \approx q^2$, then $L^* = O(q^2)$. If ridge regression were applied to $\mathbf{X}$ augmented by $O(p^2)$ interaction terms, the number of operations required would be $O(p^2q^4)$; OLS using $\mathbf{S}$ has complexity $O(q^6)$. If instead $n \approx p^2$, then regression with explicitly coded interaction terms would have complexity $O(p^6)$, whilst with the compressed data this would be reduced to $O(p^4q^2)$.

4.4.2.2 Ridge regression

Here we define the ridge regression estimator $(\hat{\alpha}, \hat{\mathbf{b}}^\eta)$ where $\eta > 0$ by

$$\hat{\mathbf{b}}^\eta := \arg\min_{\mathbf{b}} \|\mathbf{Y} - \bar{\mathbf{Y}} - \mathbf{S}\mathbf{b}\|_2^2 \quad \text{such that} \quad \|\mathbf{b}\|_2^2 \leq (1+\eta)\frac{2(2-q/p)q\ell^2(\mathbf{\Theta}^*)}{L}.$$

Note the only difference to (4.7) is that the ℓ_2-norm of $\hat{\mathbf{b}}^\eta$ is allowed to be larger here, for the same value of η.

Theorem 26. *Define*

$$\rho_2 := \exp\left(-\frac{L\eta^2}{18(2-q/p)q\{18(2-q/p)+\eta\}}\right) + \exp\left(-\frac{L\eta^2}{36(2-q/p)^2q^2(18+\eta)}\right), \tag{4.16}$$

and let L^* be as in Theorem 25. Then we have

$$\text{MSPE}((\hat{\alpha}, \hat{\mathbf{b}}^\eta)) \leq \sqrt{(2-q/p)q}\,\ell(\mathbf{\Theta}^*)\left(\frac{2\sigma\sqrt{1+\eta}+\sqrt{2}(L^*/L)}{\sqrt{n}}\right) + \rho_2\frac{\|\mathbf{f}^* - \bar{\mathbf{f}^*}\mathbf{1} + \boldsymbol{\gamma}\|_2^2}{n} + \frac{\|\boldsymbol{\gamma}\|_2^2}{n} + \frac{\sigma^2}{n},$$

where

$$\bar{\mathbf{f}^*} := \frac{1}{n}\sum_{i=1}^n f_i^*.$$

As with Theorem 22 the result here suggests choosing a large L is always better from a statistical point of view. However, for computational reasons, it may not be possible to take L much larger than L^*.

4.4.3 Logistic regression

Here we assume the model (4.9) and define the excess risk by (4.11), but in both cases with $\mathbf{X}\boldsymbol{\beta}^*$ replaced by $\mathbf{f}^*$. Let the estimator $\hat{\mathbf{b}}^\eta$ be given for some $\eta > 0$ by

$$\hat{\mathbf{b}}^\eta = \arg\min_{\mathbf{b}} \frac{1}{n}\sum_{i=1}^n \left[-Y_i\mathbf{s}_i^T\mathbf{b} + \log\{1+\exp(\mathbf{s}_i^T\mathbf{b})\}\right] \quad \text{such that} \quad \|\mathbf{b}\|_2^2 \leq (1+\eta)\frac{2(2-q/p)q\ell^2(\mathbf{\Theta}^*)}{L}.$$

Theorem 27. *Let ρ_2, $\tilde{p}$ and L^* be as in* (4.16), (4.12) *and Theorem 25 respectively. Then we have*

$$\mathbb{E}_{\mathbf{Y},\boldsymbol{\pi},\boldsymbol{\Psi}}\{\mathcal{E}(\hat{\mathbf{b}}^{\eta})\} \leq \sqrt{(2-q/p)q}\,\ell(\boldsymbol{\Theta}^*)\left(\frac{\sqrt{(1+\eta)\tilde{p}}+(\sqrt{2}L^*)/(4L)}{\sqrt{n}}\right)+\log(2)\rho_2.$$

One could continue to look at higher-order interaction models by adding three-way interactions in (4.13) and adapting (4.14) in a suitable way. However, being able to show that two-way interaction models can be fitted with MRS mapping may well be sufficient for most applications.

4.5 Extensions

4.5.1 Map aggregation

Since the compressed design matrix $\mathbf{S}$ is generated in a random fashion, we can repeat the construction $B > 1$ times to obtain B different $\mathbf{S}$ matrices. In the spirit of bagging (Breiman, 1996; Bühlmann and Yu, 2002) we can then aggregate the predictions obtained from the different random mappings by averaging them. In our experience, there has been a marked improvement when using map aggregation, and L could be chosen much lower than for $B = 1$ to achieve the same predictive accuracy. Since computational cost is typically quadratic in L, this can result in large savings. Furthermore, the computations when $B > 1$ lend themselves to a trivial parallel implementation for both the creation of the different $\mathbf{S}$ matrices and at the model fitting stage.

4.5.2 Variable importance

Typically prediction, rather than model selection, is the primary goal in large-scale applications with sparse data, one reason for this being that we cannot expect a very small subset of variables to approximate the signal well when the design matrix is sparse. Nevertheless, it is often illuminating to study the influence of specific variables or look for the variables that have the largest influence on predictions. Indeed, such study is often undertaken following fits using Random Forest (Breiman, 2001), where several variable importance measures allow practitioners to better interpret the fits produced.

We now describe how importance measures can be obtained for MRS mapping. Let $\hat{f} : \mathbb{R}^p \to \mathbb{R}$ be the regression function created following MRS mapping, and let $\hat{f}_i := \hat{f}(\mathbf{x}_i)$. Furthermore, for $1 \leq k \leq p$, let $\hat{f}^{(-k)} := \hat{f}(\mathbf{x}_i^{(-k)})$, where $\mathbf{x}_i^{(-k)}$ is equal to $\mathbf{x}_i$ but with k^{th} component set to zero.

The vector $\hat{\mathbf{f}} - \hat{\mathbf{f}}^{(-k)}$ is the difference in predictions obtained when fitting to $\mathbf{X}$, and those obtained when fitting to $\mathbf{X}$ with the k^{th} column set to zero. When the underlying model in $\mathbf{X}$ contains only main effects (4.4) and no structural error is present, we might expect that

$$\hat{\mathbf{f}} - \hat{\mathbf{f}}^{(-k)} \approx \boldsymbol{\beta}_k^* \mathbf{X}_k.$$

To obtain a measure of variable importance, one could look at the ℓ_2-norm of $\hat{\mathbf{f}} - \hat{\mathbf{f}}^{(-k)}$, for example (Breiman, 2001).

The difference in predictions can be computed relatively easily by storing an $n \times L$ matrix $\tilde{\mathbf{S}}$ with entries given by $\tilde{S}_{il} = \Psi_{\tilde{H}_{il}l} X_{i\tilde{H}_{il}}$, where

$$\tilde{H}_{il} := \underset{k \in \mathbf{z}_i \setminus H_{il}}{\arg\min}\, \pi_l(k).$$

Thus $\tilde{H}_{il}$ is the variable index in $\mathbf{z}_i$ whose value under permutation π_l is second smallest among $\{\pi_l(k) : k \in \mathbf{z}_i\}$. If $\mathbf{z}_i \setminus H_{il} = \emptyset$, we simply set $\tilde{S}_{il} = 0$. Then

$$\hat{f}_i - \hat{f}_i^{(-k)} = \sum_{l=1}^{L} (S_{il} - \tilde{S}_{il}) \mathbb{1}_{\{H_{il}=k\}} \hat{b}_l. \tag{4.17}$$

Note that we only need to store the three $n \times L$ matrices $\mathbf{S}, \tilde{\mathbf{S}}$ and $\mathbf{H}$ to compute the variable importance for all variables.

Interaction effects are not directly visible, but do manifest themselves in the form of a higher variability among $\{\hat{f}_i - \hat{f}_i^{(-k)} : \mathbf{x}_i \approx \mathbf{x}\}$, for any given value of $\mathbf{x}$, if variable k is involved in an interaction term. In principle, one could attempt to detect this increased variability, but further investigation of this is beyond the scope of the current work.

4.5.3 Other fitting procedures

Here we have only considered OLS, ridge regression and ℓ_2-penalised logistic regression as prediction methods after reducing the design matrix. However, it is also conceivable that other fitting procedures could be suitable. In particular, it would be interesting to look at matching pursuit, boosting and the Lasso, for which results in (Bühlmann, 2006; Tropp, 2004; van de Geer, 2008) could be leveraged. Matching pursuit would have the computational advantage that the entire $\mathbf{S}$ matrix would not need to be held in memory. Instead, one could create the columns during the fitting process. Such an approach may be useful for problems where the dimension of the mapping, L, needs to be very large to achieve a desired predictive accuracy.

4.6 Numerical examples

4.6.1 Regression: simulation

Here we compare the predictive performance of MRS mapping followed by OLS to ridge regression, the Lasso and Random Forest (Breiman, 2001). We apply the procedures to datasets of moderate size generated under various simulation settings. We generate data points $i = 1, \ldots, n$ from the model

$$Y_i = \sum_{k=1}^{p} X_{ik}\beta_k^* + \kappa \sum_{(k_1,k_2)\in I} X_{ik_1}X_{ik_2} + \varepsilon_i := f_i^* + \varepsilon_i, \tag{4.18}$$

where $\mathbf{X} \in \{0,1\}^{n\times p}$ is a binary design matrix, I is a collection of indices for interaction terms, and the ε_i are independent $N(0,\sigma^2)$-distributed. The interaction strength κ controls the amount the interaction terms contribute to the signal. The design matrix is generated in the following way. The components of the first predictor variable are independent Bernoulli(ς) for a sparsity parameter $\varsigma \in (0,1)$. For variables, $k = 2, \ldots, p$, we set the entry $X_{i,k}$ equal to $X_{i,k-1}$ with probability $\rho \in [0,1)$ and equal to a Bernoulli(ς) variable otherwise. The main effects β_k^* are identically 0 except for s non-zero coefficients randomly selected from $\{1, \ldots, p\}$ whose values are chosen independently from a $N(0,1)$ distribution and rescaled so that $n^{-1}\|\mathbf{X}\boldsymbol{\beta}^*\|_2^2 = 1$. Each variable index within each pair corresponding to an interaction term is drawn at random from $\{1, \ldots, p\}$. We measure predictive performance by the MSPE (4.5) with $\mathbf{X}\boldsymbol{\beta}^*$ replaced by $\mathbf{f}^*$.

Some representative results for different choices of ς, p, n, s, σ, ρ and κ are shown in Figure 4.1. For MRS mapping, we show the prediction error under progressively larger values of L. A two-fold

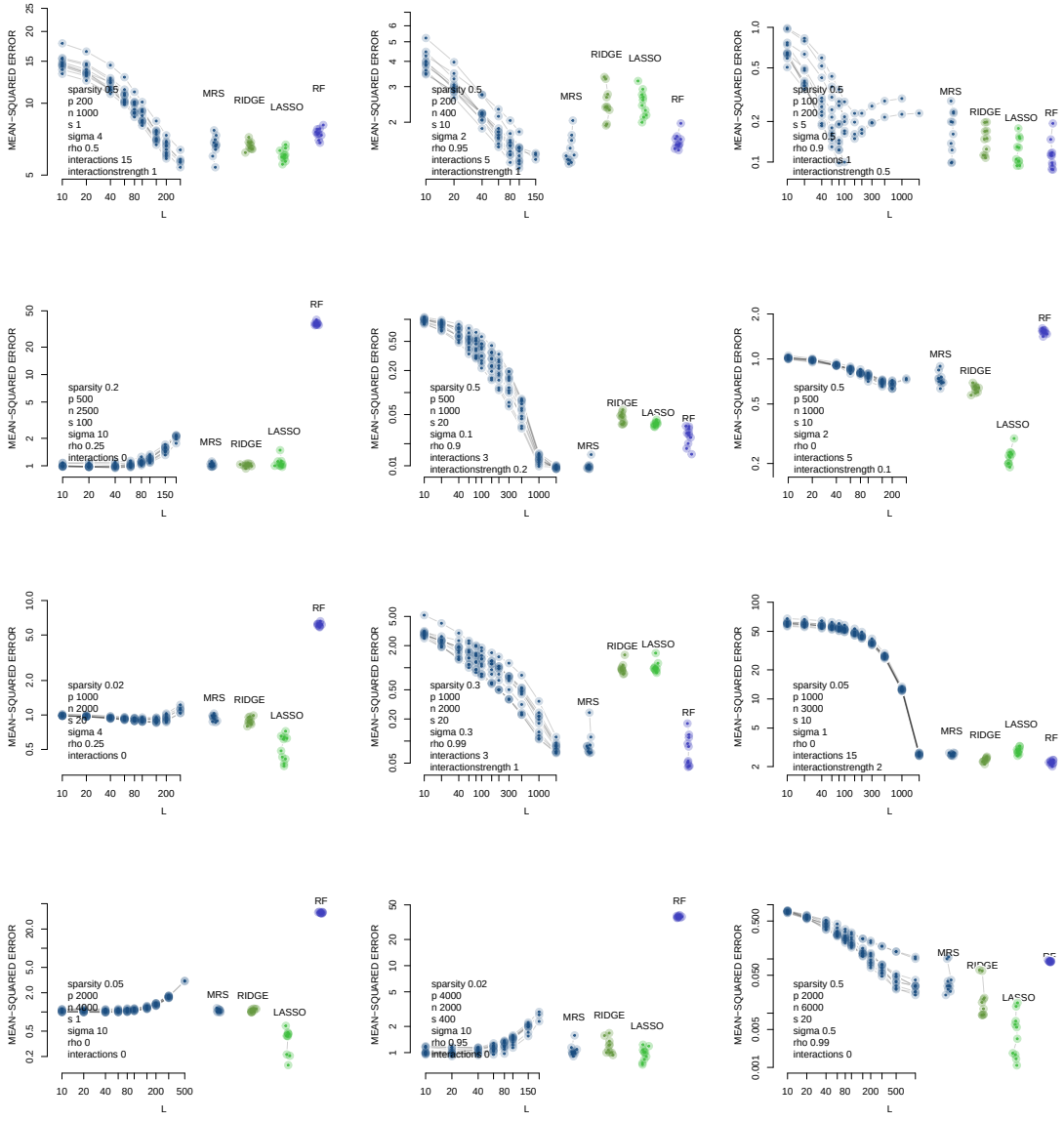


Figure 4.1: The MSPE of OLS after MRS mapping (MRS) with different choices of L, under different parameter settings in the model (4.18) compared to the MSPEs of ridge regression (RIDGE), the Lasso (LASSO) and Random Forests (RF). For each setting, the MRS mapping procedure looks at increasing values of L until the corresponding cross-validated error curve starts to rise; the MSPE for the selected value of L is shown in the first column to the right of the curve. The different curves and points correspond to one of ten different repetitions of each experiment.

cross-validated error is computed for each L and once this reaches above 5% of the minimum value, we stop increasing L further (it also stops once a value of $L = 4000$ is reached). To make OLS numerically stable we apply a small ridge penalty by inflating the diagonal values of the covariance matrix of the compressed matrix $\mathbf{S}$ by 1%. Predictions are averaged over $B = 100$ iterations (see 4.5.1). For comparison, we also present the prediction errors of Random Forest, the Lasso and ridge regression; the tuning parameters for the latter two being chosen by five-fold cross-validation.

We see that

(i) In the absence of interactions, the MSPE is very similar to that of ridge regression;

(ii) When interactions are present, the MRS mapping procedure is able to fit the interaction terms and predictive performance comes close to that of Random Forest whilst the purely linear procedures fare poorly.

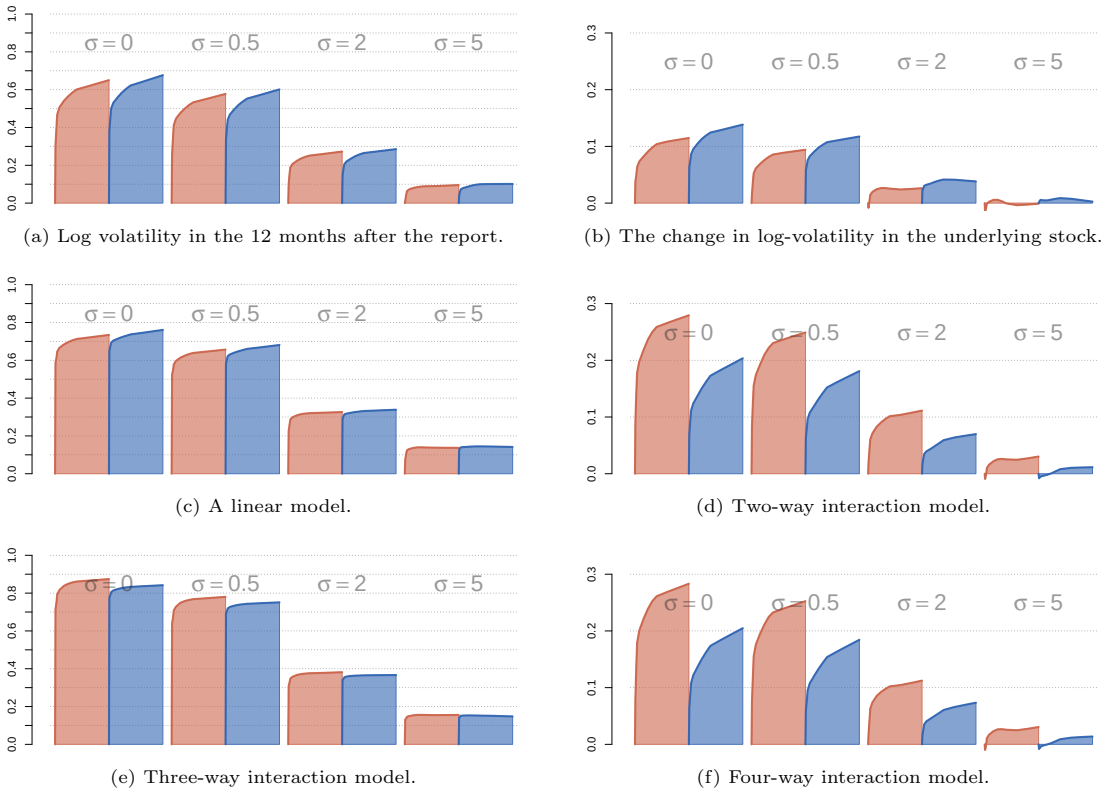
To fit interactions between $p = 1000$ variables in the conventional way we would first have expand the original design matrix to include roughly $5 \cdot 10^5$ interaction terms and then perform a penalised regression in the resulting very high-dimensional model. In contrast, MRS mapping is able to account for a large fraction of the interaction effects when performing regression in a roughly $L = 1000$-dimensional space.

4.6.2 Regression: text analysis

Kogan et al. (2009) analysed a corpus of financial reports of US companies (10-K filings) to study the extent to which a change in the volatility of the underlying stock can be forecast based on the report. One focus of their work was the change in predictive accuracy over time and tying this change to underlying financial reforms. Here, we take a more simplistic view and use the $16,087$ reports provided at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets` to try to forecast the change in log-volatility in stock returns (comparing the 12 months after and before the report) based on the predictor variables $\mathbf{X}_k$, $k = 1, \ldots, p = 4,272,227$ of log-scaled term frequencies of unigrams and bigrams. The number of non-zero predictor variables is between a few thousand up to roughly twenty thousand.

As well as using the original response values, we also generate linear and non-linear responses using the same design matrix. For the linear model, we draw each regression coefficient at random from a standard normal distribution, independently for all variables. For the non-linear experiments, we first divide predictor variables randomly into 10 groups. For each group we form a weighted average of the variables with independent standard normal distributed weights. For each observation, we then subtract the median across all weighted averages from each weighted average. Finally, a sign transformation is applied to each resulting value to form a $16,087$ by 10 transformed design matrix $\mathbf{Z}$ with entries in $\{-1, 0, 1\}$.

In total we consider six different scenarios with the response generated as follows: (a) the log-volatility in the 12 months after the report (not using the pre-report volatility); (b) the change in log-volatility in the underlying stock; (c) the linear model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\varepsilon}$ with standard normal distributed coefficients β_k^*, $k = 1, \ldots, p$; (d) a two-way interaction model based on the transformed data $\mathbf{Z}$, $\mathbf{Y} = \sum_{k=1}^{9} \mathbf{Z}_k \mathbf{Z}_{k+1} + \boldsymbol{\varepsilon}$; (e) a three-way interaction model $\mathbf{Y} = \sum_{k=1}^{8} \mathbf{Z}_k \mathbf{Z}_{k+1} \mathbf{Z}_{k+2} + \boldsymbol{\varepsilon}$; and finally (f) a four-way interaction model $\mathbf{Y} = \sum_{k=1}^{7} \mathbf{Z}_k \mathbf{Z}_{k+1} \mathbf{Z}_{k+2} \mathbf{Z}_{k+3} + \boldsymbol{\varepsilon}$. The noise term $\boldsymbol{\varepsilon}$ has independent normally distributed components with mean zero and variance σ^2 times the



(a) Log volatility in the 12 months after the report.

(b) The change in log-volatility in the underlying stock.

(c) A linear model.

(d) Two-way interaction model.

(e) Three-way interaction model.

(f) Four-way interaction model.

Figure 4.2: Correlation between predicted and actual targets for the text regression example for settings (a)–(f). For each noise level, the correlation is shown as a function of L, which varies between 0 and 250. The red curve corresponds to MRS mapping; the blue, random projections. For linear models and with the original response, the performance of MRS mapping and random projections are similar here. MRS mapping has an advantage when the signal contains stronger non-linearities.

empirical variance of the signal. We measure predictive accuracy with 5-fold cross-validation and show the resulting correlation between predicted and actual target values in Figure 4.2.

In this example, the dimensionality of the data prohibits computation of penalised regression models using the original design matrix. Thus, for comparison, we use random projections with i.i.d. standard normal entries in the projection matrix (see Section 4.2.4). Figure 4.2 shows the results. We see that random projections and MRS mapping perform similarly for the linear model in scenario (c), with the former doing slightly better. The original data (a) and (b) show a very similar pattern across the various noise levels. For the non-linear scenarios (d)–(f), however, MRS mapping outperforms random projections. While latter can only attempt to fit the best linear approximation in these examples, there is more scope to fit interactions with the MRS mapping-based regression.

4.6.3 Classification: URL identification

Ma et al. (2009) reported on a large-scale classification task of identifying malicious URLs in (near) real time. Each URL is associated with both lexical (derived for example from host name and path tokens) and host-based binary predictor variables (derived for example from WHOIS info and the IP prefix). Data was collected over the course of 4 months. On each day, 20,000 URLs were

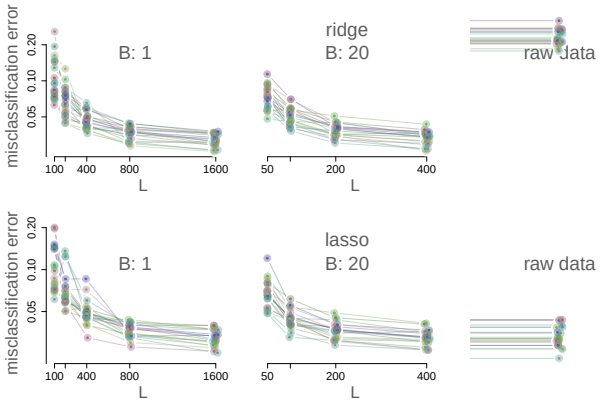


Figure 4.3: Prediction accuracy on the test data for the first 30 days of the URL identification dataset. In each plot, the misclassification error is shown as a function of L for $B = 1$ (first panel) and for $B = 20$ (second panel). The results for regression on the original data are shown in the rightmost panel. In the top row, a ridge penalty is used in the logistic model for both the transformed data and the original data, whereas a Lasso cross-validated penalty is used in the bottom row.

collected, of which roughly 1/3 were malicious, and the remaining, benign.

For a given URL, a few hundred features will be active, that is, with a nonzero entry. For each day, the total number of active features across all URLs for that day was at least $50,000$. Over the course of all days, there are more than $3 \cdot 10^6$ active features.

An important issue is that the distribution of the data is changing over time. Ma et al. (2009) propose a stochastic gradient approach that can learn in real time. Here we do not want to go into all the details of the distributional change but simply compare MRS mapping-based logistic regression with Lasso- and ridge-penalised logistic regression using the original data. Due to the size of the dataset, the latter two approaches can only be performed in acceptable computational time in a batch approach. Hence we treat the data from different days as different datasets and for each day, we train on the first $10,000$ URLs, and test on the remaining $10,000$. We use five-fold cross-validation with logistic loss to select the tuning parameters for penalised regressions, with the fits computed using the `glmnet` package Friedman et al. (2010). We then apply these same estimation procedures with the same tuning parameters to the MRS mapped data. The dimension L of the mapping was varied between 100 and 2000. As well as standard MRS mapping-based regression, we also looked at averaging over the predictions given by $B \in \{1, 20\}$ different $\mathbf{S}$ matrices, as described in Section 4.5.1. To measure performance, we record the misclassification error when the classification threshold is chosen to produce the same error rate in both classes.

Some results are shown in Figure 4.3. The misclassification error drops as L increases. In all four cases, for L in the thousands, the error approaches that when running Lasso on the original, much higher-dimensional, data. This occurs for lower L when averaging over $B = 20$ predictions, than when just using a single $\mathbf{S}$ matrix. Ridge regression on the original data performs much worse. Note that we can easily extend MRS mapping to use all days as training input, since the dimension L is low enough for subsequent regression to be computationally feasible.

4.7 Discussion

The large-scale sparse data setting presents many new challenges to statisticians that require novel approaches to overcome them. One could summarise the conventional process by which statistical methodology is developed in two stages: first a procedure that is statistically optimal for the data-generating process of interest is sought out; next, one may attempt to produce fast algorithms for the procedure. In our 'large p, large n' context, it often makes more sense to consider computational issues alongside, or even before, statistical ones.

In this work, we have taken b-bit min-wise hashing (Li and König, 2011) as our starting point. From a computational point of view, it is clear that this is very well-suited to large-scale sparse data, and can retain computational feasibility where other dimension reduction techniques, such as those based on PCA, may fail. Its statistical properties, however, are harder to discern immediately.

Rather than studying b-bit min-wise hashing directly, here we considered a variant, MRS mapping. For this latter procedure, we were able to show that not only does it take advantage of sparsity in the design matrix computationally, it also exploits this for improved statistical performance. In particular, the MSPE of regression following dimension reduction by MRS mapping is of the form $\sqrt{q/n}\|\boldsymbol{\beta}^*\|_2$ if the data follow a linear model with coefficient vector $\boldsymbol{\beta}^*$ and q is the maximal number of non-zero variables for an observation. The linear model can then be well-approximated by the low-dimensional MRS mapped data if the norm of $\|\boldsymbol{\beta}^*\|_2$ is low, as occurs, for example if the signal is approximately replicated in distinct blocks of variables.

In addition, we have shown that interaction models can be fit by a regression on the MRS mapped data that contains only main effects. Though a larger dimension of the mapped data L may be required than when approximating a main effects model, no further changes are needed to the procedure.

In summary, regression on MRS mapped data with only main effects can be a very powerful prediction engine in settings with millions of predictors and observations. Moreover, the memory footprint and computational cost of the procedure is such that this can be performed with ease on standard computing equipment. We expect to see more extensions and applications of MRS mapping and other methods based on b-bit min-wise hashing in the future.

4.8 Appendix

In the proofs which follow, we will let $\pi := \pi_1$, $\boldsymbol{\psi} := \boldsymbol{\Psi}_1$. Note that then π and $\boldsymbol{\psi}$ have the same distribution as π_l and $\boldsymbol{\Psi}_l$ respectively, for any l. Similarly, we will write M_i and H_i for M_{i1} and H_{i1} respectively, where $\mathbf{M}$ is defined as in (4.3) and $\mathbf{H}$ as in (4.1). Furthermore, we will let $\delta := q/p$.

Proof of Theorem 20. There are three steps to the proof: first we determine conditions on $\mathbf{w}$ that are necessary and sufficient for the unbiasedness property (i) to hold; in the second step, we pick $\mathbf{w}$ to minimise $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2)$; finally we compute the variance $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{X}\boldsymbol{\beta}^*\|_2^2)/n$.

Step 1: We begin by computing

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^*)_i) &= \frac{q}{L}\sum_{l=1}^{L}\mathbb{E}_{\pi_l,\boldsymbol{\Psi}_l}\left(S_{il}\sum_{k=1}^{p}\Psi_{kl}\beta_k^* w_{\pi_l(k)}\right)\\ &= q\mathbb{E}_{\pi,\boldsymbol{\psi}}\left(\sum_{j=1}^{p}X_{ij}\mathbb{1}_{\{H_i=j\}}\psi_j\sum_{k=1}^{p}\beta_k^*\psi_k w_{\pi(k)}\right).\end{aligned}$$

Using the independence of Ψ and π, and the fact that $\mathbb{E}_{\boldsymbol{\psi}}(\psi_j\psi_k) = \mathbb{1}_{\{k=j\}}$, we have that the above display equals

$$q\mathbb{E}_{\pi}\left(\sum_{k=1}^{p}X_{ik}\beta_k^*\mathbb{1}_{\{H_i=k\}}\sum_{\ell=1}^{p}w_\ell\mathbb{1}_{\{\pi(k)=\ell\}}\right).$$

Now observe that

$$\mathbb{1}_{\{H_i=k\}}\mathbb{1}_{\{\pi(k)=\ell\}} = \mathbb{1}_{\{H_i=k\}}\mathbb{1}_{\{M_i=\ell\}},$$

and $\mathbb{1}_{\{H_i=k\}}$ and $\mathbb{1}_{\{M_i=\ell\}}$ are independent. Thus we have

$$\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^*)_i) = \sum_{k=1}^{p}X_{ik}\beta_k^*\sum_{\ell=1}^{p}\mathbb{P}_\pi(M_i=\ell)w_\ell. \tag{4.19}$$

Note that

$$\mathbb{P}_\pi(M_i=\ell) = \binom{p-\ell}{q-1}\Big/\binom{p}{q},$$

so in order for unbiasedness to hold, the inner product in (4.19) must satisfy

$$\sum_{\ell=1}^{p}w_\ell\binom{p-\ell}{q-1}\Big/\binom{p}{q} = 1. \tag{4.20}$$

Step 2: To compute $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2)$, we first observe that the components of $\mathbf{b}^*$ are independent and each has expectation 0 as $\mathbb{E}_{\boldsymbol{\Psi}_l}(b_l^*|\pi_l) = 0$. Therefore

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2) &= \frac{q^2}{L}\mathbb{E}_{\pi,\boldsymbol{\psi}}\left\{\left(\sum_{k=1}^{p}\beta_k^*\psi_k w_{\pi(k)}\right)^2\right\}\\ &= \frac{q^2}{L}\mathbb{E}_{\pi}\left[\mathbb{E}_{\boldsymbol{\psi}}\left\{\left(\sum_{k=1}^{p}\beta_k^*\psi_k\sum_{\ell=1}^{p}w_\ell\mathbb{1}_{\{\pi(k)=\ell\}}\right)^2\Bigg|\pi\right\}\right]\\ &= \frac{q^2}{L}\sum_{k=1}^{p}\beta_k^{*2}\sum_{\ell=1}^{p}w_\ell^2\mathbb{P}_\pi(\pi(k)=\ell)\\ &= \frac{q^2}{pL}\|\boldsymbol{\beta}^*\|_2^2\|\mathbf{w}\|_2^2.\end{aligned} \tag{4.21}$$

Thus to minimise the $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2)$, we must minimise $\|\mathbf{w}\|_2^2$ subject to the constraint (4.20).

Now by the the Cauchy–Schwarz inequality,

$$\|\mathbf{w}\|_2^2 \geq \frac{1}{\sum_{\ell=1}^{p}\binom{p-\ell}{q-1}^2/\binom{p}{q}^2},$$

with equality if and only if

$$w_\ell = \frac{\binom{p-\ell}{q-1}^2 / \binom{p}{q}^2}{\sum_{\ell'=1}^{p} \binom{p-\ell'}{q-1}^2 / \binom{p}{q}^2}. \tag{4.22}$$

Applying part (i) of Lemma 30 with the sequences

$$b_\ell = \binom{p-\ell}{q-1} \Big/ \binom{p}{q}, \qquad a_\ell = \frac{q}{p}\left(1-\frac{q}{p}\right)^{\ell-1},$$

we conclude that

$$\begin{aligned}\sum_{\ell=1}^{p-q+1} \binom{p-\ell}{q-1}^2 \Big/ \binom{p}{q}^2 &\geq \frac{q^2}{p^2} \sum_{\ell=1}^{\infty} \left(1-\frac{q}{p}\right)^{2\ell-2} \\ &= \frac{q^2}{p^2} \frac{1}{1-(1-q/p)^2} \\ &= \frac{q}{2p-q}. \end{aligned} \tag{4.23}$$

This yields

$$\|\mathbf{w}\|_2^2 \leq \frac{2p-q}{q}. \tag{4.24}$$

Substituting into (4.21) then gives part (ii) of the result.

Step 3: Turning to the variance,

$$\begin{aligned}\frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{X}\boldsymbol{\beta}^*\|_2^2) &= \frac{1}{n}\sum_{i=1}^{n} \mathrm{Var}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^*)_i) \\ &= \frac{1}{n}\sum_{i=1}^{n}\sum_{l=1}^{L} \mathrm{Var}_{\pi_l,\boldsymbol{\Psi}_l}(S_{il}b_l^*) \\ &= \frac{1}{n}\sum_{i=1}^{n}\sum_{l=1}^{L} \mathbb{E}_{\pi_l,\boldsymbol{\Psi}_l}((S_{il}b_l^*)^2) - (\mathbf{x}_i^T\boldsymbol{\beta}^*)^2/L^2. \end{aligned}$$

Now if $\|\mathbf{X}\|_\infty \leq 1$ and hence $\|\mathbf{S}\|_\infty \leq 1$, we see that

$$\frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{X}\boldsymbol{\beta}^*\|_2^2) \leq \mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2),$$

which is property (iii) after using (ii). For the case where this does not hold, we argue as follows.

$$\begin{aligned}\frac{L^2}{q^2}\mathbb{E}_{\boldsymbol{\Psi}_l,\pi_l}((S_{il}b_l^*)^2) &= \mathbb{E}_{\pi,\boldsymbol{\psi}}\left\{\sum_{j=1}^{p} X_{ij}^2 \mathbb{1}_{\{H_i=j\}}\left(\sum_{k=1}^{p}\psi_k\beta_k^* w_{\pi(k)}\right)^2\right\} \\ &= \mathbb{E}_{\pi}\left\{\sum_{j\in\mathbf{z}_i} X_{ij}^2 \mathbb{1}_{\{H_i=j\}}\left(\sum_{k\in\mathbf{z}_i}\beta_k^{*2} w_{\pi(k)}^2 + \sum_{k\notin\mathbf{z}_i}\beta_k^{*2} w_{\pi(k)}^2\right)\right\}. \end{aligned}$$

We calculate the expectation of the terms involving $k \in \mathbf{z}_i$ and $k \notin \mathbf{z}_i$ separately. For the second

set of terms we have

$$\mathbb{E}_\pi\left(\sum_{j\in\mathbf{z}_i} X_{ij}^2 \mathbb{1}_{\{H_i=j\}} \sum_{k\notin\mathbf{z}_i} \beta_k^{*2} \sum_{\ell=1}^{p} w_\ell^2 \mathbb{1}_{\{\pi(k)=\ell\}}\right) = \frac{1}{pq}\|\mathbf{w}\|_2^2 \sum_{j\in\mathbf{z}_i} X_{ij}^2 \sum_{k\notin\mathbf{z}_i} \beta_k^{*2}$$
$$\leq \frac{2-\delta}{q^2} \sum_{j\in\mathbf{z}_i} X_{ij}^2 \sum_{k\notin\mathbf{z}_i} \beta_k^{*2}. \tag{4.25}$$

For the first, note that when $j \in \mathbf{z}_i$,

$$\mathbb{1}_{\{H_i=j\}}\mathbb{1}_{\{\pi(k)=\ell\}} = \mathbb{1}_{\{j=k\}}\mathbb{1}_{\{H_i=k\}}\mathbb{1}_{\{M_i=\ell\}} + \mathbb{1}_{\{j\neq k\}} \sum_{\ell'<\ell} \mathbb{1}_{\{M_i=\ell'\}}\mathbb{1}_{\{\pi(j)=\ell'\}}\mathbb{1}_{\{\pi(k)=\ell\}}.$$

Taking expectations we get

$$\mathbb{P}_\pi(\{H_i=j\}\cap\{\pi(k)=\ell\}) = \frac{1}{q}\left(\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}}\mathbb{1}_{\{j=k\}} + \sum_{\ell'=1}^{\ell-1}\frac{1}{p-\ell'}\frac{\binom{p-\ell'}{q-1}}{\binom{p}{q}}\mathbb{1}_{\{j\neq k\}}\right).$$

But

$$\sum_{\ell'=1}^{\ell-1}\frac{1}{p-\ell'}\frac{\binom{p-\ell'}{q-1}}{\binom{p}{q}} = \frac{q}{p(q-1)}\sum_{\ell'=1}^{\ell-1}\frac{\binom{p-1-\ell'}{q-2}}{\binom{p-1}{q-1}}$$
$$= \frac{q}{p(q-1)}\left(1-\sum_{\ell'=\ell}^{p-1}\frac{\binom{p-1-\ell'}{q-2}}{\binom{p-1}{q-1}}\right)$$
$$= \frac{q}{p(q-1)} - \frac{1}{q-1}\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}}.$$

Thus

$$\mathbb{E}_\pi\left(\sum_{j\in\mathbf{z}_i} X_{ij}^2 \mathbb{1}_{\{H_i=j\}} \sum_{k\in\mathbf{z}_i} \beta_k^{*2} \sum_{\ell=1}^{p} w_\ell^2 \mathbb{1}_{\{\pi(k)=\ell\}}\right)$$
$$= \frac{1}{q}\sum_{j\in\mathbf{z}_i} X_{ij}^2 \sum_{k\in\mathbf{z}_i} \beta_k^{*2} \sum_{\ell=1}^{p} w_\ell^2 \left\{\mathbb{1}_{\{j=k\}}\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}} + (1-\mathbb{1}_{\{j=k\}})\frac{1}{q-1}\left(\frac{q}{p}-\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}}\right)\right\}$$
$$= \frac{1}{q-1}\sum_{k\in\mathbf{z}_i} X_{ik}^2\beta_k^{*2}\sum_{\ell=1}^{p} w_\ell^2\left(\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}} - \frac{1}{p}\right) + \frac{1}{q-1}\sum_{j\in\mathbf{z}_i} X_{ij}^2 \sum_{k\in\mathbf{z}_i}\beta_k^{*2}\sum_{\ell=1}^{p} w_\ell^2\left(\frac{1}{p}-\frac{1}{q}\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}}\right)$$
$$= \text{(I)} + \text{(II)}.$$

As

$$\frac{\binom{p-\ell}{q-1}}{\binom{p}{q}} - \frac{1}{p} \leq \frac{q}{p} - \frac{1}{p} = \frac{q-1}{p},$$

we have that

$$\text{(I)} \leq \frac{2-\delta}{q}\sum_{k=1}^{p} X_{ik}^2\beta_k^{*2}. \tag{4.26}$$

Turning to (II), note that by Chebyshev's order inequality (see page 76 of Steele (2004)),

$$\sum_{\ell=1}^{p} w_\ell^2 \frac{\binom{p-\ell}{q-1}}{\binom{p}{q}} \geq \frac{1}{p}\|\mathbf{w}\|_2^2,$$

whence

$$\text{(II)} \leq \frac{2-\delta}{q^2} \sum_{j \in \mathbf{z}_i} X_{ij}^2 \sum_{k \in \mathbf{z}_i} \beta_k^{*2}. \tag{4.27}$$

Collecting together equations (4.25), (4.26) and (4.27), we get

$$\frac{L^2}{q^2}\mathbb{E}_{\boldsymbol{\Psi}_l,\pi_l}((S_{il}b_l^*)^2) \leq \frac{2-\delta}{q}\left(\frac{1}{q}\|\mathbf{x}_i\|_2^2\|\boldsymbol{\beta}^*\|_2^2 + \sum_{k=1}^{p} X_{ik}^2\beta_k^{*2}\right),$$

and so

$$\frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}\mathbf{b}^* - \mathbf{X}\boldsymbol{\beta}^*\|_2^2) \leq \frac{2-\delta}{Ln}\left(\|\mathbf{X}\|_F^2\|\boldsymbol{\beta}^*\|_2^2 + q\sum_{k=1}^{p}\|\mathbf{X}_k\|_2^2\beta_k^{*2}\right).$$

□

Proof of Theorem 21. Let $\mathbf{b}^* \in \mathbb{R}^L$ be as in Theorem 20. Let us write

$$\mathbf{Y} = \alpha^*\mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\gamma} + \boldsymbol{\varepsilon} = \alpha^*\mathbf{1} + \mathbf{S}\mathbf{b}^* + \boldsymbol{\gamma} + \boldsymbol{\Delta} + \boldsymbol{\varepsilon},$$

so $\boldsymbol{\Delta}$ is the approximation error of $\mathbf{S}\mathbf{b}^*$. Then we have

$$\text{MSPE}((\hat{\alpha}, \hat{\mathbf{b}})) = \frac{1}{n}\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\alpha^*\mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\gamma} - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}}\|_2^2).$$

Now letting $\check{\mathbf{S}} = (\mathbf{1}\ \mathbf{S})$, and $\mathbf{P}_{\check{\mathbf{S}}}$ be the projection on to the column space of $\check{\mathbf{S}}$ (so $\mathbf{P}_{\check{\mathbf{S}}} = \check{\mathbf{S}}\check{\mathbf{S}}^+$, where $\check{\mathbf{S}}^+$ denotes the Moore–Penrose pseudoinverse of $\check{\mathbf{S}}$), we have the following decomposition.

$$\begin{aligned}
\alpha^*\mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\gamma} - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}} &= \alpha^*\mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\gamma} - \mathbf{P}_{\check{\mathbf{S}}}\mathbf{Y} \\
&= \alpha^*\mathbf{1} + \mathbf{S}\mathbf{b}^* + \boldsymbol{\Delta} + \boldsymbol{\gamma} - \mathbf{P}_{\check{\mathbf{S}}}(\alpha^*\mathbf{1} + \mathbf{S}\mathbf{b}^* + \boldsymbol{\Delta} + \boldsymbol{\gamma} + \boldsymbol{\varepsilon}) \\
&= (\mathbf{I} - \mathbf{P}_{\check{\mathbf{S}}})(\boldsymbol{\Delta} + \boldsymbol{\gamma}) - \mathbf{P}_{\check{\mathbf{S}}}\boldsymbol{\varepsilon}.
\end{aligned}$$

Hence

$$\begin{aligned}
\text{MSPE}((\hat{\alpha}, \hat{\mathbf{b}})) &= \frac{1}{n}\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}(\|(\mathbf{I} - \mathbf{P}_{\check{\mathbf{S}}})(\boldsymbol{\Delta} + \boldsymbol{\gamma}) - \mathbf{P}_{\check{\mathbf{S}}}\boldsymbol{\varepsilon}\|_2^2) \\
&= \frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|(\mathbf{I} - \mathbf{P}_{\check{\mathbf{S}}})(\boldsymbol{\Delta} + \boldsymbol{\gamma})\|_2^2) + \frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}\{\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{P}_{\check{\mathbf{S}}}\boldsymbol{\varepsilon}\|_2^2 \,|\, \boldsymbol{\pi}, \boldsymbol{\Psi})\} \\
&\leq \frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\boldsymbol{\Delta}\|_2^2) + \frac{\sigma^2(L+1)}{n} + \frac{1}{n}\|\boldsymbol{\gamma}\|_2^2 \qquad (4.28) \\
&\leq \frac{(2-\delta)q\|\boldsymbol{\beta}^*\|_2^2}{L} + \frac{\sigma^2(L+1)}{n} + \frac{1}{n}\|\boldsymbol{\gamma}\|_2^2,
\end{aligned}$$

where in (4.28) we used the fact that $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\boldsymbol{\Delta}) = \mathbf{0}$, and the final line follows from property (iii) in Theorem 20 assuming bounded predictor variables with $\|\mathbf{X}\|_\infty \leq 1$. For general predictor variables, we just use property (iv) instead of (iii) in Theorem 20. □

Proof of Theorem 22. Let $\mathbf{b}^* \in \mathbb{R}^L$ be as in Theorem 20. Let the event Λ be defined by

$$\Lambda = \left\{ \|\mathbf{b}^*\|_2^2 < (1+\eta)\frac{(2-\delta)q\|\boldsymbol{\beta}^*\|_2^2}{L} \right\}. \tag{4.29}$$

We will drop the superscript η of $\hat{\mathbf{b}}^\eta$ in the following. Further, let a bar over any vector $\mathbf{v}$ denote the average of the components of $\mathbf{v}$, so $\bar{\mathbf{v}} = \sum_j v_j$. Note that $\hat{\alpha} = \overline{\mathbf{Y} - \mathbf{S}\hat{\mathbf{b}}}$, and define $\hat{a}^* = \overline{\mathbf{Y} - \mathbf{S}\mathbf{b}^*}$ and $\mathbf{f}^* = \alpha^*\mathbf{1} + \mathbf{X}\boldsymbol{\beta}^* + \boldsymbol{\gamma}$. From the definition of $\hat{\mathbf{b}}$, we have the following two inequalities:

$$\begin{aligned}\|\mathbf{Y} - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}}\|_2^2 \mathbb{1}_\Lambda &\le \|\mathbf{Y} - \hat{a}^*\mathbf{1} - \mathbf{S}\mathbf{b}^*\|_2^2 \mathbb{1}_\Lambda \\ \|\mathbf{Y} - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}}\|_2^2 \mathbb{1}_{\Lambda^c} &\le \|\mathbf{Y} - \bar{\mathbf{Y}}\mathbf{1}\|_2^2 \mathbb{1}_{\Lambda^c}.\end{aligned}$$

Noting that for any $\mathbf{v}, \mathbf{u} \in \mathbb{R}^n$, $\mathbf{v}^T(\mathbf{u} - \bar{\mathbf{u}}\mathbf{1}) = (\mathbf{v} - \bar{\mathbf{v}}\mathbf{1})^T\mathbf{u}$, rearranging the inequalities above we get

$$\|\mathbf{f}^* - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}}\|_2^2 \mathbb{1}_\Lambda \le -2(\boldsymbol{\varepsilon} - \bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}(\hat{\mathbf{b}} - \mathbf{b}^*)\mathbb{1}_\Lambda + \|\mathbf{f}^* - \hat{a}^*\mathbf{1} - \mathbf{S}\mathbf{b}^*\|_2^2 \mathbb{1}_\Lambda, \tag{4.30}$$

$$\|\mathbf{f}^* - \hat{\alpha}\mathbf{1} - \mathbf{S}\hat{\mathbf{b}}\|_2^2 \mathbb{1}_{\Lambda^c} \le -2(\boldsymbol{\varepsilon} - \bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}\hat{\mathbf{b}}\mathbb{1}_{\Lambda^c} + \|\mathbf{f}^* - \bar{\mathbf{Y}}\|_2^2 \mathbb{1}_{\Lambda^c}. \tag{4.31}$$

Observe that as $\overline{\boldsymbol{\gamma}} = 0$,

$$\begin{aligned}\|\mathbf{f}^* - \hat{a}^*\mathbf{1} - \mathbf{S}\mathbf{b}^*\|_2^2 &= \|\mathbf{X}\boldsymbol{\beta}^* - \overline{\mathbf{X}\boldsymbol{\beta}^*}\mathbf{1} - (\mathbf{S}\mathbf{b}^* - \overline{\mathbf{S}\mathbf{b}^*}\mathbf{1}) + \boldsymbol{\gamma}\|_2^2 + n\bar{\varepsilon}^2 \\ &\le \|\mathbf{X}\boldsymbol{\beta}^* - \mathbf{S}\mathbf{b}^* + \boldsymbol{\gamma}\|_2^2 + n\bar{\varepsilon}^2,\end{aligned} \tag{4.32}$$

and

$$\|\mathbf{f}^* - \bar{\mathbf{Y}}\|_2^2 = \|\mathbf{X}\boldsymbol{\beta} - \overline{\mathbf{X}\boldsymbol{\beta}^*}\mathbf{1} + \boldsymbol{\gamma}\|_2^2 + n\bar{\varepsilon}^2. \tag{4.33}$$

As both $\mathbf{b}^*$ and $\mathbb{1}_\Lambda$ are independent of $\boldsymbol{\varepsilon}$, adding together (4.30) and (4.31), simplifying using (4.32) and (4.33), and then taking expectations yields

$$\begin{aligned}\mathrm{MSPE}(\hat{\mathbf{b}}) = &-\frac{2}{n}\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}\{(\boldsymbol{\varepsilon} - \bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}\hat{\mathbf{b}}\} + \frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{X}\boldsymbol{\beta}^* - \mathbf{S}\mathbf{b}^* + \boldsymbol{\gamma}\|_2^2 \mathbb{1}_\Lambda) + \frac{\sigma^2}{n} \\ &+ \frac{1}{n}\|\mathbf{X}\boldsymbol{\beta}^* - \overline{\mathbf{X}\boldsymbol{\beta}^*}\mathbf{1} + \boldsymbol{\gamma}\|_2^2 \mathbb{P}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\Lambda^c).\end{aligned} \tag{4.34}$$

Now using the fact that

$$\|\hat{\mathbf{b}}\|_2 \le \sqrt{1+\eta}\frac{\sqrt{(2-\delta)q}\|\boldsymbol{\beta}^*\|_2}{\sqrt{L}},$$

applying the Cauchy–Schwarz inequality we have

$$\begin{aligned}-\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}\{(\boldsymbol{\varepsilon} - \bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}\hat{\mathbf{b}}\} &\le \sqrt{\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}\{\|\mathbf{S}^T(\boldsymbol{\varepsilon} - \bar{\boldsymbol{\varepsilon}}\mathbf{1})\|_2^2\}\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\hat{\mathbf{b}}\|_2^2)} \\ &\le \sqrt{\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}\{\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{S}^T(\boldsymbol{\varepsilon} - \bar{\boldsymbol{\varepsilon}}\mathbf{1})\|_2^2 | \boldsymbol{\pi}, \boldsymbol{\Psi})\}}\frac{\sqrt{(1+\eta)(2-\delta)q}\|\boldsymbol{\beta}^*\|_2}{\sqrt{L}}.\end{aligned}$$

But

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\varepsilon}}(\|\mathbf{S}^T(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})\|_2^2|\boldsymbol{\pi},\boldsymbol{\Psi}) &= \mathbb{E}_{\boldsymbol{\varepsilon}}[\mathrm{Tr}\{(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}\mathbf{S}^T(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})\}|\boldsymbol{\pi},\boldsymbol{\Psi}]\\ &= \mathbb{E}_{\boldsymbol{\varepsilon}}\{\mathrm{Tr}\{(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}\mathbf{S}^T\}|\boldsymbol{\pi},\boldsymbol{\Psi}]\\ &= \mathrm{Tr}[\mathbb{E}_{\boldsymbol{\varepsilon}}\{(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\}\mathbf{S}\mathbf{S}^T]\\ &= \sigma^2\|(\mathbf{I}-n^{-1}\mathbf{1}\mathbf{1}^T)\mathbf{S}\|_F^2 \le \sigma^2\|\mathbf{S}\|_F^2 \le \sigma^2 nL,\end{aligned}$$

whence

$$-\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}\{(\boldsymbol{\varepsilon}-\bar{\boldsymbol{\varepsilon}}\mathbf{1})^T\mathbf{S}\hat{\mathbf{b}}\} \le \sigma\sqrt{1+\eta}\sqrt{(2-\delta)qn}\|\boldsymbol{\beta}^*\|_2. \tag{4.35}$$

Meanwhile, by Lemma 28 below, we have that $\mathbb{P}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\Lambda^c) \le \rho$, with ρ defined as in (4.8). By Theorem 20, property (i), we have

$$\frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{X}\boldsymbol{\beta}^*-\mathbf{S}\mathbf{b}^*+\boldsymbol{\gamma}\|_2^2) = \frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{X}\boldsymbol{\beta}^*-\mathbf{S}\mathbf{b}^*\|_2^2)+\frac{1}{n}\|\boldsymbol{\gamma}\|_2^2, \tag{4.36}$$

whilst property (iii) gives an upper bound on the approximation error $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{X}\boldsymbol{\beta}^*-\mathbf{S}\mathbf{b}^*\|_2^2)$. Substituting into (4.34) gives the result. □

Proof of Theorem 23. The proof is very similar to that of Theorem 22. Let the event Λ be defined as in (4.29). By the definition of $\hat{\mathbf{b}}$ (dropping the superscript η), we have

$$\frac{1}{n}\sum_{i=1}^n\left[-Y_i\mathbf{s}_i^T\hat{\mathbf{b}}+\log\{1+\exp(\mathbf{s}_i^T\hat{\mathbf{b}})\}\right]\mathbb{1}_\Lambda \le \frac{1}{n}\sum_{i=1}^n\left[-Y_i\mathbf{s}_i^T\mathbf{b}^*+\log\{1+\exp(\mathbf{s}_i^T\mathbf{b}^*)\}\right]\mathbb{1}_\Lambda.$$

Using this, analogously to (4.30) and (4.31) we get,

$$\begin{aligned}\mathcal{E}(\hat{\mathbf{b}})\mathbb{1}_\Lambda &\le \frac{1}{n}\sum_{i=1}^n(Y_i-p_i)\{\mathbf{S}(\hat{\mathbf{b}}-\mathbf{b}^*)\}_i\mathbb{1}_\Lambda+\mathcal{E}(\mathbf{b}^*)\mathbb{1}_\Lambda,\\ \mathcal{E}(\hat{\mathbf{b}})\mathbb{1}_{\Lambda^c} &\le \frac{1}{n}\sum_{i=1}^n(Y_i-p_i)(\mathbf{S}\hat{\mathbf{b}})_i\mathbb{1}_{\Lambda^c}+\mathcal{E}(\mathbf{0})\mathbb{1}_{\Lambda^c},\end{aligned}$$

where $\mathcal{E}(\mathbf{0}) \le \log(2)$ is the excess risk of the the zero-vector $\mathbf{0}\in\mathbb{R}^L$. Let $\boldsymbol{\varepsilon} := \mathbf{Y}-\mathbf{p}$ be the residual vector. Adding the two equations above and taking expectations yields

$$\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}\{\mathcal{E}(\hat{\mathbf{b}})\} \le \frac{1}{n}\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}(\boldsymbol{\varepsilon}^T\mathbf{S}\hat{\mathbf{b}})+\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}\{\mathcal{E}(\mathbf{b}^*)\mathbb{1}_\Lambda\}+\mathcal{E}(\mathbf{0})\mathbb{P}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\Lambda^c).$$

From Lemma 28, we get $\mathbb{P}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\Lambda^c) \le \rho$. By the mean value theorem, we have

$$\mathcal{E}(\mathbf{b}^*) \le \frac{1}{n}\sup_{a\in\mathbb{R}}\left|\frac{e^a}{1+e^a}\left(1-\frac{e^a}{1+e^a}\right)\right|\|\mathbf{X}\boldsymbol{\beta}^*+\boldsymbol{\gamma}-\mathbf{S}\mathbf{b}^*\|_2^2 = \frac{1}{4n}\|\mathbf{X}\boldsymbol{\beta}^*+\boldsymbol{\gamma}-\mathbf{S}\mathbf{b}^*\|_2^2.$$

By (4.36), we then have

$$\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\mathcal{E}(\mathbf{b}^*)) \le \frac{1}{4n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{X}\boldsymbol{\beta}^*-\mathbf{S}\mathbf{b}^*\|_2^2)+\frac{1}{4n}\|\boldsymbol{\gamma}\|_2^2,$$

where (iii) of Theorem 20 give bounds on the quantity in the right-hand-side. Further, the same argument that leads to (4.35) gives

$$\frac{1}{n}\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}(\boldsymbol{\varepsilon}^T\mathbf{S}\hat{\mathbf{b}}) \leq \frac{1}{n}\sqrt{\mathbb{E}_{\boldsymbol{\varepsilon},\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{S}^T\boldsymbol{\varepsilon}\|_2^2)}\sqrt{1+\eta}\frac{\sqrt{(2-\delta)q}\|\boldsymbol{\beta}^*\|_2}{\sqrt{L}} = \sqrt{\frac{(1+\eta)(2-\delta)\tilde{p}}{n}}\|\boldsymbol{\beta}^*\|_2.$$

Collecting together the various inequalities, we get the required result. □

Proof of Theorem 24. The proof proceeds similarly to that of Theorem 20. As in the latter, we set

$$W_{1\ell} := w_l = \mathbb{P}_\pi(M_i = \ell), \tag{4.37}$$

to ensure $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\mathbf{S}\mathbf{b}^{*,(1)}) = \mathbf{X}\boldsymbol{\theta}^{*,(1)}$. Now we shall determine conditions on $\mathbf{w}_2$, the second row of $\mathbf{W}$, such that

$$\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^{*,(2)})_i) = \sum_{k,k_1} X_{ik}\mathbb{1}_{\{X_{ik_1}=0\}}\Theta^{*,(2)}_{kk_1}. \tag{4.38}$$

To this end, we compute

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^{*,(2)})_i) &= \frac{pq}{L}\sum_{l=1}^{L}\mathbb{E}_{\pi_l,\boldsymbol{\Psi}_l}\left(S_{il}\sum_{k=1}^{p}\Psi_{kl}\sum_{k_1=1}^{p}\Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi_l(k_1)<\pi_l(k)\}}W_{2\pi_l(k)}\right)\\
&= pq\mathbb{E}_{\pi,\boldsymbol{\psi}}\left(\sum_{j=1}^{p}X_{ij}\mathbb{1}_{\{H_i=j\}}\psi_j\sum_{k=1}^{p}\psi_k\sum_{k_1=1}^{p}\Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}W_{2\pi(k)}\right)\\
&= pq\mathbb{E}_{\pi}\left(\sum_{k=1}^{p}X_{ik}\mathbb{1}_{\{H_i=k\}}\sum_{k_1=1}^{p}\Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}\sum_{\ell=2}^{p}W_{2\ell}\mathbb{1}_{\{\pi(k)=\ell\}}\right).
\end{aligned}$$

Now observe that for $k \in \mathbf{z}_i$,

$$\mathbb{1}_{\{H_i=k\}}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}\mathbb{1}_{\{\pi(k)=\ell\}} = \mathbb{1}_{\{X_{ik_1}=0\}}\mathbb{1}_{\{H_i=k\}}\mathbb{1}_{\{M_i=\ell,\,\pi(k_1)<\ell\}},$$

and $\mathbb{1}_{\{H_i=k\}}$ and $\mathbb{1}_{\{M_i=\ell,\,\pi(k_1)<\ell\}}$ are independent. Thus we have

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^{*,(2)})_i) &= \sum_{k,k_1}X_{ik}\mathbb{1}_{\{X_{ik_1}=0\}}\Theta^{*,(2)}_{kk_1}\sum_{\ell=1}^{p}p\mathbb{P}_\pi(M_i=\ell,\pi(k_1)<\ell)W_{2\ell}\\
&= \sum_{k,k_1}X_{ik}\mathbb{1}_{\{X_{ik_1}=0\}}\Theta^{*,(2)}_{kk_1}\sum_{\ell=2}^{p}(\ell-1)\mathbb{P}_\pi(M_i=\ell|\,\pi(k_1)<\ell)W_{2\ell}\\
&= \sum_{k,k_1}X_{ik}\mathbb{1}_{\{X_{ik_1}=0\}}\Theta^{*,(2)}_{kk_1}\sum_{\ell=2}^{p}(\ell-1)\frac{\binom{p-\ell}{q-1}}{\binom{p-1}{q}}W_{2\ell}.
\end{aligned}$$

Thus if we choose $\mathbf{w}_2$ such that

$$\sum_{\ell=2}^{p}(\ell-1)\frac{\binom{p-\ell}{q-1}}{\binom{p-1}{q}}W_{2\ell} = 1, \tag{4.39}$$

property (4.38) will be satisfied.

Turning now to the variance, as $\|\mathbf{S}\|_\infty \leq \|\mathbf{X}\|_\infty \leq 1$, we have

$$\mathrm{Var}_{\boldsymbol{\pi},\boldsymbol{\Psi}}((\mathbf{S}\mathbf{b}^*)_i) \leq \frac{1}{L}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2) - f_i^{*2}, \tag{4.40}$$

and by the Cauchy–Schwarz inequality,

$$\begin{aligned}\frac{1}{L}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^*\|_2^2) = \mathbb{E}_{\pi,\boldsymbol{\psi}}(b_1^{*2}) &\leq \left\{\sqrt{\mathbb{E}_{\pi,\boldsymbol{\psi}}\left(b_1^{*,(1)^2}\right)} + \sqrt{\mathbb{E}_{\pi,\boldsymbol{\psi}}\left(b_1^{*,(2)^2}\right)}\right\}^2 \\ &= \frac{1}{L}\left\{\sqrt{\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^{*,(1)}\|_2^2)} + \sqrt{\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^{*,(2)}\|_2^2)}\right\}^2. \end{aligned} \tag{4.41}$$

We now compute $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^{*,(2)}\|_2^2)$ as follows.

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^{*,(2)}\|_2^2) &= \frac{p^2q^2}{L}\mathbb{E}_{\pi,\boldsymbol{\psi}}\left\{\left(\sum_{k=1}^p \psi_k \sum_{k_1=1}^p \Theta_{kk_1}^{*,(2)} \mathbb{1}_{\{\pi(k_1)<\pi(k)\}} W_{2\pi(k)}\right)^2\right\} \\ &= \frac{p^2q^2}{L}\mathbb{E}_{\pi}\left[\mathbb{E}_{\boldsymbol{\psi}}\left\{\left(\sum_{k=1}^p \psi_k \sum_{k_1=1}^p \Theta_{kk_1}^{*,(2)} \mathbb{1}_{\{\pi(k_1)<\pi(k)\}} \sum_{\ell=2}^p W_{2\ell}\mathbb{1}_{\{\pi(k)=\ell\}}\right)^2 \middle| \pi\right\}\right] \\ &= \frac{p^2q^2}{L}\sum_{k=1}^p\sum_{\ell=2}^p W_{2\ell}^2 \mathbb{E}_\pi\left\{\mathbb{1}_{\{\pi(k)=\ell\}}\left(\sum_{k_1=1}^p \Theta_{kk_1}^{*,(2)}\mathbb{1}_{\{\pi(k_1)<\ell\}}\right)^2\right\} \\ &= \frac{p^2q^2}{L}\sum_{\ell=2}^p W_{2\ell}^2\left(\frac{\ell-1}{p(p-1)}\sum_{k,k_1}(\Theta_{kk_1}^{*,(2)})^2 + \frac{(\ell-1)(\ell-2)}{p(p-1)(p-2)}\sum_{k_1\neq k_2}\sum_k \Theta_{kk_1}^{*,(2)}\Theta_{kk_2}^{*,(2)}\right) \\ &\leq \frac{pq^2}{(p-1)L}\sum_{k,k_1,k_2}\left|\Theta_{kk_1}^{*,(2)}\Theta_{kk_2}^{*,(2)}\right|\sum_{\ell=2}^p(\ell-1)W_{2\ell}^2. \end{aligned} \tag{4.42}$$

By the Cauchy–Schwarz inequality, choosing

$$W_{2\ell} = \frac{\binom{p-\ell}{q-1}\Big/\binom{p-1}{q}}{\sum_{\ell'=2}^p(\ell'-1)\left\{\binom{p-\ell}{q-1}\Big/\binom{p-1}{q}\right\}^2} \tag{4.43}$$

minimises (4.42) subject to (4.39) to give

$$\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^{*,(2)}\|_2^2) \leq \frac{pq^2}{(p-1)L}\sum_{k,k_1,k_2}\left|\Theta_{kk_1}^{*,(2)}\Theta_{kk_2}^{*,(2)}\right|\left\{\sum_{\ell=1}^{p-1}\ell\left(\frac{\binom{p-1-\ell}{q-1}}{\binom{p-1}{q}}\right)^2\right\}^{-1}.$$

Finally, Lemma 31 bounds the right-most term from above to yield

$$\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{b}^{*,(2)}\|_2^2) \leq \frac{2\{(2-\delta)q\}^2}{L}\sum_{k,k_1,k_2}\left|\Theta_{kk_1}^{*,(2)}\Theta_{kk_2}^{*,(2)}\right|. \tag{4.44}$$

Using property (ii) of Theorem 20 and substituting (4.44) into (4.41) gives (ii) of Theorem 24. Substituting this into (4.40) then gives property (iii). □

Proof of Theorem 25. Let $\mathbf{b}^* \in \mathbb{R}^L$ be as in Theorem 24. The proof is almost identical to that of Theorem 21, the only difference being in the approximation error term $\boldsymbol{\Delta}$, here defined by

$$\boldsymbol{\Delta} = \mathbf{f}^* - \mathbf{S}\mathbf{b}^*.$$

Analogously to (4.28), we get

$$\begin{aligned}\mathrm{MSPE}((\hat{\alpha}, \hat{\mathbf{b}})) &\leq \frac{1}{n}\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\boldsymbol{\Delta}\|_2^2) + \frac{\sigma^2(L+1)}{n} + \frac{1}{n}\|\boldsymbol{\gamma}\|_2^2 \\ &\leq \frac{(2-\delta)q\ell^2(\boldsymbol{\Theta}^*)}{L} + \frac{\sigma^2(L+1)}{n} + \frac{1}{n}\|\boldsymbol{\gamma}\|_2^2,\end{aligned}$$

the final line following from property (iii) in Theorem 24. □

Proof of Theorem 26. Let $\mathbf{b}^* \in \mathbb{R}^L$ be as in Theorem 24. The proof is almost identical to that of Theorem 22, the only differences being in the definition of the event Λ and the upper bound on the approximation error term. Here, we take

$$\Lambda = \left\{ \|\mathbf{b}^*\|_2^2 < (1+\eta)\frac{(2-\delta)q\ell^2(\boldsymbol{\Theta}^*)}{L} \right\},$$

and we note that Lemma 29 entails $\mathbb{P}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\Lambda^c) \leq \rho_2$, with ρ_2 defined as in (4.16). The proof then follows through as before, replacing the approximation error term in the main effects setting, $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{X}\boldsymbol{\beta}^* - \mathbf{S}\mathbf{b}^*\|_2^2)$, with $\mathbb{E}_{\boldsymbol{\pi},\boldsymbol{\Psi}}(\|\mathbf{f}^* - \mathbf{S}\mathbf{b}^*\|_2^2)$, and using property (iii) in Theorem 24 to bound this from above. □

Proof of Theorem 27. Modifying the proof of Theorem 23 in the same way as proof of Theorem 22 is modified to prove Theorem 26 gives the result. □

Lemma 28. *Let $\mathbf{b}^* \in \mathbb{R}^L$ be as in Theorem 24 with $\mathbf{w}$ defined as in (4.22). Then for $\eta \geq 0$,*

$$\mathbb{P}\left(\|\mathbf{b}^*\|_2^2 \geq (1+\eta)\frac{(2-\delta)q\|\boldsymbol{\beta}^*\|_2^2}{L} \right) \leq \exp\left(-\frac{L\eta^2}{18(2-\delta)q\{18(2-\delta)+\eta\}} \right).$$

Proof. We first bound the moments of b_l^{*2}, in order to later apply Bernstein's inequality (see Lemma 2.2.11 of van der Vaart and Wellner (1996)).

$$\begin{aligned}\mathbb{E}(b_l^{*2m}) &= \frac{q^{2m}}{L^{2m}}\mathbb{E}_\pi\left[\mathbb{E}_{\boldsymbol{\psi}}\left\{ \left(\sum_{k=1}^p \beta_k^* \psi_k w_{\pi_l(k)} \right)^{2m} \middle| \pi \right\} \right] \\ &\leq \frac{(2m)!}{2^m m!}\frac{q^{2m}}{L^{2m}}\mathbb{E}_\pi\left\{ \left(\sum_{k=1}^p \beta_k^{*2} w_{\pi(k)}^2 \right)^m \right\} \qquad (4.45)\end{aligned}$$

by Khintchine's inequality (see Theorem 12.3.1 of Garling (2007)). Then

$$\begin{aligned}\mathbb{E}(b_l^{*2m}) &\leq \frac{(2m)!}{2^m m!}\frac{q^{2m}}{L^{2m}}\mathbb{E}_\pi\left(\sum_{k=1}^p \beta_k^{*2} w_{\pi(k)}^2 \right) \max_\pi \left(\sum_{k=1}^p \beta_k^{*2} w_{\pi(k)}^2 \right)^{m-1} \\ &\leq \frac{(2m)!}{2^m m!}\frac{q^{2m}}{L^{2m}}\|\boldsymbol{\beta}^*\|_2^{2m}\|\mathbf{w}\|_\infty^{2m-1}\mathbb{E}_\pi(w_{\pi(k)}^2).\end{aligned}$$

By (4.24),

$$\|\mathbf{w}\|_\infty \le \frac{q/p}{q/(2p-q)} = 2-\delta,$$
$$\mathbb{E}_\pi(w^2_{\pi(k)}) \le \frac{2p-q}{qp} = \frac{2-\delta}{q}.$$

Using the inequalities

$$n! \ge \left(\frac{n}{3}\right)^n,$$
$$n! \le \left(\frac{n}{2}\right)^n \quad \text{for } n \ge 6,$$

we get

$$\frac{(2m)!}{2^m(m!)^2} \le \left(\frac{9}{2}\right)^m, \tag{4.46}$$

so

$$\mathbb{E}(b_l^{*2m}) \le m!\frac{1}{q}\left(\frac{9(2-\delta)^2q^2\|\boldsymbol{\beta}^*\|_2^2}{2L^2}\right)^m,$$

whence by Minkowski's inequality,

$$\mathbb{E}[\{b_l^{*2} - \mathbb{E}(b_l^{*2})\}^m] \le m!\frac{1}{q}\left(\frac{9(2-\delta)^2q^2\|\boldsymbol{\beta}^*\|_2^2}{L^2}\right)^m.$$

Plugging this into Bernstein's inequality, we finally arrive at

$$\begin{aligned}\mathbb{P}\left(\|\mathbf{b}^*\|_2^2 \ge (1+\eta)\frac{(2-\delta)q\|\boldsymbol{\beta}^*\|_2^2}{L}\right) &\le \mathbb{P}\left(\|\mathbf{b}^*\|_2^2 - \mathbb{E}(\|\mathbf{b}^*\|_2^2) \ge \eta\frac{(2-\delta)q\|\boldsymbol{\beta}^*\|_2^2}{L}\right)\\ &\le \exp\left(-\frac{1}{2}\frac{L\eta^2}{9(2-\delta)q\{18(2-\delta)+\eta\}}\right).\end{aligned}$$

□

Lemma 29. *Let* $\mathbf{b}^* \in \mathbb{R}^L$ *be as in Theorem 24 with* $\mathbf{W}$ *defined by equations* (4.37) *and* (4.43). *Then*

$$\mathbb{P}\left\{\|\mathbf{b}^*\|_2^2 \ge (1+\eta)\frac{(2-\delta)q}{L}\ell^2(\boldsymbol{\Theta})\right\} \le \exp\left(-\frac{L\eta^2}{18(2-\delta)q\{18(2-\delta)+\eta\}}\right) + \exp\left(-\frac{L\eta^2}{36(2-\delta)^2q^2(18+\eta)}\right).$$

Proof. We proceed as in Lemma 28, which provides a bound on the tail probability of $\|\mathbf{b}^{*,(1)}\|_2^2$. To bound the moments of $(b_l{}^{*,(2)})^2$, we argue as follows.

$$\begin{aligned}\mathbb{E}\{(b_l{}^{*,(2)})^{2m}\} &= \frac{p^{2m}q^{2m}}{L^{2m}}\mathbb{E}_\pi\left[\mathbb{E}_{\boldsymbol{\psi}}\left\{\left(\sum_{k=1}^p \psi_k \sum_{k_1=1}^p \Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}W_{2\pi(k)}\right)^{2m}\Big|\pi\right\}\right]\\ &\le \frac{(2m)!}{2^m m!}\frac{p^{2m}q^{2m}}{L^{2m}}\mathbb{E}_\pi\left[\left\{\sum_{k=1}^p\left(\sum_{k_1=1}^p \Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}\right)^2 W^2_{2\pi(k)}\right\}^m\right]\end{aligned}$$

by Khintchine's inequality. Now

$$\begin{aligned}\mathbb{E}_\pi\left[\left\{\sum_{k=1}^p\left(\sum_{k_1=1}^p \Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}\right)^2 W^2_{2\pi(k)}\right\}^m\right] &\leq \mathbb{E}_\pi\left\{\sum_{k=1}^p\left(\sum_{k_1=1}^p \Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}\right)^2 W^2_{2\pi(k)}\right\} \\ &\quad\times \max_\pi\left\{\sum_{k=1}^p\left(\sum_{k_1=1}^p \Theta^{*,(2)}_{kk_1}\mathbb{1}_{\{\pi(k_1)<\pi(k)\}}\right)^2 W^2_{2\pi(k)}\right\}^{m-1} \\ &\leq \frac{2(2-\delta)^2}{p^2}\left(\sum_{k,k_1,k_2}\left|\Theta^{*,(2)}_{kk_1}\Theta^{*,(2)}_{kk_2}\right|\right)^m \|\mathbf{w}_2\|_\infty^{2(m-1)},\end{aligned}$$

the final line following from equations (4.42) and (4.44).

Next, appealing to Lemma 31, we see that $\|\mathbf{w}_2\|_\infty \leq 2(2-\delta)^2 q/p$. Thus we have

$$\mathbb{E}\{(b_l{}^{*,(2)})^{2m}\} \leq \frac{(2m)!}{2^m m!}\frac{1}{2(2-\delta)^2q^2}\left(\frac{4(2-\delta)^4q^4}{L^2}\sum_{k,k_1,k_2}\left|\Theta^{*,(2)}_{kk_1}\Theta^{*,(2)}_{kk_2}\right|\right)^m,$$

so

$$\mathbb{E}[\{(b_l{}^{*,(2)})^2 - \mathbb{E}((b_l{}^{*,(2)})^2)\}^m] \leq m!\frac{1}{2(2-\delta)^2q^2}\left(\frac{36(2-\delta)^4q^4}{L^2}\sum_{k,k_1,k_2}\left|\Theta^{*,(2)}_{kk_1}\Theta^{*,(2)}_{kk_2}\right|\right)^m,$$

by (4.46) and Minkowski's inequality. By Bernstein's inequality,

$$\mathbb{P}\left(\|\mathbf{b}^{*,(2)}\|_2^2 - \mathbb{E}(\|\mathbf{b}^{*,(2)}\|_2^2) \geq \eta\frac{2\{(2-\delta)q\}^2}{L}\sum_{k,k_1,k_2}\left|\Theta^{*,(2)}_{kk_1}\Theta^{*,(2)}_{kk_2}\right|\right) \leq \exp\left(-\frac{L\eta^2}{36(2-\delta)^2q^2(18+\eta)}\right).$$

Putting things together, we have

$$\begin{aligned}\mathbb{P}\left\{\|\mathbf{b}^*\|_2^2 \geq (1+\eta)\frac{(2-\delta)q}{L}\ell^2(\boldsymbol{\Theta})\right\} &\leq \mathbb{P}\left\{\left(\|\mathbf{b}^{*,(1)}\|_2 + \|\mathbf{b}^{*,(2)}\|_2\right)^2 \geq (1+\eta)\frac{(2-\delta)q}{L}\ell^2(\boldsymbol{\Theta})\right\} \\ &\leq \mathbb{P}\left(\|\mathbf{b}^{*,(1)}\|_2^2 \geq (1+\eta)\frac{(2-\delta)q\|\boldsymbol{\theta}^{*(1)}\|_2^2}{L}\right) \\ &\quad + \mathbb{P}\left(\|\mathbf{b}^{*,(2)}\|_2^2 - \mathbb{E}(\|\mathbf{b}^{*,(2)}\|_2^2) \geq \eta\frac{2\{(2-\delta)q\}^2}{L}\sum_{k,k_1,k_2}\left|\Theta^{*,(2)}_{kk_1}\Theta^{*,(2)}_{kk_2}\right|\right) \\ &\leq \exp\left(-\frac{L\eta^2}{18(2-\delta)q\{18(2-\delta)+\eta\}}\right) + \exp\left(-\frac{L\eta^2}{36(2-\delta)^2q^2(18+\eta)}\right),\end{aligned}$$

using Lemma 28 in the final line. □

Lemma 30. *Let $(a_i)_{i=1}^\infty$ and $(b_i)_{i=1}^\infty$ be two sequences of non-negative, non-increasing, real numbers such that that there is some $i^* \in \mathbb{N}$ for which*

$$\begin{aligned}a_i &\leq b_i \quad \textit{for all } i \leq i^*, \\ a_i &\geq b_i \quad \textit{for all } i > i^*.\end{aligned}$$

(i) If

$$\sum_{i=1}^{\infty} a_i \le \sum_{i=1}^{\infty} b_i < \infty,$$

then

$$\sum_{i=1}^{\infty} a_i^2 \le \sum_{i=1}^{\infty} b_i^2.$$

(ii) If $(c_i)_{i=1}^{\infty}$ is a sequence of non-negative, non-decreasing real numbers and

$$\sum_{i=1}^{\infty} b_i \le \sum_{i=1}^{\infty} a_i < \infty, \ \sum_{i=1}^{\infty} c_i a_i \,, \sum_{i=1}^{\infty} c_i b_i < \infty,$$

then

$$\sum_{i=1}^{\infty} c_i a_i \ge \sum_{i=1}^{\infty} c_i b_i.$$

Proof. For (i), observe that

$$\begin{aligned}\sum_{i=1}^{i^*}(b_i^2 - a_i^2) = \sum_{i=1}^{i^*}(b_i + a_i)(b_i - a_i) &\ge (b_{i^*} + a_{i^*}) \sum_{i=1}^{i^*}(b_i - a_i) \\ &\ge (b_{i^*} + a_{i^*}) \sum_{i>i^*}(a_i - b_i) \ge \sum_{i>i^*}(b_i + a_i)(a_i - b_i) = \sum_{i>i^*}({a_i}^2 - {b_i}^2).\end{aligned}$$

For (ii) we argue,

$$\sum_{i=1}^{i^*} c_i(b_i - a_i) \le c_{i^*} \sum_{i=1}^{i^*}(b_i - a_i) \le c_{i^*} \sum_{i>i^*}(a_i - b_i) \le \sum_{i>i^*} c_i(a_i - b_i). \qquad \square$$

Lemma 31. *Let $q, p \in \mathbb{N}$ with $q \ge 1$, $p \ge \max\{q, 3\}$. We have*

$$\sum_{\ell=1}^{p-1} \ell \left(\frac{\binom{p-1-\ell}{q-1}}{\binom{p-1}{q}} \right)^2 \ge \frac{1}{2(2-q/p)^2} \frac{p}{p-1}.$$

Proof. Let the sequences $(a_\ell)_{\ell=1}^{\infty}$ and $(b_\ell)_{\ell=1}^{\infty}$ be defined by

$$a_\ell = \begin{cases} \left(\frac{\binom{p-1-\ell}{q-1}}{\binom{p-1}{q}} \right)^2 & \text{if } 1 \le \ell \le p-1 \\ 0 & \text{otherwise,} \end{cases} \qquad b_\ell = \begin{cases} \left(\frac{q}{p-1} \right)^2 & \text{if } \ell \le \left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor \\ \frac{q}{2(p-1)-q} - \left(\frac{q}{p-1} \right)^2 \left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor & \text{if } \ell = \left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor + 1 \\ 0 & \text{otherwise.} \end{cases}$$

Let the sequence $(c_\ell)_{\ell=1}^{\infty}$ be defined by $c_\ell = \ell$. By (4.23), the sequences $(a_\ell)_{\ell=1}^{\infty}$, $(b_\ell)_{\ell=1}^{\infty}$ and $(c_\ell)_{\ell=1}^{\infty}$ satisfy the hypotheses of Lemma 30. Thus

$$\sum_{\ell=1}^{p-1} \ell a_\ell \ge \sum_{\ell=1}^{p-1} \ell b_\ell,$$

and

$$\sum_{\ell=1}^{p-1} \ell b_\ell = \frac{1}{2}\left(\frac{q}{p-1}\right)^2 \left(\left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor + 1\right) \left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor + \left(\frac{q}{p-1}\right)^2 \left(\frac{(p-1)^2}{\{2(p-1)-q\}q} - \left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor\right) \left(\left\lfloor \frac{(p-1)^2}{\{2(p-1)-q\}q} \right\rfloor + 1\right).$$

Letting $x = (p-1)^2/[\{2(p-1)-q\}q]$, we have

$$\begin{aligned}\sum_{\ell=1}^{p-1} \ell b_\ell &= \frac{1}{2}(\lfloor x \rfloor + 1)\lfloor x \rfloor + (x - \lfloor x \rfloor)(\lfloor x \rfloor + 1) \\ &= \frac{1}{2}x(x+1) - \frac{1}{2}\{(x - \lfloor x \rfloor)\lfloor x \rfloor + (x - \lfloor x \rfloor)(x+1)\} + (x - \lfloor x \rfloor)(\lfloor x \rfloor + 1).\end{aligned}$$

Since $1 \geq 1/2 + (x - \lfloor x \rfloor)/2$, we see that

$$(x - \lfloor x \rfloor)(\lfloor x \rfloor + 1) \geq \frac{1}{2}(x - \lfloor x \rfloor)(x + 1 + \lfloor x \rfloor),$$

so

$$\begin{aligned}\sum_{\ell=1}^{p-1} \ell b_\ell &\geq \frac{1}{2}x(x+1) \\ &= \frac{1}{2}\left(\frac{(p-1)^2}{\{2(p-1)-q\}q} + 1\right)\frac{q}{2(p-1)-q} \\ &= \frac{1}{2(p-1)}\frac{p + \{2 - q/(p-1)\}q - 1}{\{2 - q/(p-1)\}^2} \\ &\geq \frac{1}{2(p-1)}\frac{p}{\{2 - q/(p-1)\}^2} \\ &\geq \frac{1}{2(2-q/p)^2}\frac{p}{p-1}.\end{aligned}$$

□

References

R. Agrawal, R. Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases*, volume 1215, pages 487–499, 1994. 54

D. Aha, D. Kibler, and M. Albert. Instance-based learning algorithms. *Machine learning*, 6:37–66, 1991. 64

U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A.J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Cell Biology*, 96:6745–6750, 1999. 13

D. Amaratunga and J. Cabrera. *Exploration and analysis of DNA microarray and protein array data*. Wiley series in probability and statistics. Wiley-Interscience, 2004. 13

A. Asuncion and D.J. Newman. UCI Machine Learning Repository, 2007. URL `http://archive.ics.uci.edu/ml`. 41

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Structured sparsity through convex optimization. *Statistical Science*, 27:450–468, 2012a. 28

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4:1–106, 2012b. 28

F.R. Bach. Bolasso: model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th International Conference on Machine learning*, pages 33–40, 2008. 2, 4

G. Biau, F. Crou, and A. Guyader. On the rate of convergence of the bagged nearest neighbor estimate. *Journal of Machine Learning Research*, 11:687–712, 2010. 4

P.J. Bickel, Y. Ritov, and A.B. Tsybakov. Hierarchical selection of variables in sparse high-dimensional regression. *IMS Collections*, 6:56–69, 2010. 28

J. Bien, J. Taylor, and R. Tibshirani. A lasso for hierarchical interactions. *Annals of Statistics*, 41 (3):1111–1141, 2013. 28, 41

B. Bollobás. *Combinatorics*. Cambridge University Press, 1986. 38

A. Bosch, A. Zisserman, and X. Muoz. Image classification using random forests and ferns. In *IEEE 11th International Conference on Computer Vision, 2007*, pages 1–8. IEEE, 2007. 70

L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996. 1, 4, 28, 89

L. Breiman. Using adaptive bagging to debias regressions. Technical report, University of California, Berkeley, 1999. 1

L. Breiman. Random Forests. *Machine Learning*, 45:5–32, 2001. 28, 41, 48, 54, 69, 86, 89, 90

L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, 1984. 54, 67

A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 327–336. ACM, 1998. 62

P. Bühlmann. Boosting for high-dimensional linear models. *Annals of Statistics*, 34:559–583, 2006. 90

P. Bühlmann. Statistical significance in high-dimensional linear models. *Bernoulli*, 19:1212–1242, 2013. 2

P. Bühlmann and S. van de Geer. *Statistics for high-dimensional data*. Springer, 2011a. 27, 85

P. Bühlmann and S.A. van de Geer. On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2011b. 44, 48

P. Bühlmann and B. Yu. Analyzing Bagging. *The Annals of Statistics*, 30:927–961, 2002. 1, 89

J.L. Carter and M.N. Wegman. Universal classes of hash functions. *Journal of computer and system sciences*, 18:143–154, 1979. 80

E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. Ullman, and C. Yang. Finding interesting associations without support pruning. *IEEE Transactions on Knowledge and Data Engineering*, 13:64–78, 2001. 62

M. L. Cule, R. J. Samworth, and M. I. Stewart. Maximum likelihood estimation of a multi-dimensional log-concave density. *Journal of the Royal Statistical Society, Series B*, 72:545–607, 2010. 7

M. Datar and S. Muthukrishnan. Estimating rarity and similarity over data stream windows. *Lecture Notes in Computer Science*, 2461:323, 2002. 62

S. Dharmadhikari and K. Joag-Dev. *Unimodality, Convexity and Applications*. Academic Press, 1988. 8

P. Drineas, M.W. Mahoney, and S. Muthukrishnan. Sampling algorithms for ℓ_2 regression and applications. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1127–1136. ACM, 2006. 77

P. Drineas, M.W. Michael W Mahoney, S. Muthukrishnan, and T. Sarls. Faster least squares approximation. *Numerische Mathematik*, 117:219–249, 2011. 77

S. Dudoit, J. Fridlyand, and T.P. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, 97: 77–87, 2002. 13

L. Dümbgen and K. Rufibach. Maximum likelihood estimation of a log-concave density and its distribution function: Basic properties and uniform consistency. *Bernoulli*, 15:48–60, 2009. 7

L. Dümbgen, R. J. Samworth, and D. Schuhmacher. Stochastic search for semiparametric linear regression models. In *From Probability to Statistics and Back: High-Dimensional Models and ProcessesA Festschrift in Honor of Jon Wellner*, volume 6, pages 78–90. Institute of Mathematical Statistics, 2012. 4

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least Angle Regression. *Annals of Statistics*, 32:407–451, 2004. 28, 33, 34, 84

J. Fan and J. Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society, Series B*, 70:849–911, 2008. 84

J. Fan and J. Lv. A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 20:101–148, 2010. 2

J. Fan, R. Samworth, and Y. Wu. Ultrahigh dimensional feature selection: beyond the linear model. *Journal of Machine Learning Research*, 10:2013–2038, 2009. 4

M. Fanty and R. Cole. Spoken letter recognition. In R.P. Lippman, J. Moody, and D.S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 13, San Mateo, CA, 1991. Morgan Kaufmann. 42

J. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19:1–67, 1991. 28, 38, 41

J. Friedman and B. Popescu. Predictive learning via rule ensembles. *Annals of Applied Statistics*, 2:916–954, 2008. 54, 66, 86

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33:1–22, 2010. 12, 33, 37, 69, 94

D. J. H. Garling. *Inequalities: A Journey into Linear Analysis*. Cambridge University Press, 2007. 104

P. Hall and R. J. Samworth. Properties of bagged nearest neighbour classifiers. *Journal of the Royal Statistical Society, Series B*, 67:363–379, 2005. 4

J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *SIGMOD Rec.*, 29(2):1–12, May 2000. 54

Y. Han and L. Yu. A variance reduction framework for stable feature selection. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 206–215, 2010. 2

Trevor Hastie, Robert Tibshirani, Friedrich Leisch, Kurt Hornik, and Brian D. Ripley. *mda: Mixture and flexible discriminant analysis*, 2013. URL `http://CRAN.R-project.org/package=mda`. R package version 0.4-4. 41

A.E. Hoerl and R.W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970. 84

A. Javanmard and A. Montanari. Confidence intervals and hypothesis testing for high-dimensional regression. *arXiv preprint arXiv:1306.3171*, 2013. 2

Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, and Francis Bach. Proximal Methods for Hierarchical Sparse Coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011. 28

W.B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984. 81

I.T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer New York, 1986. 81

A. Kalousis, J. Prados, and M. Hilario. Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and Information Systems*, 12:95–116, 2007. 2

R. Koenker and I. Mizera. Quasi-concave density estimation. *Annals of Statistics*, 38:2998–3027, 2010. 8

S. Kogan, D. Levin, B. Routledge, J. Sagi, and N. Smith. Predicting risk from financial reports with regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 272–280. Association for Computational Linguistics, 2009. 92

L. Kuncheva. A stability index for feature selection. In *Proceedings of the 25th International Multi-Conference on Artificial Intelligence and Applications*, pages 390–395, Calgary, 2007. ACTA. 2

T. Lange, M. Braun, V. Roth, and J. Buhmann. Stability-based model selection. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in neural information processing systems*, volume 15, Cambridge, MA, 2003. MIT Press. 2

D. Lewis, Y. Yang, T. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004. 67

P. Li and A.C. König. Theory and applications of b-bit minwise hashing. *Communications of the ACM*, 54:101–109, 2011. i, 78, 80, 86, 95

P. Li, A. Shrivastava, and C. König. Training logistic regression and svm on 200gb data using b-bit minwise hashing and comparisons with vowpal wabbit (VW). *arXiv:1108.3072*, 2011. 78

P. Li, A. Owen, and C.-H. Zhang. One Permutation Hashing. In *Advances in Neural Information Processing Systems 25*, pages 3122–3130, 2012a. 80

P. Li, A. Shrivastava, and C. König. *b*-Bit Minwise Hashing in Practice: Large-Scale Batch and Online Learning and Using GPUs for Fast Preprocessing with Simple Hash Functions. *arXiv preprint arXiv:1205.2958*, 2012b. 80

Y. Lin and H.H. Zhang. Component selection and smoothing in multivariate nonparametric regression. *Annals of Statistics*, 35:2272–2297, 2006. 28

Richard Lockhart, Jonathan Taylor, Ryan Tibshirani, and Robert Tibshirani. A significance test for the lasso. *Annals of Statistics, To appear*, 2013. 2

W. Loh and Y. Shih. Split selection methods for classification trees. *Statistica Sinica*, 7:815–840, 1997. 54

Steven Loscalzo, Lei Yu, and Chris Ding. Consensus group stable feature selection. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 567–576, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9. doi: 10.1145/1557019.1557084. URL `http://doi.acm.org/10.1145/1557019.1557084`. 2

J. Ma, L.K. Saul, S. Savage, and G.M. Voelker. Identifying suspicious URLs: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 681–688. ACM, 2009. 93, 94

O. Maillard and R. Munos. Linear regression with random projections. *Journal of Machine Learning Research*, 13:2735–2772, 2012. 83

M. Marchand and M. Sokolova. Learning with decision lists of data-dependent features. *Journal of Machine Learning Research*, 6:427, 2006. 54, 66

C. Matheus and L. Rendell. Constructive induction on decision trees. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, volume 645650. Citeseer, 1989. 64

L. Meier, S. van de Geer, and P. Bühlmann. High-dimensional additive modelling. *Annals of Statistics*, 37:3779–3821, 2009. 38

N. Meinshausen. Relaxed Lasso. *Computational Statistics and Data Analysis*, 52:374–393, 2007. 34

N. Meinshausen. Node Harvest. *Annals of Applied Statistics*, 4:2049–2072, 2010. 54

N. Meinshausen. Assumption-free confidence intervals for groups of variables in sparse high-dimensional regression. *arXiv preprint arXiv:1309.3489*, 2013. 2

N. Meinshausen and P. Bühlmann. High dimensional graphs and variable selection with the Lasso. *Annals of Statistics*, 34:1436–1462, 2006. 44

N. Meinshausen and P. Bühlmann. Stability selection. *Journal of the Royal Statistical Society, Series B*, 72:417–473, 2010. i, 1, 4, 5, 7, 10, 12, 13

M. Osborne, B. Presnell, and B. Turlach. A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*, 20:389–403, 2000a. 33

M.R. Osborne, B. Presnell, and B.A. Turlach. On the LASSO and its Dual. *Journal of Computational and Graphical Statistics*, 9:319–337, 2000b. 33

Jian Pei, Jiawei Han, Hongjun Lu, Shojiro Nishio, Shiwei Tang, and Dongqing Yang. H-mine: Hyper-structure mining of frequent patterns in large databases. In Nick Cercone, Tsau Young Lin, and Xindong Wu, editors, *ICDM*, pages 441–448. IEEE Computer Society, 2001. 54

R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. URL `http://www.R-project.org`. ISBN 3-900051-07-0. 12, 69

P. Radchenko and G. James. Variable selection using adaptive nonlinear interaction structures in high dimensions. *Journal of the American Statistical Association*, 105:1541–1553, 2010. 28, 38

P. Ravikumar, J. Lafferty, H. Liu, and L. Wasserman. Sparse additive models. *Journal of the Royal Statistical Society, Series B*, 71:1009–1030, 2009. 38

R. Rivest. Learning decision lists. *Machine learning*, 2:229–246, 1987. 54, 66

Yvan Saeys, Thomas Abeel, and Yves Peer. Robust feature selection using ensemble feature selection techniques. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5212 of *Lecture Notes in Computer Science*, pages 313–325. Springer Berlin Heidelberg, 2008. 2

R. J. Samworth. Optimal weighted nearest neighbour classifiers. *Annals of Statistics*, 30(5):2733–2763, 2012. 4

A. Seregin and J. A. Wellner. Nonparametric estimation of convex-transformed densities. *Annals of Statistics*, 38:3751–3781, 2010. 8

R. D. Shah and R. J. Samworth. Variable selection with error control: another look at stability selection. *Journal of the Royal Statistical Society, Series B*, 75:55–80, 2013. iii

M. Steele. *The Cauchy–Schwarz Master Class*. Cambridge University Press, 2004. 99

C. Strobl, A. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis. Conditional variable importance for random forests. *BMC Bioinformatics*, 9:307, 2008. 54, 67

R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996. i, 2, 12, 27, 69, 84

J.A. Tropp. Greed is good: algorithmic results for sparse approximation. *Information Theory, IEEE Transactions on*, 50:2231–2242, 2004. 90

B. Turlach. Discussion of 'Least angle regression'. *Annals of Statistics*, 32:481–490, 2004. 28

S. van de Geer, P. Bühlmann, and Y. Ritov. On asymptotically optimal confidence regions and tests for high-dimensional models. *arXiv preprint arXiv:1303.0518*, 2013. 2

S.A. van de Geer. High-dimensional generalized linear models and the lasso. *The Annals of Statistics*, 36:614–645, 2008. 90

A.W. van der Vaart and J.A. Wellner. *Weak Convergence and Empirical Processes*. Springer-Verlag, 1996. 104

M.J. Wainwright. Sharp thresholds for high-dimensional and noisy recovery of sparsity. *IEEE Transactions on Information Theory*, 55:2183–2202, 2009. 44, 48, 49

G. Walther. Detecting the presence of mixing with multiscale maximum likelihood. *Journal of the American Statistical Association*, 97:508–513, 2002. 7

J. Wu, B. Devlin, S. Ringquist, M. Trucco, and K. Roeder. Screen and clean: a tool for identifying interactions in genome-wide association studies. *Genetic Epidemiology*, 34:275–285, 2010. 28

H.F. Yu, C.-J. Hsieh, K.-W. Chang, and C.-J. Lin. Large linear classification when data cannot fit in memory. *ACM Transactions on Knowledge Discovery from Data*, 5:23, 2012. 78

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006. 37

M. Yuan, V. R. Joseph, and Y. Lin. An efficient variable selection approach for analyzing designed experiments. *Technometrics*, 49:430–439, 2007. 28

M. Yuan, R. Joseph, and H. Zou. Structured variable selection and estimation. *Annals of Applied Statistics*, 3:1738–1757, 2009. 28

C.-H. Zhang and S. S. Zhang. Confidence intervals for low-dimensional parameters with high-dimensional data. *Journal of the Royal Statistical Society, Series B, To appear*, 2013. 2

P. Zhao and B. Yu. On Model Selection Consistency of Lasso. *Journal of Machine Learning Research*, 7:2541–2563, 2006. 44

P. Zhao, G. Rocha, and B. Yu. The composite absolute families penalty for grouped and hierarchical variable selection. *Annals of Statistics*, 37:3648–3497, 2009. 28, 38

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006. 34, 44