Penalized Regression Methods for Interaction and Mixed-Effects Models with Applications to Genomic and Brain Imaging Data

Sahir Rai Bhatnagar

Doctor of Philosophy

Department of Epidemiology, Biostatistics and Occupational Health

McGill University
Montréal, Québec, Canada
July 2018

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Doctor of Philosophy

Dedication

This thesis is dedicated to my family, Dadi, Papa, Maa, Sameer, Marie-Pierre, Louis, Mathieu, Chandni, Amir, Navdeep, Carlos, Gloria, and Karen.

Acknowledgements

I am most grateful to Celia Greenwood and Yi Yang for the supervision of this thesis, their advice and guidance not only in professional issues, but also in all other fundamental aspects.

Preface & Contribution of Authors

Manuscript 1: Bhatnagar SR, Yang Y, Khundrakpam B, Evans A, Blanchette M, Bouchard L, Greenwood CMT (2017). An analytic approach for interpretable predictive models in high dimensional data, in the presence of interactions with exposures. Genetic Epidemiology. Apr 1;42(3):233-49. DOI 10.1002/gepi.22112.

Software: https://cran.r-project.org/package=eclust

SRB, CMTG, YY, and MB contributed to the conceptualization of this research; SRB, LB, and BK contributed to the data curation; SRB contributed to the formal analysis, software, visualization; SRB and CMTG contributed to the methodology; SRB and CMTG contributed to writing the original draft; All authors contributed to editing the draft.

Manuscript 2: Bhatnagar SR, Yang Y, Lovato A, Greenwood CMT (2018+). Sparse Additive Interaction Models.

Software: https://github.com/sahirbhatnagar/sail

Manuscript 3: Bhatnagar SR, Oualkacha K, Yang Y, Forest M, Greenwood CMT (2018+). A General Framework for Variable Selection in Linear Mixed Models with Applications to Genetic Studies with Structured Populations.

Software: https://github.com/sahirbhatnagar/ggmix

Abstract

In high-dimensional (HD) data, where the number of covariates (p) greatly exceeds the number of observations (n), estimation can benefit from the bet-on-sparsity principle, i.e., only a small number of predictors are relevant in the response. This assumption can lead to more interpretable models, improved predictive accuracy, and algorithms that are computationally efficient. In genomic and brain imaging studies, where the sample sizes are particularly small due to high data collection costs, we must often assume a sparse model because there isn't enough information to estimate p parameters. For these reasons, penalized regression methods such as the lasso and group-lasso have generated substantial interest since they can set model coefficients exactly to zero. In the penalized regression framework, many approaches have been developed for main effects. However, there is a need for developing interaction and mixed-effects models. Indeed, accurate capture of interactions may hold the potential to better understand biological phenomena and improve prediction accuracy since they may reflect important modulation of a biological system by an external factor. Furthermore, penalized mixed-effects models that account for correlations due to groupings of observations can improve sensitivity and specificity. This thesis is composed primarily of three manuscripts. The first manuscript describes a novel strategy called `eclust` for dimension reduction that leverages the effects of an exposure variable with broad impact on HD measures. With `eclust`, we found improved prediction and variable selection performance compared to methods that do not consider the exposure in the clustering step, or to methods that use the original data as features. We further illustrate this modeling framework through the analysis of three data sets from very different fields, each with HD data, a binary exposure, and a phenotype of interest. In the second manuscript, we propose a method called `sail` for detecting non-linear interactions that automatically enforces the strong heredity property using both the ℓ_1 and ℓ_2 penalty functions. We describe a blockwise coordinate descent procedure for solving the objective function and provide performance metrics on both simulated and real data. The third manuscript develops a general penalized mixed model

framework to account for correlations in genetic data due to relatedness called `ggmix`. Our method can accommodate several sparsity-inducing penalties such as the lasso, elastic net and group lasso and also readily handles prior annotation information in the form of weights. Our algorithm has theoretical guarantees of convergence and we again assess its performance in both simulated and real data. We provide efficient implementations of all our algorithms in open source software.

Abrégé

Il est aujourd'hui possible

Table of contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis, we consider the prediction of an outcome variable y observed on n individuals from p variables, where p is much larger than n. Challenges in this high-dimensional (HD) context include not only building a good predictor which will perform well in an independent dataset, but also being able to interpret the factors that contribute to the predictions. This latter issue can be very challenging in ultra-high dimensional predictor sets. For example, multiple different sets of covariates may provide equivalent measures of goodness of fit (J. Fan et al., 2014), and therefore how does one decide which are important?

When $p >> n$, standard generalized linear models (GLMs) methodology cannot uniquely estimate the unknown coefficient vector β. Moreover, even when $p \leq n$ with p close to n, standard errors of GLMs are likely to be inflated and parameter estimates unstable (Reid et al., 2016). In these instances it may be useful to assume the *Bet on Sparsity Principle* which says:

> Use a procedure that does well in sparse problems, since no procedure does well in dense problems (J. Friedman et al., 2001).

In fact, sometimes this assumption is necessary since we often do not have a large enough sample size to estimate so many parameters. Even when we do, we might want to identify a

relatively small number of predictors that play an important role on the response. Applied researchers in the health sciences might prefer a simpler model because it can shed light on the etiology of disease. The sparsity assumption may also result in faster computations and more stable predictions on new datasets.

With the advent of high-throughput technologies in genomics and brain imaging studies, computational approaches to variable selection have become increasingly important. Broadly speaking, there are three main approaches to analyze such HD data: 1) univariate regression followed by a filtering step often based on multiple testing corrections, and then subsequently a multivariable model using the reduced set of variables 2) multivariable penalized regression and 3) dimension reduction followed by a multivariable regression.

In this thesis, we focus on the use of penalized regression methods for variable selection and prediction in HD settings. In Chapter 2, we provide a critical review of the literature on penalized regression methods and the computational algorithms used to fit these models. In Chapter 3, we develop the sparse additive interaction learning model `sail` for detecting non-linear interactions with a key environmental or exposure variable in HD data. In Chapter 4, we develop a general penalized linear mixed model framework called `ggmix` that simultaneously selects and estimates variables while accounting for between individual correlations in one step. Chapter 5 explores whether use of exposure-dependent clustering relationships in dimension reduction can improve predictive modelling in a two-step framework that we develop called `eclust`. Chapters 3, 4 and 5 were originally written as stand-alone papers and as a result, there is some inconsistency in notation and overlap with the literature review chapter. Chapter 5 has been published in *Genetic Epidemiology*. Chapters 3 and 4 will be submitted for publication shortly after the submission of the thesis. We have published open source and freely available `R` packages for each of the methods developed Chapters 3, 4 and 5 (available at https://github.com/greenwoodlab). Table 1.1 provides an overview of our software packages including some of their key characteristics. In Chapter 6, we conclude

with an overview of the three manuscripts.

Table 1.1: Overview of our software packages for the three methods developped in the thesis. **Model** describes the type of loss function that has been developped in the thesis for a given method. **Penalty** are the penalty functions that can applied to the loss function. **Feature** describes the defining characteristics of the method. **Data** describes the inputs required for these methods, where x is the design matrix, y is the response, e is a single exposure variable and $\mathbf{\Psi}$ is an empirical covariance matrix. See Chapter 2 for more details.

	`eclust`	`sail`	`ggmix`
Model			
Least-Squares	✓	✓	✓
Binary Classification	✓		
Penalty			
Ridge	✓		✓
Lasso	✓	✓	✓
Elastic Net	✓		✓
Group Lasso		✓	✓
Feature			
Interactions	✓	✓	
Flexible Modeling	✓	✓	
Random Effects			✓
Data	(x, y, e)	(x, y, e)	$(x, y, \mathbf{\Psi})$

Chapter 2

Literature Review

The Literature Review is comprised of five sections. The first is a description of three general analytic strategies for high-dimensional data. The second and third sections describe two penalization methods that this thesis builds upon, namely the lasso and the group lasso. For each method we detail the algorithms used to fit these models and their convergence properties. In the fourth section we introduce penalized interaction models. This is followed by a brief introduction to linear mixed-effects models.

2.1 High-dimensional regression methods

We briefly introduce three of the main analytic strategies used to analyze HD data using the following notation. For n observations and p covariates, consider the multiple linear regression model

$$\boldsymbol{y} = \beta_0 + \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{2.1}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the response, $\mathbf{X} \in \mathbb{R}^{n \times p}$ is the design matrix, $\beta_0 \in \mathbb{R}$ is the intercept, $\boldsymbol{\beta} \in \mathbb{R}^p$ is the coefficient vector corresponding to $\mathbf{X}$ and $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ is a vector of iid random errors.

2.1.1 Univariate regression

Genome-wide association studies (GWAS) have become the standard method for analyzing genetic datasets. A GWAS consists of a series of univariate regressions followed by a multiple testing correction. This approach is simple and easy to implement, and has successfully identified thousands of genetic variants associated with complex diseases (https://www.genome.gov/gwastudies/). Despite these impressive findings, the discovered markers have only been able to explain a small proportion of the phenotypic variance known as the missing heritability problem (Manolio et al., 2009). One plausible explanation is that there are many causal variants that each explain a small amount of variation with small effect sizes (J. Yang et al., 2010). GWAS are likely to miss these true associations due to the stringent significance thresholds required to reduce the number of false positives (Manolio et al., 2009). Most statistical methods for performing multiple testing adjustments assume weak dependence among the variables being tested (Leek & Storey, 2008). Dependence among multiple tests can lead to incorrect Type 1 error rates (X. Lin et al., 2013) and highly variable significance measures (Leek & Storey, 2008). Even in the presence of weakly dependent variables, adjusting for multiple tests in whole genome studies can result in low power. Furthermore, the univariate regression approach does not allow for modeling the joint effect of many variants which may be biologically more plausible (Schadt, 2009).

Univariate regression results may serve as a filtering step for subsequent multivariable models using the reduced set of predictors. Polygenic risk scores, which combine multiple genetic markers (based on a univariate filter) into a single score, have increased the explanatory power of a set of variables for predicting disease risk (Dudbridge, 2013). However, the final model may not be very good because filtering, i.e., assessing whether to keep a variable in the final model, is performed separately for each predictor. In the next section, we introduce multivariable penalized regression approaches which have been proposed to address some of these limitations.

2.1.2 Multivariable penalized regression

The least squares estimate for $\boldsymbol{\beta}$ in the linear model is given by $\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\boldsymbol{y}$. In high-dimensional data, the problem is that $\mathbf{X}^T\mathbf{X}$ is singular because the number of covariates greatly exceeds the number of subjects. For example DNA microarrays measure the expression of approximately 20,000 genes. However, due to funding constraints, the sample size is often less than a few hundred. A common solution to this problem is through penalized regression, i.e., applying a constraint on the values of $\boldsymbol{\beta}$. The problem can be formulated as finding the vector $\boldsymbol{\beta}$ that minimizes the penalized sum of squares:

$$\underbrace{\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} X_{ij}\beta_j\right)^2}_{\text{goodness of fit}} + \underbrace{\sum_{j=1}^{p} p(\beta_j;\lambda,\gamma)}_{\text{penalty}} \tag{2.2}$$

The first term of (2.2) is the squared loss of the data and can be generalized to any loss function, while the second term is a penalty which depends on non-negative tuning parameters λ and γ that control the amount of shrinkage to be applied to $\boldsymbol{\beta}$ and the degree of concavity of the penalty function, respectively. Several penalty terms have been developed in the literature. Ridge regression places a bound on the square of the coefficients (ℓ_2 penalty) (Hoerl & Kennard, 1970) which has the effect of shrinking the magnitude of the coefficients. This however does not produce parsimonious models as none of the coefficients can be shrunk to exactly 0. The Lasso (Tibshirani, 1996) overcomes this problem by placing a bound on the sum of the absolute values of the coefficients (ℓ_1 penalty) which effectively shrinks some of them to 0, thereby simultaneously performing variable selection. The Lasso, along with other forms of penalization (e.g. SCAD J. Fan & Li (2001), Fused Lasso (Tibshirani et al., 2005), Adaptive Lasso (Zou, 2006), Relaxed Lasso (Meinshausen, 2007), MCP (C.-H. Zhang, 2010)) have proven successful in many practical problems. Despite these encouraging results, such methods have low sensitivity for identifying associated variables in the presence of high empirical correlations between covariates, because only one variable tends to be se-

lected from the group of correlated or nearly linearly dependent variables (Bühlmann et al., 2013). As a consequence, there is rarely consistency on which variables are chosen from one dataset to another (e.g. in cross-validation folds). This behavior may not be desirable in high-dimensional datasets in which large sets of predictors are highly correlated and are also associated with the response since it may limit interpretability of results. The elastic net was proposed to benefit from the strengths of ridge regression's treatment of correlated variables and lasso's sparsity (Zou & Hastie, 2005). By placing both an ℓ_1 and ℓ_2 penalty on $\boldsymbol{\beta}$, the elastic net achieves model parsimony while yielding similar regression coefficients for correlated variables. These methods however do not take advantage of any (possibly known) grouping structure in the data. For example, cortical thickness measurements from magnetic resonance imaging (MRI) scans are often grouped into cortical regions of the Automated Anatomical Labelling (AAL) atlas (Tzourio-Mazoyer et al., 2002). Genes involved in the same cellular process (e.g. KEGG pathway (Kanehisa et al., 2008)) can also be placed into biologically meaningful groups. When regularizing with the ℓ_1 penalty, each variable is selected individually regardless of its position in the design matrix. Existing structures between the variables (e.g. spatial, networks, pathways) are ignored even though in many real-life applications the estimation can benefit from this prior knowledge in terms of both prediction accuracy and interpretability (Bach et al., 2012). The group lasso (Yuan & Lin, 2006) (and generalizations thereof) overcomes this problem by producing a structured sparsity (Bach et al., 2012), i.e., given a predetermined grouping of non-overlapping variables, all members of the group are either zero or non-zero. The main drawback when applying these methods is that these groups may not be known *a priori*. Even in genomics, known pathways may not be relevant to the response of interest, and the accuracy of inferring gene networks is still in its infancy.

2.1.3 Dimension reduction together with regression

Due to the unknown grouping problem, several authors have suggested a two-step procedure where they first cluster or group variables in the design matrix and then subsequently proceed to model fitting where the feature space is some summary measure of each group. This idea dates back to 1957 when Kendall (Kendall, 1957) first proposed using principal components in regression. Hierarchical clustering based on the correlation of the design matrix has also been used to create groups of genes in microarray studies and for each level of hierarchy, the cluster average was used as the new set of potential predictors in forward-backward selection (Hastie et al., 2001) or the lasso (Park et al., 2007). Bühlmann et al. (2013) proposed a bottom-up agglomerative clustering algorithm based on canonical correlations and used the group lasso on the derived clusters. There are some advantages to these methods over the ones previously mentioned in Sections 2.1.1 and 2.1.2. First, the results may be more interpretable than the traditional lasso (and related methods) because the non-zero components of the prediction model represent sets of genes as opposed to individual ones. Second, by using genes which cluster well, we bias the inputs towards correlated sets of genes which are more likely to have similar function. Third, taking a summary measure of the resulting clusters can reduce the variance in prediction (overfitting) due to the compressed dimension of the feature space. Lastly, from a practical point of view this approach is flexible and easy to implement because efficient algorithms exist for both clustering (Müllner, 2013) and model fitting (J. Friedman et al., 2010; Y. Yang & Zou, 2014).

In this context, we introduce a new two-step procedure called `eclust` (Bhatnagar et al., 2018) in Chapter 5 of the thesis. Our method is motivated by the fact that exposure variables (e.g. smoking) can alter correlation patterns between clusters of high-dimensional variables, i.e., alter network properties of the variables. However, it is not well understood whether such altered clustering is informative in prediction. In this paper, we explore whether use of exposure-dependent clustering relationships in dimension reduction can improve predictive

modelling in a two-step framework.

Now that these three general strategies for predictive modelling with $p >> n$ have been described, the next section describes several penalized methods in detail.

2.2 Lasso

For least-squares loss, the lasso estimator (Tibshirani, 1996; Zou, 2006) is defined as

$$\widehat{\boldsymbol{\beta}}(\lambda) = \underset{(\beta_0,\boldsymbol{\beta})}{\arg\min} \frac{1}{2} \sum_{i=1}^{n} w_i (y_i - \beta_0 - (\mathbf{X}\boldsymbol{\beta})_i)^2 + \lambda \sum_{j=1}^{p} v_j |\beta_j| \tag{2.3}$$

where $(\mathbf{X}\boldsymbol{\beta})_i$ is the ith element of the n-length vector $\mathbf{X}\boldsymbol{\beta}$, $\lambda > 0$ is a tuning parameter which controls the amount of regularization, w_i is a known weight for the ith observation, and v_j is the penalty factor for the jth covariate. These penalty factors are assumed to be known and allow parameters to be penalized differently. In particular, when $v_j = 1$ for $j = 1, \ldots, p$ then all parameters are regularized equally by λ, and when $v_j = 0$ the jth covariate is not penalized, i.e., it will always be included in the model. Note also that the intercept is not penalized. The estimator (2.3) simultaneously does variable selection and shrinks the regression coefficients towards 0. Depending on the choice of λ, $\widehat{\beta}_j(\lambda) = 0$ for some j's, and $\widehat{\beta}_j(\lambda)$ can be thought of as a shrunken least squares estimator (Bühlmann & Van De Geer, 2011). There are two potential benefits brought on by imposing a penalty term to the loss function. First, the least-squares estimate has low bias but a large prediction variance. Prediction accuracy can sometimes be improved by shrinking or biasing the regression coefficients towards 0. This is known as the bias-variance trade-off. Second, interpretation of the model becomes easier when only a small subset of predictors have been identified in being relevant to the response.

The lasso estimator can produce biased estimates for large coefficients (Zou, 2006). In

Section 2.2.4 we present the adaptive lasso which is a method that can mitigate this bias. It is also worth noting that (2.3) is a convex optimization problem and thus can be solved very efficiently using a block coordinate descent algorithm (J. Friedman et al., 2007; Tseng & Yun, 2009) for which we provide further details in Section 2.2.1. Other algorithms for solving this problem exist, including LARS (Efron et al., 2004) and the homotopy algorithm (Osborne et al., 2000), but these have been largely succeeded by the coordinate descent algorithm due to its speed and computational efficiency.

2.2.1 Block coordinate descent algorithm

In a series of seminal papers, Tseng laid the groundwork for a general purpose block coordinate descent algorithm (CGD) (Tseng, 2001; Tseng et al., 1988; Tseng & Yun, 2009) to minimize the sum of a smooth function f (i.e. continuously differentiable) and a separable convex function P of the form

$$Q_\lambda(\boldsymbol{\Theta}) = \arg\min_{\boldsymbol{\Theta}} \; f(\boldsymbol{\Theta}) + \lambda P(\boldsymbol{\Theta}) \tag{2.4}$$

At each iteration, the algorithm approximates $f(\boldsymbol{\Theta})$ in (2.4) by a strictly convex quadratic function and then applies block coordinate descent to generate a descent direction followed by an inexact line search along this direction (Tseng & Yun, 2009). For continuously differentiable $f(\cdot)$ and convex and block-separable $P(\cdot)$ (e.g. $P(\boldsymbol{\beta}) = \sum_i P_i(\beta_i)$), Tseng & Yun (2009) show that the solution generated by the CGD method is a stationary point of (2.4) if the coordinates are updated in a Gauss-Seidel manner, i.e., $Q_\lambda(\boldsymbol{\Theta})$ is minimized with respect to one parameter while holding all others fixed. The separability of the penalty function into a sum of functions of each individual parameter is the key to applying this algorithm to lasso type problems. Indeed, the CGD algorithm has been successfully applied in fixed effects models (J. Friedman et al., 2010; Meier et al., 2008) and linear mixed models (Schelldorfer et al., 2011). Following Tseng & Yun (2009), the general purpose CGD algorithm for

solving (2.4) is given by Algorithm 1.

Algorithm 1 Coordinate Gradient Descent Algorithm to solve (2.4)

Set the iteration counter $k \leftarrow 0$ and choose initial values for the parameter vector $\boldsymbol{\Theta}^{(0)}$

repeat

Approximate the Hessian $\nabla^2 f(\boldsymbol{\Theta}^{(k)})$ by a symmetric matix $H^{(k)}$:

$$H^{(k)} = \text{diag}\left[\min\left\{\max\left\{\left[\nabla^2 f(\boldsymbol{\Theta}^{(k)})\right]_{jj}, 10^{-2}\right\} 10^{9}\right\}\right]_{j=1,\ldots,p} \tag{2.5}$$

for $j = 1, \ldots, p$ **do**

Solve the descent direction $d^{(k)} := d_{H^{(k)}}(\Theta_j^{(k)})$

$$d_{H^{(k)}}(\Theta_j^{(k)}) \leftarrow \arg\min_d \left\{\nabla f(\Theta_j^{(k)})d + \frac{1}{2}d^2 H_{jj}^{(k)} + \lambda P(\Theta_j^{(k)} + d)\right\} \tag{2.6}$$

Choose a stepsize

$$\alpha_j^{(k)} \leftarrow \text{line search given by the Armijo rule}$$

Update

$$\widehat{\Theta}_j^{(k+1)} \leftarrow \widehat{\Theta}_j^{(k)} + \alpha_j^{(k)} d^{(k)}$$

$k \leftarrow k + 1$

until convergence criterion is satisfied

The Armijo rule is defined as follows (Tseng & Yun, 2009):

Choose $\alpha_{init}^{(k)} > 0$ and let $\alpha^{(k)}$ be the largest element of $\left\{\alpha_{init}^{k}\delta^r\right\}_{r=0,1,2,\ldots}$ satisfying

$$Q_\lambda(\Theta_j^{(k)} + \alpha^{(k)} d^{(k)}) \leq Q_\lambda(\Theta_j^{(k)}) + \alpha^{(k)}\varrho\Delta^{(k)} \tag{2.7}$$

where $0 < \delta < 1$, $0 < \varrho < 1$, $0 \leq \gamma < 1$ and

$$\Delta^{(k)} := \nabla f(\Theta_j^{(k)})d^{(k)} + \gamma(d^{(k)})^2 H_{jj}^{(k)} + \lambda P(\Theta_j^{(k)} + d^{(k)}) - \lambda P(\Theta_j^{(k)}) \tag{2.8}$$

Common choices for the constants are $\delta = 0.1$, $\varrho = 0.001$, $\gamma = 0$, $\alpha_{init}^{(k)} = 1$ for all k (Bertsekas, 1999). We use Algorithm 9 to solve the lasso estimator with least-squares loss given by (2.3).

Without loss of generality, we assume the penalty factors (v_j) are all equal to 1.

Descent Direction

For simplicity, we remove the iteration counter (k) from the derivation below.

For $\Theta_j^{(k)} \in \{\beta_1, \ldots, \beta_p\}$, let

$$d_H(\Theta_j) = \arg\min_d G(d) \tag{2.9}$$

where

$$G(d) = \nabla f(\Theta_j)d + \frac{1}{2}d^2 H_{jj} + \lambda|\Theta_j + d|$$

Since $G(d)$ is not differentiable at $-\Theta_j$, we calculate the subdifferential $\partial G(d)$ and search for d with $0 \in \partial G(d)$:

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} + \lambda u \tag{2.10}$$

where

$$u = \begin{cases} 1 & \text{if} \quad d > -\Theta_j \\ -1 & \text{if} \quad d < -\Theta_j \\ [-1, 1] & \text{if} \quad d = \Theta_j \end{cases} \tag{2.11}$$

We consider each of the three cases in (2.10) below

1. $d > -\Theta_j$

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} + \lambda = 0$$
$$d = \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}$$

Since $\lambda > 0$ and $H_{jj} > 0$, we have

$$\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}} > \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}} = d \overset{\text{def}}{>} -\Theta_j$$

The solution can be written compactly as

$$d = \text{mid}\left\{\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}, -\Theta_j, \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}\right\}$$

where $\text{mid}\{a, b, c\}$ denotes the median (mid-point) of a, b, c.

2. $d < -\Theta_j$

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} - \lambda = 0$$
$$d = \frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}$$

Since $\lambda > 0$ and $H_{jj} > 0$, we have

$$\frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}} < \frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}} = d \overset{\text{def}}{<} -\Theta_j$$

Again, the solution can be written compactly as

$$d = \text{mid}\left\{\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}, -\Theta_j, \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}\right\}$$

3. $d_j = -\Theta_j$

There exists $u \in [-1, 1]$ such that

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} + \lambda u = 0$$
$$d = \frac{-(\nabla f(\Theta_j) + \lambda u)}{H_{jj}}$$

For $-1 \leq u \leq 1$, $\lambda > 0$ and $H_{jj} > 0$ we have

$$\frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}} \leq d \overset{\text{def}}{=} -\Theta_j \leq \frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}$$

The solution can again be written compactly as

$$d = \text{mid}\left\{\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}, -\Theta_j, \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}\right\}$$

We see all three cases lead to the same solution for (2.9). Therefore the descent direction for $\Theta_j^{(k)} \in \{\beta_1, \ldots, \beta_p\}$ for the ℓ_1 penalty is given by

$$d = \text{mid}\left\{\frac{-(\nabla f(\beta_j) - \lambda)}{H_{jj}}, -\beta_j, \frac{-(\nabla f(\beta_j) + \lambda)}{H_{jj}}\right\} \tag{2.12}$$

Solution for the β parameter

If the Hessian $\nabla^2 f(\boldsymbol{\Theta}^{(k)}) > 0$ then $H^{(k)}$ defined in (2.5) is equal to $\nabla^2 f(\boldsymbol{\Theta}^{(k)})$. Using $\alpha_{init} = 1$, the largest element of $\left\{\alpha_{init}^{(k)}\delta^r\right\}_{r=0,1,2,\ldots}$ satisfying the Armijo Rule inequality is reached for $\alpha^{(k)} = \alpha_{init}^{(k)}\delta^0 = 1$. The Armijo rule update for the $\boldsymbol{\beta}$ parameter is then given by

$$\beta_j^{(k+1)} \leftarrow \beta_j^{(k)} + d^{(k)}, \qquad j = 1, \ldots, p \tag{2.13}$$

Substituting the descent direction given by (B.11) into (2.13) we get

$$\beta_j^{(k+1)} = \text{mid}\left\{\beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) - \lambda)}{H_{jj}}, 0, \beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) + \lambda)}{H_{jj}}\right\} \tag{2.14}$$

We can further simplify this expression. First, we can re-write the loss function in (2.3) as

$$g(\boldsymbol{\beta}^{(k)}) = \frac{1}{2}\sum_{i=1}^{n} w_i \left(y_i - \sum_{\ell \neq j} X_{i\ell}\beta_\ell^{(k)} - X_{ij}\beta_j^{(k)}\right)^2 \tag{2.15}$$

The gradient and Hessian are given by

$$\nabla f(\beta_j^{(k)}) \coloneqq \frac{\partial}{\partial \beta_j^{(k)}} g(\boldsymbol{\beta}^{(k)}) = -\sum_{i=1}^{n} w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} - X_{ij} \beta_j^{(k)} \right) \tag{2.16}$$

$$H_{jj} \coloneqq \frac{\partial^2}{\partial {\beta_j^{(k)}}^2} g(\boldsymbol{\beta}^{(k)}) = \sum_{i=1}^{n} w_i X_{ij}^2 \tag{2.17}$$

Substituting (2.16) and (2.17) into $\beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) - \lambda)}{H_{jj}}$

$$\begin{aligned}
&\beta_j^{(k)} + \frac{\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} - X_{ij} \beta_j^{(k)} \right) + \lambda}{\sum_{i=1}^n w_i X_{ij}^2} \\
&= \beta_j^{(k)} + \frac{\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} \right) + \lambda}{\sum_{i=1}^n w_i X_{ij}^2} - \frac{\sum_{i=1}^n w_i X_{ij}^2 \beta_j^{(k)}}{\sum_{i=1}^n w_i X_{ij}^2} \\
&= \frac{\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} \right) + \lambda}{\sum_{i=1}^n w_i X_{ij}^2}
\end{aligned} \tag{2.18}$$

Similarly, substituting (2.16) and (2.17) in $\beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) + \lambda)}{H_{jj}}$ we get

$$\frac{\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} \right) - \lambda}{\sum_{i=1}^n w_i X_{ij}^2} \tag{2.19}$$

Finally, substituting (2.18) and (2.19) into (2.14) we get

$$\begin{aligned}
\beta_j^{(k+1)} &= \text{mid} \left\{ \frac{\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} \right) - \lambda}{\sum_{i=1}^n w_i X_{ij}^2}, 0, \frac{\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} \right) + \lambda}{\sum_{i=1}^n w_i X_{ij}^2} \right\} \\
&= \frac{\mathcal{S}_\lambda \left(\sum_{i=1}^n w_i X_{ij} \left(y_i - \sum_{\ell \neq j} X_{i\ell} \beta_\ell^{(k)} \right) \right)}{\sum_{i=1}^n w_i X_{ij}^2}
\end{aligned} \tag{2.20}$$

Where $\mathcal{S}_\lambda(x)$ is the soft-thresholding operator

$$\mathcal{S}_\lambda(x) = \text{sign}(x)(|x| - \lambda)_+$$

$\text{sign}(x)$ is the signum function

$$\text{sign}(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases}$$

and $(x)_+ = \max(x, 0)$. Since there is a closed form solution, which can be computed very quickly for each parameter update (given by (2.20)), the CGD algorithm is an attractive approach for solving the lasso estimator.

2.2.2 Lambda sequence

In general, the solution to (2.3) is computed over a decreasing sequence of values for the tuning parameter λ, beginning with the smallest value λ_{max} for which the entire coefficient vector $\widehat{\boldsymbol{\beta}} = \mathbf{0}_p$ (J. Friedman et al., 2010). To determine λ_{max}, we turn to the Karush-Kuhn-Tucker (KKT) optimality conditions for (2.3). These conditions can be written as

$$\begin{gathered} \frac{1}{v_j} \sum_{i=1}^{n} w_i X_{ij} \left(y_i - \sum_{j=1}^{p} X_{ij} \hat{\beta}_j \right) = \lambda \gamma_j, \\ \gamma_j \in \begin{cases} \text{sign}(\hat{\beta}_j) & \text{if} \quad \hat{\beta}_j \neq 0 \\ [-1, 1] & \text{if} \quad \hat{\beta}_j = 0 \end{cases}, \qquad \text{for } j = 1, \ldots, p \end{gathered} \tag{2.21}$$

where γ_j is the subgradient of the function $f(x) = |x|$ evaluated at $x = \hat{\beta}_j$. From (2.21), we can solve for the smallest value of λ such that the entire vector $(\hat{\beta}_1, \ldots, \hat{\beta}_p)$ is 0. This is given by

$$\lambda_{max} = \max_j \left\{ \left| \frac{1}{v_j} \sum_{i=1}^{n} w_i X_{ij} y_i \right| \right\}, \quad j = 1, \ldots, p \tag{2.22}$$

Following J. Friedman et al. (2010), we can choose $\tau\lambda_{max}$ to be the smallest value of tuning parameters λ_{min}, and construct a sequence of K values decreasing from λ_{max} to λ_{min} on the log scale. The defaults are set to $K = 100$, $\tau = 0.01$ if $n < p$ and $\tau = 0.001$ if $n \geq p$. The optimal value of λ can be chosen using 5-fold or 10-fold cross-validation. For least-squares loss, this corresponds to choosing the λ which minimizes the mean squared error.

2.2.3 Warm starts

The way in which we have derived the sequence of tuning parameters using the KKT conditions (Section 2.2.2), allows us to exploit warm starts which has been shown to lead to computational speedups (J. Friedman et al., 2010). That is, the solution $\widehat{\boldsymbol{\Theta}}$ for λ_k is used as the initial value $\boldsymbol{\Theta}^{(0)}$ for λ_{k+1}.

2.2.4 Adaptive lasso

It has been shown that the lasso estimator can produce biased estimates for large coefficients and give inconsistent variable selection results at the optimal λ for prediction, i.e., many noise features are included in the prediction model (Zou, 2006). To overcome the bias problems of the lasso, Zou (2006) proposed the adaptive lasso which allows a different amount of shrinkage for each regression coefficient using adaptive weights. Adaptive weighting has been shown to construct oracle procedures (J. Fan & Li, 2001), i.e., asymptotically, it performs as well as if the true model were given in advance. The adaptive lasso can be described as a two-stage procedure:

1. Calculate the initial regression estimates $\widehat{\boldsymbol{\beta}}_{init}$ from (2.3)
2. Refit (2.3) using penalty factors v_j equal to $1/\left|\hat{\beta}_{init,j}\right|$ for $j = 1, \ldots, p$.

As we can see from the weights, the adaptive lasso will shrink larger coefficients less which leads to consistent variable selection results under weaker conditions than the lasso (Bühlmann

& Van De Geer, 2011). We detail the adaptive lasso procedure in Algorithm 2.

Algorithm 2 Adaptive lasso algorithm

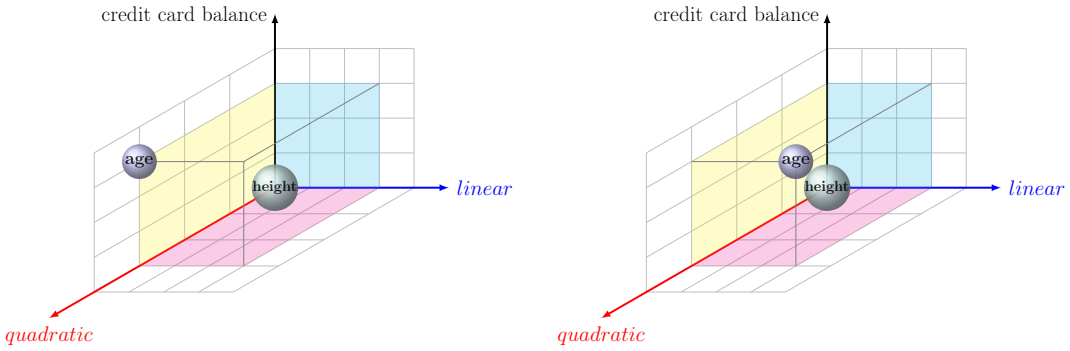
1. For a decreasing sequence $\lambda = \lambda_{max}, \ldots, \lambda_{min}$, fit the lasso with $v_j = 1$ for $j = 1, \ldots, p$
2. Use cross-validation or a data splitting procedure to determine the optimal value for the tuning parameter: $\lambda^{[opt]} \in \{\lambda_{max}, \ldots, \lambda_{min}\}$
3. Let $\hat{\beta}^{[opt]}_{init,j}$ for $j = 1, \ldots, p$ be the coefficient estimates corresponding to the model at $\lambda^{[opt]}$
4. Set the weights to be $v_j = \left(\left|\hat{\beta}^{[opt]}_{init,j}\right|\right)^{-1}$ for $j = 1, \ldots, p$
5. Refit the lasso with the weights defined in step 4), and use cross-validation or a data splitting procedure to choose the optimal value of λ

2.3 Group Lasso

One main drawback of the lasso is that it ignores the grouping structure of the design matrix. When given a predetermined grouping of non-overlapping variables, we might want the coefficients of all members of the group to be either zero or non-zero. For example, when dealing with categorical predictors where each factor is expressed through a set of indicator variables, removing an irrelevant factor is equivalent to setting the coefficients of all the indicator variables to 0. In a generalized additive model (Hastie & Tibshirani, 1987), where each variable is projected on to a set of basis function, e.g. $f_j(X_j) = \sum_{\ell=1}^{m_j} \psi_{j\ell}(X_j)\beta_{j\ell}$, we would want all $\{\beta_{j\ell}\}_{\ell=1}^{m_j}$ to be either zero or non-zero. This key difference between the lasso and group lasso penalty is illustrated in Figure 2.1. Suppose we want to predict an individual's credit card balance from their age and height using the following additive model:

$$\text{credit card balance} = \beta_0 + \beta_{11}\text{age} + \beta_{12}\text{age}^2 + \beta_{21}\text{height} + \beta_{22}\text{height}^2 + \varepsilon \tag{2.23}$$

In Figures 2.1a and 2.1b we see that both the lasso and group lasso set the linear and quadratic terms for height $(\hat{\beta}_{21}, \hat{\beta}_{22})$ to 0. However, the lasso estimates only a nonzero quadratic term for age $(\hat{\beta}_{11} = 0, \hat{\beta}_{12} \neq 0)$ while the group lasso estimates both linear and



(a) Lasso (b) Group Lasso

Figure 2.1: The selected components from Model (2.23) for (a) the lasso and (b) the group lasso. The origin of the xyz axis is zero which means both the linear and quadratic terms are 0. In this toy example, the lasso selects only the quadratic term for age while the group lasso selects both linear and quadratic terms.

quadratic terms to be nonzero ($\hat{\beta}_{11} \neq 0, \hat{\beta}_{12} \neq 0$). We now provide details on the group lasso estimator.

Assume that the predictors in the design matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ belong to K groups and define the cardinality of index set I_k to be p_k. These groups are known *a priori* such that $(1, 2, \ldots, p) = \bigcup_{k=1}^{K} I_k$, and are also non-overlapping, i.e., $I_k \bigcap I_{k'} = \emptyset$ for $k \neq k'$. Therefore, group k contains p_k predictors corresponding to $\mathbf{X}^{(k)}$, i.e., the columns of the design matrix X_j for $j \in I_k$, and $1 \leq k \leq K$. The intercept belongs to its own group, i.e., $I_1 = \{1\}$. The group lasso partitions the variable coefficients into K groups $\boldsymbol{\beta} = ([\boldsymbol{\beta}^{(1)}]^\intercal, [\boldsymbol{\beta}^{(2)}]^\intercal, \cdots, [\boldsymbol{\beta}^{(K)}]^\intercal)^\intercal$, where $\boldsymbol{\beta}^{(k)}$ denotes the segment of $\boldsymbol{\beta}$ corresponding to group k. For least-squares loss, the group lasso estimator (Yuan & Lin, 2006) is given by:

$$\widehat{\boldsymbol{\beta}}(\lambda) = \underset{(\beta_0, \boldsymbol{\beta})}{\arg\min} \frac{1}{2} \sum_{i=1}^{n} w_i (y_i - \beta_0 - (\mathbf{X}\boldsymbol{\beta})_i)^2 + \lambda \sum_{k=1}^{K} v_k \|\boldsymbol{\beta}^{(k)}\|_2 \tag{2.24}$$

where $\left\|\boldsymbol{\beta}^{(k)}\right\|_2 = \sqrt{\sum_{j \in I_k} \beta_j^2}$ and $\lambda > 0$ is the tuning parameter. As in the lasso estimator (2.3), there are both observation weights w_i, and penalty factors $v_k \geq 0$ which control the relative strength of the terms within the group lasso penalty. These penalty factors are often set to $\sqrt{p_k}$ (Yuan & Lin, 2006). Note that the same penalty factor is applied to

all the coefficients in a group. Solving the group lasso estimator is more challenging than the lasso since there is no closed form solution for (2.24). In the next section, we detail a majorization-minimization (MM) type algorithm (Lange et al., 2000; Y. Yang & Zou, 2015) used to solve (2.24).

2.3.1 Groupwise majorization descent algorithm

This description of the groupwise majorization descent (GMD) algorithm used to solve (2.24) follows mainly from Y. Yang & Zou (2015). The main difference here is that we consider a more general loss function of the form

$$L(\boldsymbol{\beta} \mid \mathbf{D}) = \frac{1}{2} [\boldsymbol{y} - \widehat{\boldsymbol{y}}]^\top \mathbf{W} [\boldsymbol{y} - \widehat{\boldsymbol{y}}] \tag{2.25}$$

where $\widehat{\boldsymbol{y}} = \hat{\beta}_0 + \mathbf{X}\widehat{\boldsymbol{\beta}}$, $\mathbf{D}$ is the working data $\{\boldsymbol{y}, \mathbf{X}\}$, and $\mathbf{W}$ is an $n \times n$ nonsingular and known weight matrix. This weight matrix can be used when the elements of $\boldsymbol{y}$ are correlated as is done in generalized least squares. The original proposal in Y. Yang & Zou (2015) is a special case of (2.25) where $\mathbf{W}$ is a diagonal matrix with entries equal to w_i. The loss function (2.25) satisfies the quadratic majorization (QM) condition, since $L(\boldsymbol{\beta} \mid \mathbf{D})$ is differentiable as a function of $\boldsymbol{\beta}$, i.e., $\nabla L(\boldsymbol{\beta}|\mathbf{D}) = -(\boldsymbol{y} - \hat{\boldsymbol{y}})^\top \mathbf{W}\mathbf{X}$, and there exists a $p \times p$ matrix $\mathbf{H} = \mathbf{X}^\top \mathbf{W}\mathbf{X}$ which only depends on the data $\mathbf{D}$, such that for all $\boldsymbol{\beta}, \boldsymbol{\beta}^*$,

$$L(\boldsymbol{\beta} \mid \mathbf{D}) \leq L(\boldsymbol{\beta}^* \mid \mathbf{D}) + (\boldsymbol{\beta} - \boldsymbol{\beta}^*)^\top \nabla L(\boldsymbol{\beta}^*|\mathbf{D}) + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}^*)^\top \mathbf{H}(\boldsymbol{\beta} - \boldsymbol{\beta}^*). \tag{2.26}$$

We can exploit the fact that the loss function (2.25) satisfies the QM condition to majorize the loss function in (2.24) by (2.26), and that the penalty term $\sum_{k=1}^{K} v_k \|\boldsymbol{\beta}^{(k)}\|_2$ in (2.24) is separable with respect to the indices of the variables $k = 1, \ldots, K$ to performs groupwise updates for each $\boldsymbol{\beta}^{(k)}$.

Let $\widetilde{\boldsymbol{\beta}}$ denote the current solution of $\boldsymbol{\beta}$ and define $\mathbf{H}^{(k)}$ as the sub-matrix of $\mathbf{H}$ corresponding to group k. For example, if group 2 is $\{2, 4\}$ then $\mathbf{H}^{(2)}$ is a 2×2 matrix with

$$\mathbf{H}^{(2)} = \begin{bmatrix} h_{2,2} & h_{2,4} \\ h_{4,2} & h_{4,4} \end{bmatrix},$$

where $h_{i,j}$ is the i, jth entry of the $\mathbf{H}$ matrix. Write $\boldsymbol{\beta}$ such that $\boldsymbol{\beta}^{(k')} = \widetilde{\boldsymbol{\beta}}^{(k')}$ for $k' \neq k$. Given $\boldsymbol{\beta}^{(k')} = \widetilde{\boldsymbol{\beta}}^{(k')}$ for $k' \neq k$, the estimator for $\boldsymbol{\beta}^{(k)}$ is given by

$$\widehat{\boldsymbol{\beta}}^{(k)}(\lambda) = \arg\min_{\boldsymbol{\beta}^{(k)}} L(\boldsymbol{\beta} \mid \mathbf{D}) + \lambda v_k \|\boldsymbol{\beta}^{(k)}\|_2. \tag{2.27}$$

Using $\boldsymbol{\beta} - \widetilde{\boldsymbol{\beta}} = (\underbrace{0, \ldots, 0}_{k-1}, \boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)}, \underbrace{0, \ldots, 0}_{K-k})$ and the majorization defined in (2.26) we can write

$$L(\boldsymbol{\beta} \mid \mathbf{D}) \leq L(\widetilde{\boldsymbol{\beta}} \mid \mathbf{D}) - (\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)})^\top U^{(k)} + \frac{1}{2}(\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)})^\top \mathbf{H}^{(k)} (\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)}). \tag{2.28}$$

where

$$U^{(k)} = \frac{\partial}{\partial \boldsymbol{\beta}^{(k)}} L(\boldsymbol{\beta} \mid \mathbf{D}) = -(\boldsymbol{y} - \hat{\boldsymbol{y}})^\top \mathbf{W} \mathbf{X}^{(k)}, \tag{2.29}$$

$$\mathbf{H}^{(k)} = \frac{\partial^2}{\partial \boldsymbol{\beta}^{(k)} \partial \boldsymbol{\beta}^{(k)\top}} L(\boldsymbol{\beta} \mid \mathbf{D}) = (\mathbf{X}^{(k)})^\top \mathbf{W} \mathbf{X}^{(k)}. \tag{2.30}$$

The upper bound in (2.28) can be majorized even further to avoid expensive matrix multiplications and storing $\mathbf{H}^{(k)}$ in memory. Let η_k be the largest eigenvalue of $\mathbf{H}^{(k)}$. We set $\gamma_k = (1 + \varepsilon^*)\eta_k$, where $\varepsilon^* = 10^{-6}$ and substitute γ_k for $\mathbf{H}^{(k)}$ in (2.28):

$$L(\boldsymbol{\beta} \mid \mathbf{D}) \leq L(\widetilde{\boldsymbol{\beta}} \mid \mathbf{D}) - (\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)})^\top U_{(k)} + \frac{1}{2}\gamma_k (\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)})^\top (\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)}). \tag{2.31}$$

It is important to note that the inequality strictly holds unless when $\boldsymbol{\beta}^{(k)} = \widetilde{\boldsymbol{\beta}}^{(k)}$. Substituting

the upper bound in (2.31) for $L(\boldsymbol{\beta} \mid \mathbf{D})$ in (2.27) we get the following estimator for $\boldsymbol{\beta}^{(k)}$

$$\arg\min_{\boldsymbol{\beta}^{(k)}} L(\widetilde{\boldsymbol{\beta}} \mid \mathbf{D}) - (\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)})^\top U^{(k)} + \frac{1}{2}\gamma_k(\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)})^\top(\boldsymbol{\beta}^{(k)} - \widetilde{\boldsymbol{\beta}}^{(k)}) + \lambda v_k \|\boldsymbol{\beta}^{(k)}\|_2. \quad (2.32)$$

Let $\widetilde{\boldsymbol{\beta}}^{(k)}(\text{new})$ be the solution to (2.32) which has a simple closed-from expression:

$$\widetilde{\boldsymbol{\beta}}^{(k)}(\text{new}) = \frac{1}{\gamma_k}\left(U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\right)\left(1 - \frac{\lambda v_k}{\|U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\|_2}\right)_+. \quad (2.33)$$

Algorithm 3 summarizes the details of GMD for the group lasso with least-squares loss function given by (2.25). The GMD algorithm has the strict descent property and converges to the right answer. The proof follows directly from Section 3.1 of Y. Yang & Zou (2015).

In Figure 2.2 we provide an illustration of the quadratic majorization technique for updating a parameter β_j. The solution lies at the minimum of the F curve but we cannot solve for this directly since there is no closed form solution. Instead we majorize F using Q_1 which is a function consisting of the quadratic approximation of F plus the penalty term evaluated at $\beta_j^{(m)}$. The minimum of Q_1, for which there is a closed form solution, corresponds to the next iteration $\beta_j^{(m+1)}$. We then majorize again using Q_2 and solve for the minimum. This process is repeated until convergence.

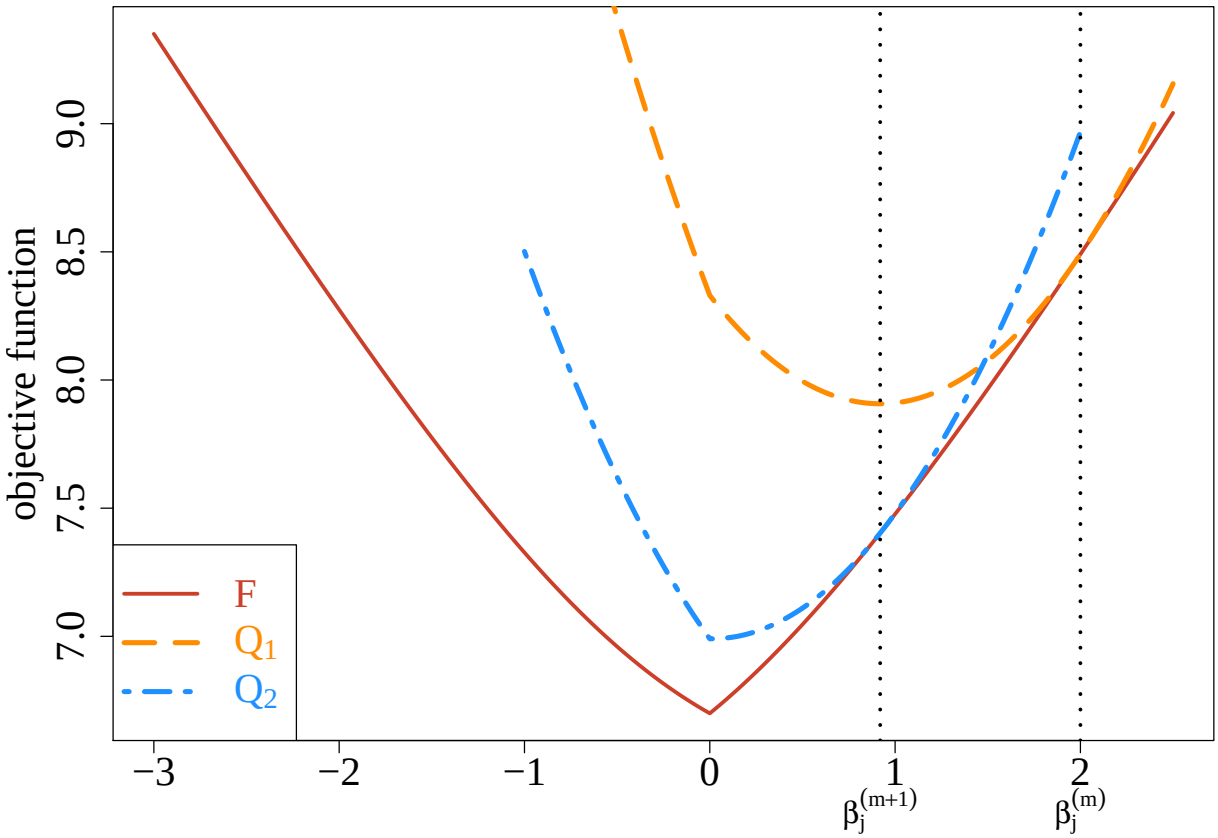


Figure 2.2: Illustration of the quadratic majorization technique for updating the coefficient β_j from the m step to the $(m+1)$ step. F is the objective function for which there is no closed form solution. Q_1 and Q_2 are the surrogate functions for which we can find the minimum.

Algorithm 3 The GMD algorithm for group lasso with least-squares loss function given by (2.25).

---

1. For $k = 1, \ldots, K$, compute γ_k, the largest eigenvalue of $\mathbf{H}^{(k)} = (\mathbf{X}^{(k)})^\top \mathbf{W} \mathbf{X}^{(k)}$
2. Initialize $\widetilde{\boldsymbol{\beta}}$.
3. Repeat the following cyclic groupwise updates until convergence:
   — for $k = 1, \ldots, K$, do step (3.1)–(3.3)
   3.1 Compute $U(\widetilde{\boldsymbol{\beta}}) = -\nabla L(\widetilde{\boldsymbol{\beta}} | \mathbf{D}) = -(\boldsymbol{y} - \hat{\boldsymbol{y}})^\top \mathbf{W} \mathbf{X}^{(k)}$
   3.2 Compute $\widetilde{\boldsymbol{\beta}}^{(k)}(\text{new}) = \frac{1}{\gamma_k}\left(U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\right)\left(1 - \frac{\lambda v_k}{\|U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\|_2}\right)_+$.
   3.3 Set $\widetilde{\boldsymbol{\beta}}^{(k)} = \widetilde{\boldsymbol{\beta}}^{(k)}(\text{new})$.

2.3.2 Lambda sequence

Similar to Section 2.2.2, we compute the solution to (2.24) over a decreasing sequence of values for the tuning parameter λ starting with λ_{max}. From the update formula (2.33) we have that for all k

$$\begin{cases} \widetilde{\boldsymbol{\beta}}^{(k)} = \frac{1}{\gamma_k}\left(U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\right)\left(1 - \frac{\lambda v_k}{\|U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\|_2}\right) & \text{if } \|U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\|_2 > \lambda v_k \\ \widetilde{\boldsymbol{\beta}}^{(k)} = \mathbf{0} & \text{if } \|U^{(k)} + \gamma_k \widetilde{\boldsymbol{\beta}}^{(k)}\|_2 \leq \lambda v_k \ . \end{cases}$$

We can then directly obtain the KKT conditions for $k = 1, \ldots, K$:

$$\begin{aligned} -U^{(k)} + \lambda v_k \cdot \frac{\widetilde{\boldsymbol{\beta}}^{(k)}}{\|\widetilde{\boldsymbol{\beta}}^{(k)}\|_2} = \mathbf{0} \qquad & \text{if } \widetilde{\boldsymbol{\beta}}^{(k)} \neq \mathbf{0}, \\ \left\|U^{(k)}\right\|_2 \leq \lambda w_k \qquad & \text{if } \widetilde{\boldsymbol{\beta}}^{(k)} = \mathbf{0} \ . \end{aligned} \tag{2.34}$$

Using (2.34) we can solve for the smallest value of λ such that the entire vector $\left\{\widehat{\boldsymbol{\beta}}^{(k)}\right\}_{k=1}^{K}$ is 0. This is given by

$$\lambda_{max} = \max_k \frac{1}{v_k}\left\|U^{(k)}\right\|_2, \quad k = 1, \ldots, K, \ v_k \neq 0 \tag{2.35}$$

2.3.3 Warm starts and adaptive group lasso

Warm starts can also be implemented for the group lasso as described in Section 2.2.3 for the lasso. Furthermore, the adaptive group lasso can be computed using Algorithm 2; the main difference is that the coefficient estimates in Step 1 are obtained from a group lasso fit and the weights for group k are given by $v_k = \left(\left\|\widehat{\boldsymbol{\beta}}^{(k)}\right\|_2\right)^{-1}$.

2.4 Penalized interaction models

In this section, we introduce penalized regression methods in the context of interaction models. We consider a regression model for an n-length outcome variable $\boldsymbol{y}$ which follows an exponential family. Let $E = (e_1, \ldots, e_n)$ be the binary environment or exposure vector and $\mathbf{X}$ the $n \times p$ matrix of high-dimensional data. Consider the regression model with main effects and their interactions with E:

$$g(\boldsymbol{\mu}) = \beta_0 + \underbrace{\beta_1 X_1 + \cdots + \beta_p X_p + \beta_E E}_{\text{main effects}} + \underbrace{\alpha_{1E}(X_1 E) + \cdots + \alpha_{pE}(X_p E)}_{\text{interactions}} \tag{2.36}$$

where $g(\cdot)$ is a known link function and $\boldsymbol{\mu} = \mathsf{E}[\boldsymbol{y}|\mathbf{X}, E, \boldsymbol{\beta}, \boldsymbol{\alpha}]$. Due to the large number of parameters to estimate with respect to the number of observations, we might consider adding the lasso penalty introduced in Section 2.2. A natural extension of the lasso penalty applied to the interaction model (2.36) is

$$\underset{\beta_0, \boldsymbol{\beta}, \boldsymbol{\alpha}}{\arg\min} \frac{1}{2} \|Y - g(\boldsymbol{\mu})\|_2^2 + \lambda \left(\|\boldsymbol{\beta}\|_1 + \|\boldsymbol{\alpha}\|_1 \right) , \tag{2.37}$$

where $\|Y - g(\boldsymbol{\mu})\|_2^2 = \sum_i (y_i - g(\mu_i))^2$, $\|\boldsymbol{\beta}\|_1 = \sum_j |\beta_j|$, $\|\boldsymbol{\alpha}\|_1 = \sum_j |\alpha_j|$ and $\lambda \geq 0$ is the tuning parameter that can set some of the coefficients to zero when sufficiently large. A limitation of (2.37) is that the selected model may have main effects that are 0 but the corresponding interaction terms are not. This is due to the penalty which treats both main effects and interactions equally. While there may exist situations where this can occur, statisticians have long argued that large main effects are more likely to lead to detectable interactions than small ones (Cox, 1984) and therefore, main effects should enter the model before interactions. This is known as the strong heredity principle (Chipman, 1996):

$$\hat{\alpha}_{jE} \neq 0 \qquad \Rightarrow \qquad \hat{\beta}_j \neq 0 \qquad \text{and} \qquad \hat{\beta}_E \neq 0 \tag{2.38}$$

In words, the interaction term will only have a non-zero estimate if its corresponding main effects are estimated to be non-zero. One benefit brought by hierarchy is that the number of measured variables can be reduced, referred to as practical sparsity (Bien et al., 2013; She & Jiang, 2014). For example, a model involving $X_1, E, X_1 \cdot E$ is more parsimonious than a model involving $X_1, E, X_2 \cdot E$, because in the first model a researcher would only have to measure two variables compared to three in the second model.

There have been several proposals for modeling interactions with the strong heredity constraint in the penalization literature including Composite Absolute Penalties (CAP) (Zhao et al., 2009), Variable selection using Adaptive Nonlinear Interaction Structures in High dimensions (VANISH) (Radchenko & James, 2010), Strong Hierarchical Lasso (hierNet) (Bien et al., 2013), Group-Lasso Interaction Network (glinternet) (Lim & Hastie, 2015), Group Regularized Estimation under Structural Hierarchy (GRESH) (She & Jiang, 2014) and a Framework for Modeling Interactions with a Convex Penalty (FAMILY) (Haris et al., 2014). A popular approach to achieve this structured sparsity is via the group lasso penalty where each group contains both main effects and their corresponding interactions. For example, consider a two-way interaction model of the form

$$\begin{aligned} y =& \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 \\ &+ \beta_{12} X_1 X_2 + \beta_{13} X_1 X_3 + \beta_{23} X_2 X_3 + \varepsilon \end{aligned}$$

A grouping structure that would satisfy the strong heredity property is:

group	1	2	3	4			5			6		
parameter	β_1	β_2	β_3	β_1	β_2	β_{12}	β_1	β_3	β_{13}	β_2	β_3	β_{23}

A limitation of this approach is that its impossible to select one variable without selecting all the groups containing it (Jacob et al., 2009). This implies that if a main effect has been selected, all the interaction terms with that main effect will necessarily be selected as well which can be an issue when the truth contains no interactions.

In Chapter 3 of the thesis, we introduce a new structured sparsity called `sail` which satisfies the strong heredity property while overcoming this issue, i.e., a selected main effect does not automatically imply that the corresponding interactions will also be selected. Furthermore, our method can accommodate non-linear interaction effects on the response. The context of Chapter 3 is summarized in Table 2.1.

Table 2.1: Overview of methods with structured sparsity for penalized regression models

Type	Method	Software
Linear	`CAP` (Zhao et al., 2009)	✗
	`SHIM` (Choi et al., 2010)	✗
	`hiernet` (Bien et al., 2013)	`hierNet(x, y)`
	`GRESH` (She & Jiang, 2014)	✗
	`FAMILY` (Haris, Witten, & Simon, 2016)	`FAMILY(x, z, y)`
	`glinternet` (Lim & Hastie, 2015)	`glinternet(x, y)`
	`RAMP` (Hao et al., 2018)	`RAMP(x, y)`
	`LassoBacktracking` (Shah, 2016)	`LassoBT(x, y)`
Non-linear	`VANISH` (Radchenko & James, 2010)	✗
	`sail` (Chapter 3 of the thesis)	`sail(x, y, e)`

2.5 Linear mixed-effects models

Linear mixed-effects models (LMMs) (Laird & Ware, 1982) are a class of statistical models used to account for correlations induced by a natural grouping of the observations such as students in schools or repeated measurements on the same individual. By introducing a subject specific random effect, LMMs can simultaneously model the mean and covariance structure to produce valid standard errors. In this section we briefly introduce LMMs as a preface to Chapter 4 of the thesis which is about penalized LMMs.

Let $i = 1, \ldots, N$ be the grouping index, $j = 1, \ldots, n_i$ the observation index within a group and $N_T = \sum_{i=1}^{N} n_i$ the total number of observations. For each group let $\boldsymbol{y}_i = (y_1, \ldots, y_{n_i})$ be the observed vector of responses, $\mathbf{X}_i$ an $n_i \times (p+1)$ design matrix (with the column of 1s for the intercept), $\boldsymbol{b}_i$ a group-specific random effect vector of length n_i and $\boldsymbol{\varepsilon}_i = (\varepsilon_{i1}, \ldots, \varepsilon_{in_i})$ the individual error terms. Denote the stacked vectors $\mathbf{Y} = (\boldsymbol{y}_i, \ldots, \boldsymbol{y}_N)^T \in \mathbb{R}^{N_T \times 1}$, $\boldsymbol{b} = (\boldsymbol{b}_i, \ldots, \boldsymbol{b}_N)^T \in \mathbb{R}^{N_T \times 1}$, $\boldsymbol{\varepsilon} = (\boldsymbol{\varepsilon}_i, \ldots, \boldsymbol{\varepsilon}_N)^T \in \mathbb{R}^{N_T \times 1}$, and the stacked matrix $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)^T \in \mathbb{R}^{N_T \times (p+1)}$. Furthermore, let $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^T \in \mathbb{R}^{(p+1)\times 1}$ be a vector of fixed effects regression coefficients corresponding to $\mathbf{X}$. We consider the following LMM with a single random effect:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{b} + \boldsymbol{\varepsilon} \tag{2.39}$$

where the random effect $\boldsymbol{b}$ and the error variance $\boldsymbol{\varepsilon}$ are assigned the distributions

$$\boldsymbol{b} \sim \mathcal{N}(0, \eta\sigma^2\boldsymbol{\Phi}) \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, (1-\eta)\sigma^2\mathbf{I}) \tag{2.40}$$

Here, $\boldsymbol{\Phi}_{N_T \times N_T}$ is a known positive semi-definite and symmetric covariance matrix, $\mathbf{I}_{N_T \times N_T}$ is the identity matrix and parameters σ^2 and $\eta \in [0, 1]$ determine how the variance is divided between $\boldsymbol{b}$ and $\boldsymbol{\varepsilon}$. The joint density of $\mathbf{Y}$ is multivariate normal:

$$\mathbf{Y}|(\boldsymbol{\beta}, \eta, \sigma^2) \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \eta\sigma^2\boldsymbol{\Phi} + (1-\eta)\sigma^2\mathbf{I}) \tag{2.41}$$

In high-dimensional contexts, the classic LMM is no longer applicable for the same reasons why the linear model does not apply, i.e., singularity of $\mathbf{X}^T\mathbf{X}$. However, naively applying a penalized method such as the lasso to clustered data is also invalid. Indeed, ignoring the grouping structure can lead to incorrect inference due to biased standard errors of parameters, and can also violate the normality of residuals assumption. In Chapter 4 of this thesis, we develop a general framework for penalized mixed models that simultaneously performs

variable selection while accounting for the correlations induced by the natural clustering of the observations.

Chapter 3

Sparse Additive Interaction Learning

Sahir Rai Bhatnagar[1,2], Amanda Lovato[2], Yi Yang[3], Celia MT Greenwood[1,2]

[1]Department of Epidemiology, Biostatistics and Occupational Health, McGill University

[2]Lady Davis Institute, Jewish General Hospital, Montréal, QC

[3]Department of Mathematics and Statistics, McGill University

Abstract

Diseases are now thought to be the result of changes in entire biological networks whose states are affected by a complex interaction of genetic and environmental factors. In general, power to estimate interactions is low, the number of possible interactions could be enormous and their effects may be non-linear. Existing approaches such as the lasso might keep an interaction but remove a main effect, which is problematic for interpretation. We develop a model for linear and non-linear interactions in penalized regression models that automatically enforces the strong heredity property. A computationally efficient fitting algorithm combined with a non-parametric screening approach scales to high-dimensional datasets and has been implemented in an R package. We apply our method to identify gene-prenatal maternal depression interactions on negative emotionality in mother–infant dyads from the Maternal Adversity, Vulnerability, and Neurodevelopment (MAVAN) cohort.

3.1 Introduction

Computational approaches to variable selection have become increasingly important with the advent of high-throughput technologies in genomics and brain imaging studies, where the data has become massive, yet where it is believed that the number of truly important variables is small relative to the total number of variables. Although many approaches have been developed for main effects, there is an enduring interest in powerful methods for estimating interactions, since interactions may reflect important modulation of a genomic system by an external factor (Bhatnagar et al., 2018). Accurate capture of interactions may hold the potential to better understanding biological phenomena and improving prediction accuracy. For example, a model that considered interactions between brain imaging data and genetic features had better classification accuracy compared to a model that considered the main effects only (Ning et al., 2018). Furthermore, the manifestations of disease are often considered to be the result of changes in entire biological networks whose states are affected by a complex interaction of genetic and environmental factors (Schadt, 2009). However, there is a general deficit of such replicated interactions in the literature (Timpson et al., 2018). Indeed, power to detect interactions is always lower than for main effects, and in high-dimensional settings ($p >> n$), this lack of power to detect interactions is exacerbated, since the number of possible interactions could be enormous and their effects may be non-linear. Hence, analytic methods that may improve power are essential.

Interactions may occur in numerous types and of varying complexities. In this paper, we consider one specific type of interaction models, where one (exposure) variable is involved in possibly non-linear interactions with a high-dimensional set of measures $\mathbf{X}$ leading to effects on a response variable, Y. We propose a multivariable penalization procedure for detecting non-linear interactions $\mathbf{X}$ and E.

3.1.1 A Sparse additive interaction model

Let $Y = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$ be a continuous outcome variable, $X_E = (E_1, \ldots, E_n) \in \mathbb{R}^n$ a binary or continuous environment vector, and $\mathbf{X} = (X_1, \ldots, X_p) \in \mathbb{R}^{n \times p}$ a matrix of predictors, possibly high-dimensional. Furthermore let $f_j : \mathbb{R} \to \mathbb{R}$ be a smoothing method for variable X_j by a projection on to a set of basis functions:

$$f_j(X_j) = \sum_{\ell=1}^{m_j} \psi_{j\ell}(X_j)\beta_{j\ell} \tag{3.1}$$

Here, the $\{\psi_{j\ell}\}_1^{m_j}$ are a family of basis functions in X_j (Hastie et al., 2015). Let $\boldsymbol{\Psi}_j$ be the $n \times m_j$ matrix of evaluations of the $\psi_{j\ell}$ and $\boldsymbol{\theta}_j = (\beta_{j1}, \ldots, \beta_{jm_j}) \in \mathbb{R}^{m_j}$ for $j = 1, \ldots, p$ ($\boldsymbol{\theta}_j$ is a m_j-dimensional column vector of basis coefficients for the jth main effect). In this article we consider an additive interaction regression model of the form

$$g(\boldsymbol{\mu}) = \beta_0 \cdot \mathbf{1} + \sum_{j=1}^{p} \boldsymbol{\Psi}_j\boldsymbol{\theta}_j + \beta_E X_E + \sum_{j=1}^{p} (X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\tau}_j \tag{3.2}$$

where $g(\cdot)$ is a known link function, $\boldsymbol{\mu} = \mathsf{E}[Y|\boldsymbol{\Psi}, X_E]$, β_0 is the intercept, β_E is the coefficient for the environment variable, $\boldsymbol{\tau}_j = (\tau_{j1}, \ldots, \tau_{jm_j}) \in \mathbb{R}^{m_j}$ are the basis coefficients for the jth interaction term, and $(X_E \circ \boldsymbol{\Psi}_j)$ is the $n \times m_j$ matrix formed by the component-wise multiplication of the column vector X_E by each column of $\boldsymbol{\Psi}_j$. For a continuous response, we use the squared-error loss to estimate the parameters:

$$\mathcal{L}(\boldsymbol{\Theta}|\mathbf{D}) = \frac{1}{2n}\left\| Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j\boldsymbol{\theta}_j - \beta_E X_E - \sum_{j=1}^{p} (X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\tau}_j \right\|_2^2 \tag{3.3}$$

and for binary response $Y_i \in \{-1, +1\}$ we use the logistic loss:

$$\mathcal{L}(\boldsymbol{\Theta}|\mathbf{D}) = \frac{1}{n}\sum_i \log\left(1 + \exp\left\{-Y_i\left(\beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j\boldsymbol{\theta}_j - \beta_E X_E - \sum_{j=1}^{p} (X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\tau}_j\right)\right\}\right) \tag{3.4}$$

where $\boldsymbol{\Theta} \coloneqq (\beta_0, \beta_E, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p, \boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_p)$ and $\mathbf{D} \coloneqq (Y, \boldsymbol{\Psi}, X_E)$ is the working data. Here we assume that p is large relative to n, and particularly that $\sum_{j=1}^{p} m_j/n$ is large. Due to the large number of parameters to estimate with respect to the number of observations, one commonly-used approach is to shrink the regression coefficients by placing a constraint on the values of $(\beta_E, \boldsymbol{\theta}_j, \boldsymbol{\tau}_j)$. Certain constraints have the added benefit of producing a sparse model in the sense that many of the coefficients will be set exactly to 0 (Bühlmann & Van De Geer, 2011). Such a reduced predictor set can lead to a more interpretable model with smaller prediction variance, albeit at the cost of having biased parameter estimates (J. Fan et al., 2014). In light of these goals, we consider the following objective function:

$$\arg\min_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta}|\mathbf{D}) + \lambda(1-\alpha)\left(w_E|\beta_E| + \sum_{j=1}^{p} w_j \|\boldsymbol{\theta}_j\|_2\right) + \lambda\alpha \sum_{j=1}^{p} w_{jE} \|\boldsymbol{\tau}_j\|_2 \tag{3.5}$$

where $\|\boldsymbol{\theta}_j\|_2 = \sqrt{\sum_{k=1}^{m_j} \beta_{jk}^2}$, $\|\boldsymbol{\tau}_j\|_2 = \sqrt{\sum_{k=1}^{m_j} \tau_{jk}^2}$, $\lambda > 0$ and $\alpha \in (0, 1)$ are adjustable tuning parameters, w_E, w_j, w_{jE} are non-negative penalty factors for $j = 1, \ldots, p$ which serve as a way of allowing parameters to be penalized differently. The first term in the penalty penalizes the main effects while the second term penalizes the interactions. The parameter α controls the relative weight on the two penalties. Note that we do not penalize the intercept.

An issue with (3.5) is that since no constraint is placed on the structure of the model, it is possible that an estimated interaction term is nonzero while the corresponding main effects are zero. While there may be certain situations where this is plausible, statisticians have generally argued that interactions should only be included if the corresponding main effects are also in the model (McCullagh & Nelder, 1989). This is known as the strong heredity principle (Chipman, 1996). Indeed, large main effects are more likely to lead to detectable interactions (Cox, 1984). In the next section we discuss how a simple reparametrization of the model (3.5) can lead to this desirable property.

3.1.2 Strong and weak heredity

The strong heredity principle states that an interaction term can only have a non-zero estimate if its corresponding main effects are estimated to be non-zero. The weak heredity principle allows for a non-zero interaction estimate as long as one of the corresponding main effects is estimated to be non-zero (Chipman, 1996). In the context of penalized regression methods, these principles can be formulated as structured sparsity (Bach et al., 2012) problems. Several authors have proposed to modify the type of penalty in order to achieve the heredity principle (Bien et al., 2013; Haris et al., 2014; Lim & Hastie, 2014; Radchenko & James, 2010). We take an alternative approach. Following Choi et al. (Choi et al., 2010), we introduce a new set of parameters $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p) \in \mathbb{R}^p$ and reparametrize the coefficients for the interaction terms $\boldsymbol{\tau}_j$ in (3.2) as a function of γ_j and the main effect parameters $\boldsymbol{\theta}_j$ and β_E. This reparametrization for both strong and weak heredity is summarized in Table 3.1.

Table 3.1: Reparametrization for strong and weak heredity principle for `sail` model

Type	Feature	Reparametrization
Strong heredity	$\hat{\boldsymbol{\tau}}_j \neq 0 \Rightarrow \hat{\boldsymbol{\theta}}_j \neq 0$ and $\hat{\beta}_E \neq 0$	$\boldsymbol{\tau}_j = \gamma_j \beta_E \boldsymbol{\theta}_j$
Weak heredity	$\hat{\boldsymbol{\tau}}_j \neq 0 \Rightarrow \hat{\boldsymbol{\theta}}_j \neq 0$ or $\hat{\beta}_E \neq 0$	$\boldsymbol{\tau}_j = \gamma_j (\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j)$

To perform variable selection in this new parametrization, we penalize $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ instead of penalizing $\boldsymbol{\tau}$ as in (3.5), leading to the following objective function:

$$\arg\min_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta}|\mathbf{D}) + \lambda(1-\alpha)\left(w_E|\beta_E| + \sum_{j=1}^{p} w_j \|\boldsymbol{\theta}_j\|_2\right) + \lambda\alpha \sum_{j=1}^{p} w_{jE}|\gamma_j| \tag{3.6}$$

This penalty allows for the possibility of excluding the interaction term from the model even if the corresponding main effects are non-zero. Furthermore, smaller values for α would lead to more interactions being included in the final model while values approaching 1 would

favor main effects. Similar to the elastic net (Zou & Hastie, 2005), we fix α and obtain a solution path over a sequence of λ values.

3.1.3 Toy example

We present here a toy example to better illustrate our method. With a sample size of $n = 100$, we sample $p = 20$ covariates $X_1, \dots X_p$ independently from a $N(0, 1)$ distribution truncated to the interval [0,1]. Data were generated from a model which follows the strong heredity principle, but where only one covariate, X_2, is involved in an interaction with a binary exposure variable, E:

$$Y = f_1(X_1) + f_2(X_2) + 1.75E + 1.5E \cdot f_2(X_2) + \varepsilon \tag{3.7}$$

For illustration, function $f_1(\cdot)$ is assumed to be linear, whereas function $f_2(\cdot)$ is non-linear: $f_1(x) = -3x$, $f_2(x) = 2(2x - 1)^3$. The error term ε is generated from a normal distribution with variance chosen such that the signal-to-noise ratio (SNR) is 2. We generated a single simulated dataset and used the strong heredity `sail` method with cubic B-splines to estimate the functional forms. 10-fold cross-validation (CV) was used to choose the optimal value of penalization. We used $\alpha = 0.5$ and default values were used for all other arguments. We plot the solution path for both main effects and interactions in Figure 3.1, coloring lines to correspond to the selected model. We see that our method is able to correctly identify the true model. We can also visually see the effect of the penalty and strong heredity principle working in tandem, i.e., the interaction term $E \cdot f_2(X_2)$ (orange lines in the bottom panel) can only be nonzero if the main effects E and $f_2(X_2)$ (black and orange lines respectively in the top panel) are nonzero, while nonzero main effects doesn't necessarily imply a nonzero interaction.

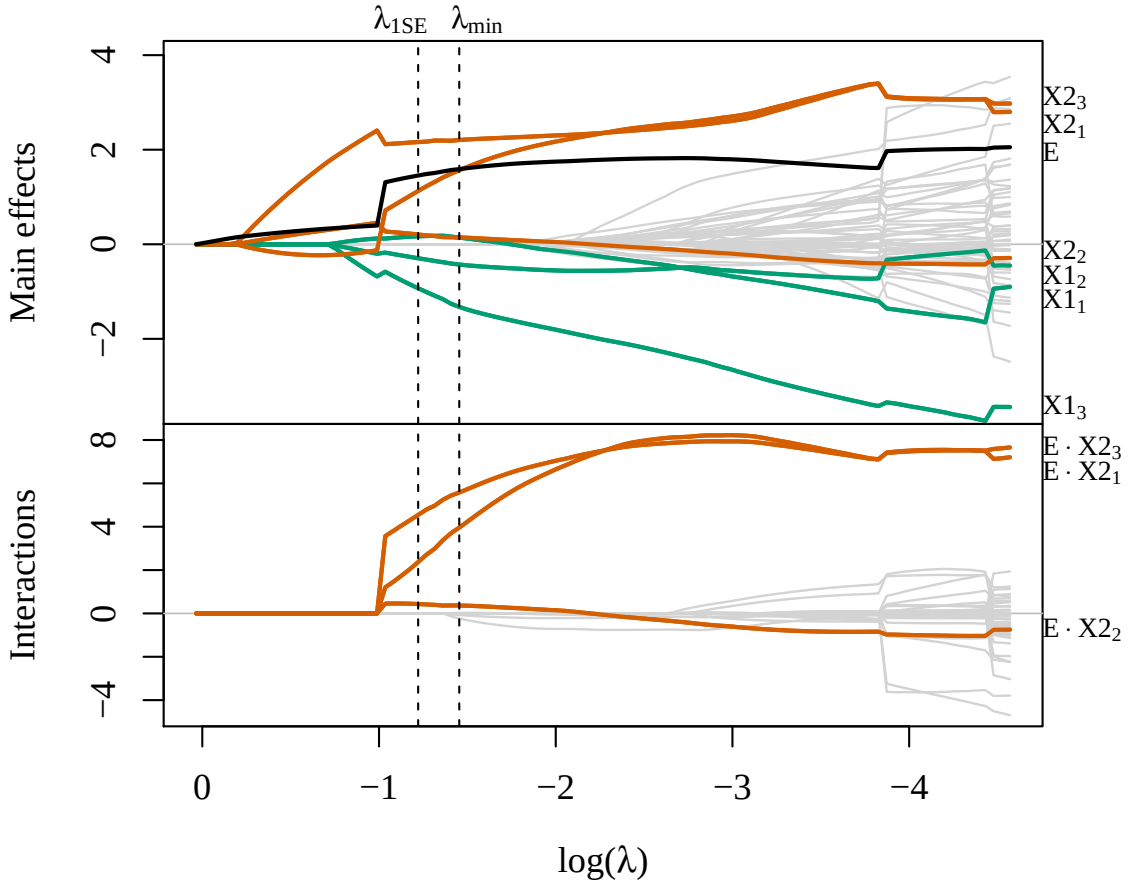


Figure 3.1: Toy example solution path for main effects (top) and interactions (bottom). $\{X1_1, X1_2, X1_3\}$ and $\{X2_1, X2_2, X2_3\}$ are the three basis coefficients for X_1 and X_2, respectively. λ_{1SE} is the largest value of penalization for which the CV error is within one standard error of the minimizing value λ_{min}.

In Figure 3.2, we plot the true and estimated component functions $\hat{f}_1(X_1)$ and $E \cdot \hat{f}_2(X_2)$, and their estimates from this analysis with `sail`. We are able to capture the shape of the correct functional form, but the means are not well aligned with the data. Lack-of-fit for $f_1(X_1)$ can be partially explained by acknowledging that `sail` is trying to fit a cubic spline to a linear function. Nevertheless, this example demonstrates that `sail` can still identify trends reasonably well.

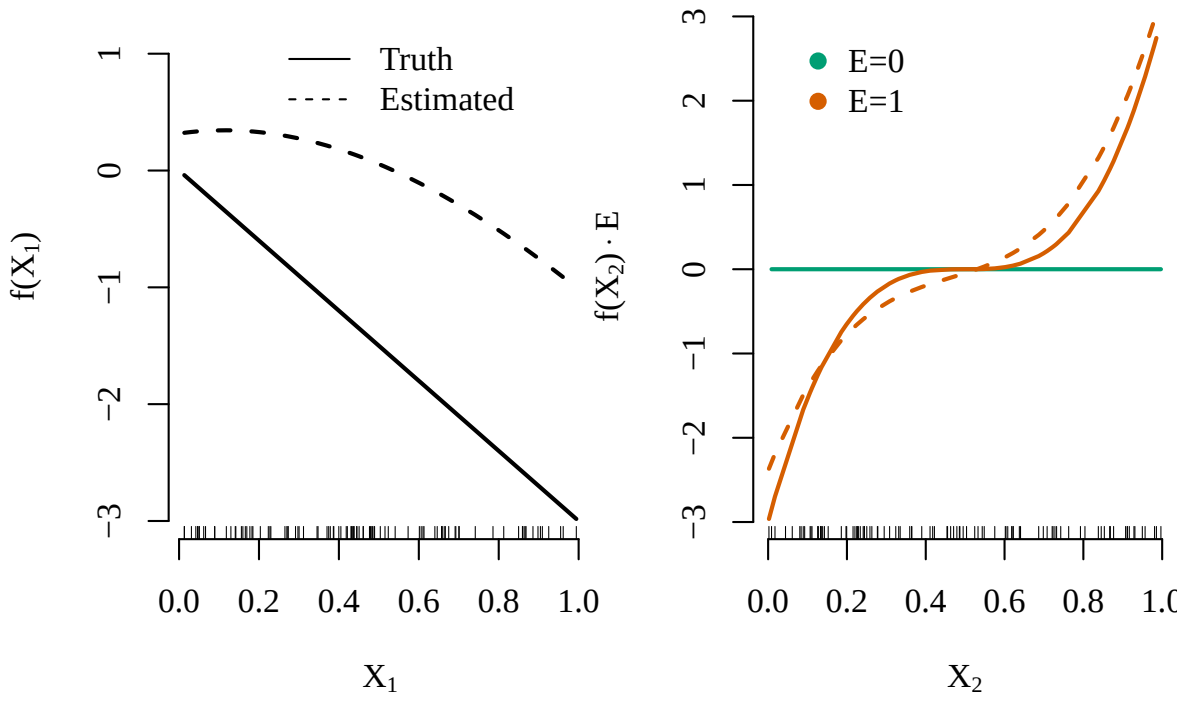


Figure 3.2: Estimated smooth functions for X_1 and the $X_2 \cdot E$ interaction by the `sail` method based on λ_{min}.

3.1.4 Related Work

Methods for variable selection of interactions can be broken down into two categories: linear and non-linear interaction effects. Many of the linear effect methods consider all pairwise interactions in $\mathbf{X}$ (Bien et al., 2013; Choi et al., 2010; She & Jiang, 2014; Zhao et al., 2009) which can be computationally prohibitive when p is large. The computational limitation can be perceived through the relatively small number of variables used in simulations and real data analysis in (Bien et al., 2013; Choi et al., 2010; She & Jiang, 2014; Zhao et al., 2009). More recent proposals for selection of interactions allow the user to restrict the search space to interaction candidates (Haris, Witten, & Simon, 2016; Lim & Hastie, 2015). This is useful when the researcher wants to impose prior information on the model. Two-stage procedures, where interaction candidates are considered from an original screen of main effects, have

shown good performance when p is large (Hao et al., 2018; Shah, 2016) in the linear setting. There are many fewer methods available for estimating non-linear interactions. For example, Radchenko and James (2010) (Radchenko & James, 2010) proposed a model of the form

$$Y = \beta_0 + \sum_{j=1}^{p} f_j(X_j) + \sum_{j>k} f_{jk}(X_j, X_k) + \varepsilon$$

where $f(\cdot)$ are smooth component functions. This method is more computationally expensive than `sail` since it considers all pairwise interactions between the basis functions, and its effectiveness in simulations or real-data applications is unknown as there is no software implementation.

While working on this paper, we were made aware of the recently proposed pliable lasso (Tibshirani & Friedman, 2017) which considers the interactions between $\mathbf{X}_{n\times p}$ and another matrix $\mathbf{Z}_{n\times K}$ and takes the form

$$Y = \beta_0 + \sum_{j=1}^{p} \beta_j X_j + \sum_{j=1}^{K} \theta_j Z_j + \sum_{j=1}^{p} (X_j \circ \mathbf{Z})\boldsymbol{\alpha}_j + \varepsilon \tag{3.8}$$

where α_j is a K-dimensional vector. Our proposal is most closely related to this method with $\mathbf{Z}$ being a single column matrix; the key difference being the non-linearity effects of our predictor variables. As pointed out by the authors of the pliable lasso, either their or ours can be seen as a varying coefficient model, i.e., the effect of X varies as a function of the exposure variable E or $\mathbf{Z}$ in equation 3.8.

The main contributions of this paper are fourfold. First, we develop a model for non-linear interactions with a key exposure variable, following either the weak or strong heredity principle, that is computationally efficient and scales to the high-dimensional setting ($n << p$). Second, through simulation studies, we show improved performance over existing methods that only consider linear interactions or additive main effects. Third, we show that our method possesses the oracle property (J. Fan & Li, 2001), i.e., it performs as well as if the true model

were known in advance. Fourth, all of our algorithms are implemented in the `sail` R package hosted on GitHub with extensive documentation (http://sahirbhatnagar.com/sail/). In particular, our implementation also allows for linear interaction models, user-defined basis expansions, a cross-validation procedure for selecting the optimal tuning parameter, and differential shrinkage parameters to apply the adaptive lasso (Zou, 2006) idea.

The rest of the paper is organized as follows. Section 3.2 describes our optimization procedure and some details about the algorithm used to fit the `sail` model for the least squares case. In Section 3.3, through simulation studies we compare the performance of our proposed approach and demonstrate the scenarios where it can be advantageous to use `sail` over existing methods. Section 3.4 contains some real data examples and Section 3.5 discusses some limitations and future directions.

3.2 Algorithm and Computational Details

In this section we describe a blockwise coordinate descent algorithm for fitting the least-squares version of the `sail` model in (3.6). We fix the value for α and minimize the objective function over a decreasing sequence of λ values ($\lambda_{max} > \cdots > \lambda_{min}$). We use the subgradient equations to determine the maximal value λ_{max} such that all estimates are zero. Due to the heredity principle, this reduces to finding the largest λ such that all main effects $(\beta_E, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p)$ are zero. Following Friedman et al. (J. Friedman et al., 2010), we construct a λ-sequence of 100 values decreasing from λ_{max} to $0.001\lambda_{max}$ on the log scale, and use the warm start strategy where the solution for λ_ℓ is used as a starting value for $\lambda_{\ell+1}$.

3.2.1 Blockwise coordinate descent for least-squares loss

The strong heredity `sail` model with least-squares loss has the form

$$\hat{Y} = \beta_0 \cdot \mathbf{1} + \sum_{j=1}^{p} \mathbf{\Psi}_j \boldsymbol{\theta}_j + \beta_E X_E + \sum_{j=1}^{p} \gamma_j \beta_E (X_E \circ \mathbf{\Psi}_j) \boldsymbol{\theta}_j \tag{3.9}$$

and the objective function is given by

$$Q(\mathbf{\Theta}) = \frac{1}{2n} \left\| Y - \hat{Y} \right\|_2^2 + \lambda(1-\alpha) \left(w_E |\beta_E| + \sum_{j=1}^{p} w_j \|\boldsymbol{\theta}_j\|_2 \right) + \lambda\alpha \sum_{j=1}^{p} w_{jE} |\gamma_j| \tag{3.10}$$

Solving (3.10) in a blockwise manner allows us to leverage computationally fast algorithms for ℓ_1 and ℓ_2 norm penalized regression. We show in Supplemental Section A.1 that by careful construction of pseudo responses and pseudo design matrices, existing efficient algorithms can be used to estimate the parameters. Indeed, the objective function simplifies to a modified lasso problem when holding all θ_j fixed, and a modified group lasso problem when holding β_E and all γ_j fixed.

Denote the n-dimensional residual column vector $R = Y - \hat{Y}$. The subgradient equations are given by

$$\frac{\partial Q}{\partial \beta_0} = \frac{1}{n} \left(Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \mathbf{\Psi}_j \boldsymbol{\theta}_j - \beta_E X_E - \sum_{j=1}^{p} \gamma_j \beta_E (X_E \circ \mathbf{\Psi}_j) \boldsymbol{\theta}_j \right)^\top \mathbf{1} = 0 \tag{3.11}$$

$$\frac{\partial Q}{\partial \beta_E} = -\frac{1}{n} \left(X_E + \sum_{j=1}^{p} \gamma_j (X_E \circ \mathbf{\Psi}_j) \boldsymbol{\theta}_j \right)^\top R + \lambda(1-\alpha) w_E s_1 = 0 \tag{3.12}$$

$$\frac{\partial Q}{\partial \boldsymbol{\theta}_j} = -\frac{1}{n} \left(\mathbf{\Psi}_j + \gamma_j \beta_E (X_E \circ \mathbf{\Psi}_j) \right)^\top R + \lambda(1-\alpha) w_j s_2 = \mathbf{0} \tag{3.13}$$

$$\frac{\partial Q}{\partial \gamma_j} = -\frac{1}{n} \left(\beta_E (X_E \circ \mathbf{\Psi}_j) \boldsymbol{\theta}_j \right)^\top R + \lambda\alpha w_{jE} s_3 = 0 \tag{3.14}$$

where s_1 is in the subgradient of the ℓ_1 norm:

$$s_1 \in \begin{cases} \text{sign}\,(\beta_E) & \text{if } \beta_E \neq 0 \\ [-1, 1] & \text{if } \beta_E = 0, \end{cases}$$

s_2 is in the subgradient of the ℓ_2 norm:

$$s_2 \in \begin{cases} \dfrac{\boldsymbol{\theta}_j}{\|\boldsymbol{\theta}_j\|_2} & \text{if } \boldsymbol{\theta}_j \neq \mathbf{0} \\ u \in \mathbb{R}^{m_j} : \|u\|_2 \leq 1 & \text{if } \boldsymbol{\theta}_j = \mathbf{0}, \end{cases}$$

and s_3 is in the subgradient of the ℓ_1 norm:

$$s_3 \in \begin{cases} \text{sign}\,(\gamma_j) & \text{if } \gamma_j \neq 0 \\ [-1, 1] & \text{if } \gamma_j = 0. \end{cases}$$

Define the partial residuals, without the jth predictor for $j = 1, \ldots, p$, as

$$R_{(-j)} = Y - \beta_0 \cdot \mathbf{1} - \sum_{\ell \neq j} \boldsymbol{\Psi}_\ell \boldsymbol{\theta}_\ell - \beta_E X_E - \sum_{\ell \neq j} \gamma_\ell \beta_E (X_E \circ \boldsymbol{\Psi}_\ell) \boldsymbol{\theta}_\ell$$

the partial residual without X_E as

$$R_{(-E)} = Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \boldsymbol{\theta}_j$$

and the partial residual without the jth interaction for $j = 1, \ldots, p$, as

$$R_{(-jE)} = Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \boldsymbol{\theta}_j - \beta_E X_E - \sum_{\ell \neq j} \gamma_\ell \beta_E (X_E \circ \boldsymbol{\Psi}_\ell) \boldsymbol{\theta}_\ell$$

From the subgradient equations (3.11)–(3.14) we see that

$$\hat{\beta}_0 = \left(Y - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \hat{\boldsymbol{\theta}}_j - \hat{\beta}_E X_E - \sum_{j=1}^{p} \hat{\gamma}_j \hat{\beta}_E (X_E \circ \boldsymbol{\Psi}_j) \hat{\boldsymbol{\theta}}_j \right)^\top \mathbf{1} \tag{3.15}$$

$$\hat{\beta}_E = S\left(\frac{1}{n \cdot w_E} \left(X_E + \sum_{j=1}^{p} \hat{\gamma}_j (X_E \circ \boldsymbol{\Psi}_j) \hat{\boldsymbol{\theta}}_j \right)^\top R_{(-E)}, \lambda(1-\alpha) \right) \tag{3.16}$$

$$\lambda(1-\alpha) w_j \frac{\boldsymbol{\theta}_j}{\|\boldsymbol{\theta}_j\|_2} = \frac{1}{n} \left(\boldsymbol{\Psi}_j + \gamma_j \beta_E (X_E \circ \boldsymbol{\Psi}_j) \right)^\top R_{(-j)} \tag{3.17}$$

$$\hat{\gamma}_j = S\left(\frac{1}{n \cdot w_{jE}} \left(\beta_E (X_E \circ \boldsymbol{\Psi}_j) \boldsymbol{\theta}_j \right)^\top R_{(-jE)}, \lambda\alpha \right) \tag{3.18}$$

where $S(x,t) = \text{sign}(x)(|x| - t)$ is the soft-thresholding operator. We see from (3.15) and (3.16) that there are closed form solutions for the intercept and β_E. From (3.18), each γ_j also has a closed form solution and can be solved efficiently for $j = 1, \ldots, p$ using a coordinate descent procedure (J. Friedman et al., 2010). Since there is no closed form solution for β_j, we use a quadratic majorization technique (Y. Yang & Zou, 2015) to solve (3.17). Furthermore, we update each $\boldsymbol{\theta}_j$ in a coordinate wise fashion and leverage this to implement further computational speedups which are detailed in Supplemental Section A.1.2. From these estimates, we compute the interaction effects using the reparametrizations presented in Table 3.1, e.g., $\hat{\boldsymbol{\tau}}_j = \hat{\gamma}_j \hat{\beta}_E \hat{\boldsymbol{\theta}}_j$, $j = 1, \ldots, p$ for the strong heredity `sail` model. We provide an overview of the computations in Algorithm 4. A more detailed version of this algorithm is given in Section A.1.1 of the Appendix.

Algorithm 4 Blockwise Coordinate Descent for Least-Squares `sail` with Strong Heredity.

For a decreasing sequence $\lambda = \lambda_{max}, \ldots, \lambda_{min}$ and fixed α:

1. Initialize $\beta_0^{(0)}, \beta_E^{(0)}, \boldsymbol{\theta}_j^{(0)}, \gamma_j^{(0)}$ for $j = 1, \ldots, p$ and set iteration counter $k \leftarrow 0$.
2. Repeat the following until convergence:
   (a) update $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$
      i. Compute the pseudo design: $\widetilde{X}_j \leftarrow \beta_E^{(k)}(X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\theta}_j^{(k)} \qquad$ for $j = 1, \ldots, p$
      ii. Compute the pseudo response $\widetilde{Y}$ by removing the contribution of every term not involving $\boldsymbol{\gamma}$ from Y
      iii. Solve:

$$\boldsymbol{\gamma}^{(k)(new)} \leftarrow \arg\min_{\boldsymbol{\gamma}} \frac{1}{2n}\left\|\widetilde{Y} - \sum_j \gamma_j \widetilde{X}_j\right\|_2^2 + \lambda\alpha \sum_j w_{jE}|\gamma_j| \tag{3.19}$$

      iv. Set $\boldsymbol{\gamma}^{(k)} = \boldsymbol{\gamma}^{(k)(new)}$
   (b) update $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p)$
      — for $j = 1, \ldots, p$
      i. Compute the pseudo design: $\widetilde{X}_j \leftarrow \boldsymbol{\Psi}_j + \gamma_j^{(k)}\beta_E^{(k)}(X_E \circ \boldsymbol{\Psi}_j)$
      ii. Compute the pseudo response $(\widetilde{Y})$ by removing the contribution of every term not involving $\boldsymbol{\theta}_j$ from Y
      iii. Solve:

$$\boldsymbol{\theta}_j^{(k)(new)} \leftarrow \arg\min_{\boldsymbol{\theta}_j} \frac{1}{2n}\left\|\widetilde{Y} - \widetilde{X}_j\boldsymbol{\theta}_j\right\|_2^2 + \lambda(1-\alpha)w_j\|\theta_j\|_2 \tag{3.20}$$

      iv. Set $\boldsymbol{\theta}_j^{(k)} \leftarrow \boldsymbol{\theta}_j^{(k)(new)}$
   (c) update β_E
      i. Compute the pseudo design: $\widetilde{X}_E \leftarrow X_E + \sum_j \gamma_j^{(k)}\widetilde{\boldsymbol{\Psi}}_j\boldsymbol{\theta}_j^{(k)}$
      ii. Compute the pseudo response $(\widetilde{Y})$ by removing the contribution of every term not involving β_E from Y
      iii. Soft-threshold update ($S(x,t) = \text{sign}(x)(|x| - t)_+$):

$$\beta_E^{(k)(new)} \leftarrow S\left(\frac{1}{n \cdot w_E}\widetilde{X}_E^\top \widetilde{Y}, \lambda(1-\alpha)\right) \tag{3.21}$$

      iv. Set $\beta_E^{(k+1)} \leftarrow \beta_E^{(k)(new)}$, $k \leftarrow k+1$

3.2.2 Maximum penalty parameter (Lambda max)

The subgradient equations (3.12)–(3.14) can be used to determine the largest value of λ such that all coefficients are 0. From the subgradient Equation (3.12), we see that $\beta_E = 0$ is a solution if

$$\frac{1}{w_E}\left|\frac{1}{n}\left(X_E + \sum_{j=1}^{p}\gamma_j(X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\theta}_j\right)^\top R_{(-E)}\right| \leq \lambda(1-\alpha) \tag{3.22}$$

From the subgradient Equation (3.13), we see that $\boldsymbol{\theta}_j = \mathbf{0}$ is a solution if

$$\frac{1}{w_j}\left\|\frac{1}{n}\left(\boldsymbol{\Psi}_j + \gamma_j\beta_E(X_E \circ \boldsymbol{\Psi}_j)\right)^\top R_{(-j)}\right\|_2 \leq \lambda(1-\alpha) \tag{3.23}$$

From the subgradient Equation (3.14), we see that $\gamma_j = 0$ is a solution if

$$\frac{1}{w_{jE}}\left|\frac{1}{n}\left(\beta_E(X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\theta}_j\right)^\top R_{(-jE)}\right| \leq \lambda\alpha \tag{3.24}$$

Due to the strong heredity property, the parameter vector $(\beta_E, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p, \gamma_1, \ldots, \gamma_p)$ will be entirely equal to $\mathbf{0}$ if $(\beta_E, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p) = \mathbf{0}$. Therefore, the smallest value of λ for which the entire parameter vector (excluding the intercept) is $\mathbf{0}$ is:

$$\lambda_{max} = \frac{1}{n(1-\alpha)}\max\left\{\frac{1}{w_E}\left(X_E + \sum_{j=1}^{p}\gamma_j(X_E \circ \boldsymbol{\Psi}_j)\boldsymbol{\theta}_j\right)^\top R_{(-E)}, \max_j \frac{1}{w_j}\left\|\left(\boldsymbol{\Psi}_j + \gamma_j\beta_E(X_E \circ \boldsymbol{\Psi}_j)\right)^\top R_{(-j)}\right\|_2\right\} \tag{3.25}$$

which reduces to

$$\lambda_{max} = \frac{1}{n(1-\alpha)}\max\left\{\frac{1}{w_E}\left(X_E\right)^\top R_{(-E)}, \max_j \frac{1}{w_j}\left\|\left(\boldsymbol{\Psi}_j\right)^\top R_{(-j)}\right\|_2\right\}$$

3.2.3 Weak Heredity

Our method can be easily adapted to enforce the weak heredity property:

$$\hat{\alpha}_{jE} \neq 0 \quad \Rightarrow \quad \hat{\beta}_j \neq 0 \quad \text{or} \quad \hat{\beta}_E \neq 0$$

That is, an interaction term can only be present if at least one of it's corresponding main effects is nonzero. To do so, we reparametrize the coefficients for the interaction terms in (3.2) as $\boldsymbol{\alpha}_j = \gamma_j(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j)$, where $\mathbf{1}_{m_j}$ is a vector of ones with dimension m_j (i.e. the length of $\boldsymbol{\theta}_j$). We defer the algorithm details for fitting the `sail` model with weak heredity in Section A.1.3 of the Appendix, as it is very similar to Algorithm 4 for the strong heredity `sail` model.

3.2.4 Adaptive `sail`

The weights for the environment variable, main effects and interactions are given by w_E, w_j and w_{jE} respectively. These weights serve as a means of allowing a different penalty to be applied to each variable. In particular, any variable with a weight of zero is not penalized at all. This feature is usually selected for one of two reasons:

1. Prior knowledge about the importance of certain variables is known. Larger weights will penalize the variable more, while smaller weights will penalize the variable less
2. Allows users to apply the adaptive `sail`, similar to the adaptive lasso (Zou, 2006)

We describe the adaptive `sail` in Algorithm 5. This is a general procedure that can be applied to the weak and strong heredity settings, as well as both least squares and logistic loss functions. We provide this capability in the `sail` package using the `penalty.factor` argument and provide an example in Section A.3.6 of the Appendix.

Algorithm 5 Adaptive `sail` algorithm

1. For a decreasing sequence $\lambda = \lambda_{max}, \ldots, \lambda_{min}$ and fixed α run the `sail` algorithm
2. Use cross-validation or a data splitting procedure to determine the optimal value for the tuning parameter: $\lambda^{[opt]} \in \{\lambda_{max}, \ldots, \lambda_{min}\}$
3. Let $\widehat{\beta_E}^{[opt]}, \widehat{\boldsymbol{\theta}}_j^{[opt]}$ and $\widehat{\boldsymbol{\tau}}_j^{[opt]}$ for $j = 1, \ldots, p$ be the coefficient estimates corresponding to the model at $\lambda^{[opt]}$
4. Set the weights to be
$$w_E = \left(\left|\widehat{\beta_E}^{[opt]}\right|\right)^{-1}, \; w_j = \left(\|\widehat{\boldsymbol{\theta}}_j^{[opt]}\|_2\right)^{-1}, \; w_{jE} = \left(\|\widehat{\boldsymbol{\tau}}_j^{[opt]}\|_2\right)^{-1} \text{ for } j = 1, \ldots, p$$
5. Run the `sail` algorithm with the weights defined in step 4), and use cross-validation or a data splitting procedure to choose the optimal value of λ

3.2.5 Flexible design matrix

The definition of the basis expansion functions in (3.1) is very flexible, in the sense that our algorithms are independent of this choice. As a result, the user can apply any basis expansion they desire. In the extreme case, one could apply the identity map, i.e., $f_j(X_j) = X_j$ which leads to a linear interaction model (referred to as `linear sail`). When little information is known a priori about the relationship between the predictors and the response, by default, we choose to apply the same basis expansion to all columns of $\mathbf{X}$. This is a reasonable approach when all the variables are continuous. However, there are often situations when the data contains a combination of categorical and continuous variables. In these cases it may be sub-optimal to apply a basis expansion to the categorical variables. Owing to the flexible nature of our algorithm, we can handle this scenario in our implementation by allowing a user-defined design matrix. The only extra information needed is the group membership of each column in the design matrix. We provide such an example in the `sail` package showcase in Section A.3.7 of the Appendix.

3.3 Simulation Study

In this section, we use simulated data to understand the performance of `sail` in different scenarios.

3.3.1 Comparator Methods

Since there are no other packages that directly address our chosen problem, we selected comparator methods based on the following criteria: 1) penalized regression methods that can handle high-dimensional data ($n < p$), 2) allows at least one of linear effects, non-linear effects or interaction effects, and 3) has a software implementation in `R`. The selected methods can be grouped into three categories:

1. Linear main effects: `lasso` (Tibshirani, 1996), `adaptive lasso` (Zou, 2006)
2. Linear interactions: `lassoBT` (Shah, 2016), `GLinternet` (Lim & Hastie, 2015)
3. Non-linear main effects: `HierBasis` (Haris, Shojaie, & Simon, 2016), `SPAM` (Ravikumar et al., 2009), `gamsel` (Chouldechova & Hastie, 2015)

For `GLinternet` we specified the `interactionCandidates` argument so as to only consider interactions between the environment and all other X variables. For all other methods we supplied $(\mathbf{X}, X_E)$ as the data matrix, 100 for the number of tuning parameters to fit, and used the default values otherwise[1]. `lassoBT` considers all pairwise interactions as there is no way for the user to restrict the search space. `SPAM` applies the same basis expansion to every column of the data matrix; we chose 5 basis spline functions. `HierBasis` and `gamsel` selects whether a term in an additive model is nonzero, linear, or a non-linear spline up to a specified max degrees of freedom per variable.

We compare the above listed methods with our main proposal method `sail`, as well as

[1] `R` code for each method available at https://github.com/sahirbhatnagar/sail/blob/master/my_sims/method_functions.R

with `adaptive sail` (Algorithm 5), `sail weak` which has the weak heredity property and `linear sail` as described in Section 3.2.5. For each function f_j, we use a B-spline basis matrix with `degree=5` implemented in the `bs` function in `R` (R Core Team, 2016). We center the environment variable and the basis functions before running the `sail` method.

3.3.2 Simulation Design

To make the comparisons with other methods as fair as possible, we followed a simulation framework that has been previously used for variable selection methods in additive models (Huang et al., 2010; Y. Lin et al., 2006). We extend this framework to include interaction effects as well. The covariates are simulated as follows. First, we generate $z_1, \ldots, z_p, u, v$ independently from a standard normal distribution truncated to the interval [0,1] for $i = 1, \ldots, n$. Then we set $x_j = (z_j + t \cdot u)/(1+t)$ for $j = 1, \ldots, 4$ and $x_j = (z_j + t \cdot v)/(1+t)$ for $j = 5, \ldots, p$, where the parameter t controls the amount of correlation among predictors. The first four variables are nonzero (i.e. active in the response), while the rest of the variables are zero (i.e. are noise variables). This leads to a compound symmetry correlation structure where $Corr(x_j, x_k) = t^2/(1+t^2)$, for $1 \leq j \leq 4, 1 \leq k \leq 4$, and $Corr(x_j, x_k) = t^2/(1+t^2)$, for $5 \leq j \leq p, 5 \leq k \leq p$, but the covariates of the nonzero and zero components are independent. We consider the case when $p = 1000$ and $t = 0$. The outcome Y is then generated following one of the models and assumptions described below.

We evaluate the performance of our method on three of its defining characteristics: 1) the strong heredity property, 2) non-linearity of predictor effects and 3) interactions. Simulation scenarios are designed specifically to test the performance of these characteristics

1. **Hierarchy simulation**

   Scenario (a) Truth obeys strong hierarchy. In this situation, the true model for

Y contains main effect terms for all covariates involved in interactions.

$$Y = \sum_{j=1}^{4} f_j(X_j) + \beta_E \cdot X_E + X_E \cdot f_3(X_3) + X_E \cdot f_4(X_4) + \varepsilon$$

Scenario (b) Truth obeys weak hierarchy. Here, in addition to the interaction, the E variable has its own main effect but the covariates X_3 and X_4 do not.

$$Y = f_1(X_1) + f_2(X_2) + \beta_E \cdot X_E + X_E \cdot f_3(X_3) + X_E \cdot f_4(X_4) + \varepsilon$$

Scenario (c) Truth only has interactions. In this simulation, the covariates involved in interactions do not have main effects as well.

$$Y = X_E \cdot f_3(X_3) + X_E \cdot f_4(X_4) + \varepsilon$$

2. **Non-linearity simulation scenario**

Truth is linear. `sail` is designed to model non-linearity; here we assess its performance if the true model is completely linear.

$$Y = 5X_1 + 3(X_2 + 1) + 4X_3 + 6(X_4 - 2) + \beta_E \cdot X_E + X_E \cdot 4X_3 + X_E \cdot 6(X_4 - 2) + \varepsilon$$

3. **Interactions simulation scenario**

Truth only has main effects. `sail` is designed to capture interactions; here we assess its performance when there are none in the true model.

$$Y = \sum_{j=1}^{4} f_j(X_j) + \beta_E \cdot X_E + \varepsilon$$

The true component functions are the same as in (Huang et al., 2010; Y. Lin et al., 2006) and are given by $f_1(t) = 5t$, $f_2(t) = 3(2t-1)^2$, $f_3(t) = 4\sin(2\pi t)/(2-\sin(2\pi t))$, $f_4(t) = 6(0.1\sin(2\pi t) + 0.2\cos(2\pi t) + 0.3\sin(2\pi t)^2 + 0.4\cos(2\pi t)^3 + 0.5\sin(2\pi t)^3)$. We set $\beta_E = 2$ and draw ε from a normal distribution with variance chosen such that the signal-to-noise ratio is 2. Using this setup, we generated 200 replications consisting of a training set of $n = 200$, a validation set of $n = 200$ and a test set of $n = 800$. The training set was used to fit the model and the validation set was used to select the optimal tuning parameter corresponding to the minimum prediction mean squared error (MSE). Variable selection results including true positive rate, false positive rate and number of active variables (the number of variables with a non-zero coefficient estimate) were assessed on the training set, and MSE was assessed on the test set.

3.3.3 Results

The test set MSE results for each of the five simulation scenarios are shown in Figure 3.3, while Figure 3.4 shows the mean true positive rate (TPR) vs. the mean false positive rate (FPR) ± 1 standard deviation (SD).

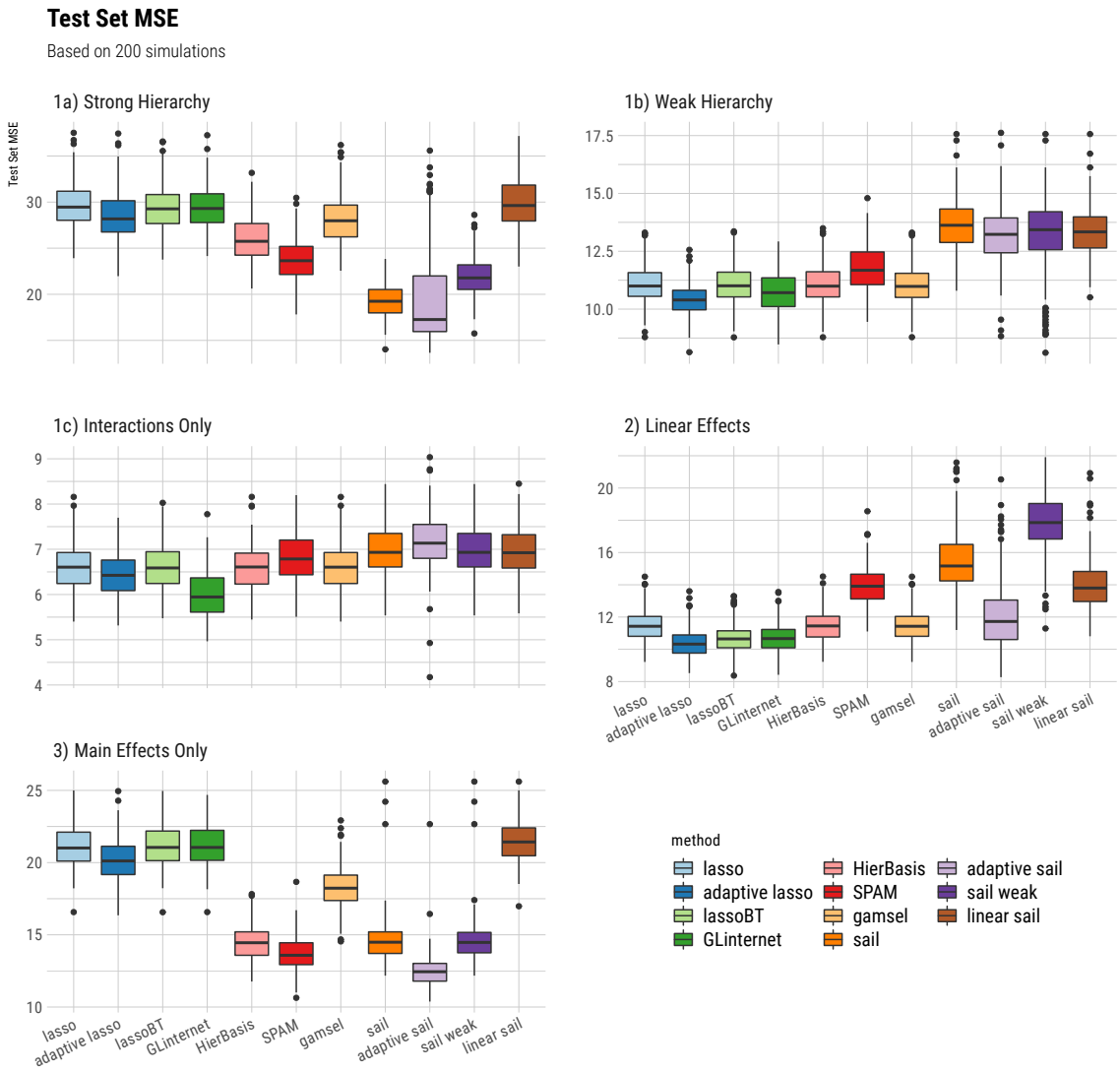


Figure 3.3: Boxplots of the test set mean squared error from 200 simulations for each of the five simulation scenarios.

We see that `sail` ,`adaptive sail` and `sail weak` have the best performance in terms of both MSE and yielding correct sparse models when the truth follows a strong hierarchy (scenario 1a), as we would expect, since this is exactly the scenario that our method is trying to target.

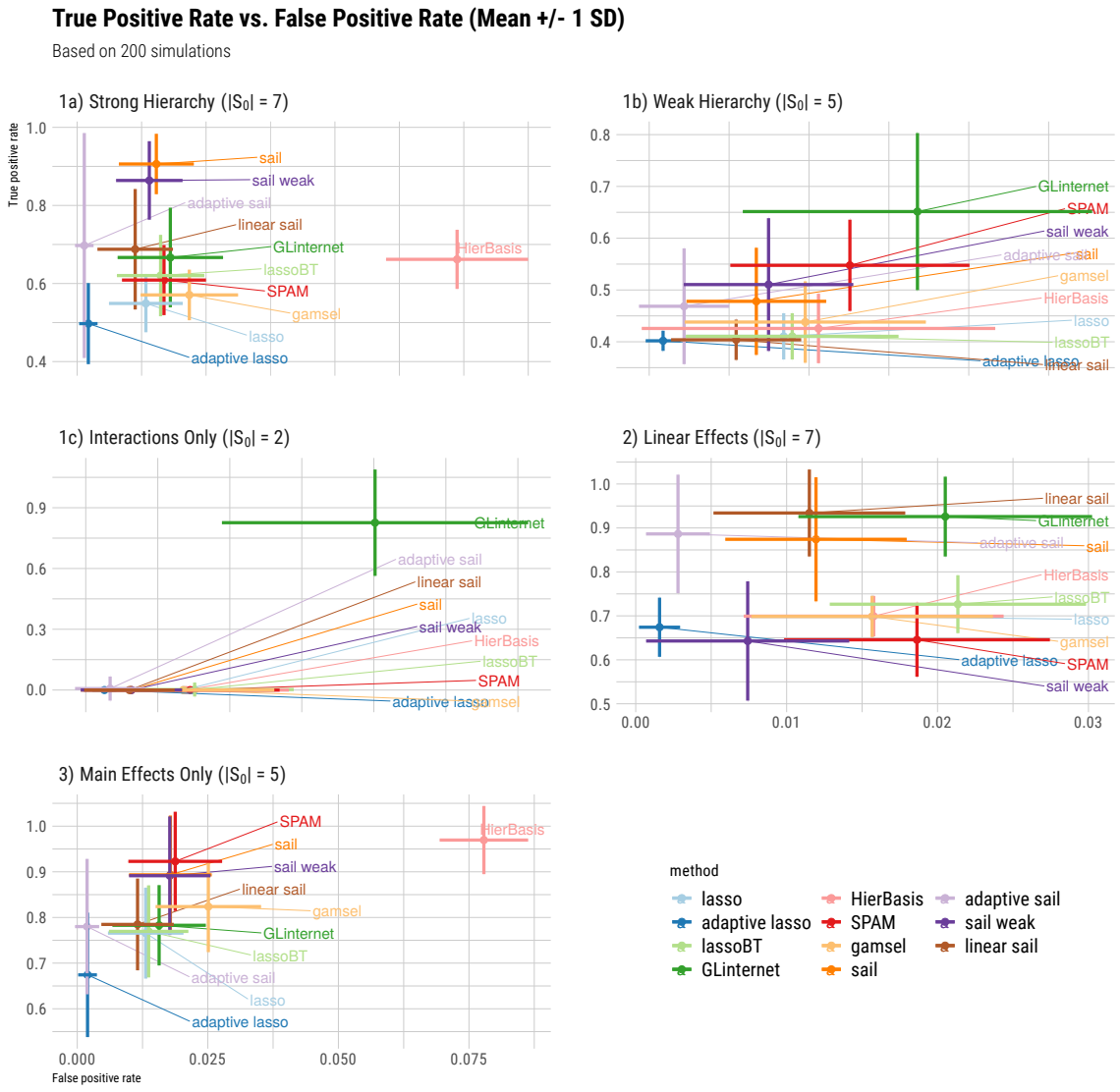


Figure 3.4: Means ± 1 standard deviation of true positive rate vs. false positive rate from 200 simulations for each of the five scenarios. $|S_0|$ is the number of truly associated variables.

Our method is also competitive when only main effects are present (scenario 3) and performs just as well as methods that only consider linear and non-linear main effects (`HierBasis`, `SPAM`), owing to the penalization applied to the interaction parameter. Due to the heredity property, our method is unable to capture any of the truly associated variables when only interactions are present (scenario 1c). However, the other methods also fail to capture any

signal, with the exception of `GLinternet` which has a high TPR and FPR. When only linear effects and interactions are present (scenario 2), we see that `linear sail` has a high TPR and low FPR as compared to the other linear interaction methods (`lassoBT` and `GLinternet`) though the test set MSE is not as good. The `lasso` and `adaptive lasso` have good test set MSE performance but poor sensitivity. Additional results are available in Section A.2 of the Appendix. Specifically, in Figure A.1 we plot the mean MSE against the mean number of active variables ± 1 standard deviation (SD). Figures A.2 and A.3 show the true positive and false positive rates, respectively. Figure A.4 shows the number of active variables.

We visually inspected whether our method could correctly capture the shape of the association between the predictors and the response for both main and interaction effects. To do so, we plotted the true and predicted curves for scenario 1a) only. Figure 3.5 shows each of the four main effects with the estimated curves from each of the 200 simulations along with the true curve. We can see the effect of the penalty on the parameters, i.e., decreasing prediction variance at the cost of increased bias. This is particularly well illustrated in the bottom right panel where `sail` smooths out the very wiggly component function $f_4(x)$. Nevertheless, the primary shapes are clearly being captured.

To visualize the estimated interaction effects, we ordered the 200 simulation runs by the euclidean distance between the estimated and true regression functions. Following Radchenko et al. (Radchenko & James, 2010), we then identified the 25th, 50th, and 75th best simulations and plotted, in Figures 3.6 and 3.7, the interaction effects of X_E with $f_3(X_3)$ and $f_4(X_4)$, respectively. We see that `sail` does a good job at capturing the true interaction surface for $X_E \cdot f_3(X_3)$. Again, the smoothing and shrinkage effect is apparent when looking at the interaction surfaces for $X_E \cdot f_4(X_4)$

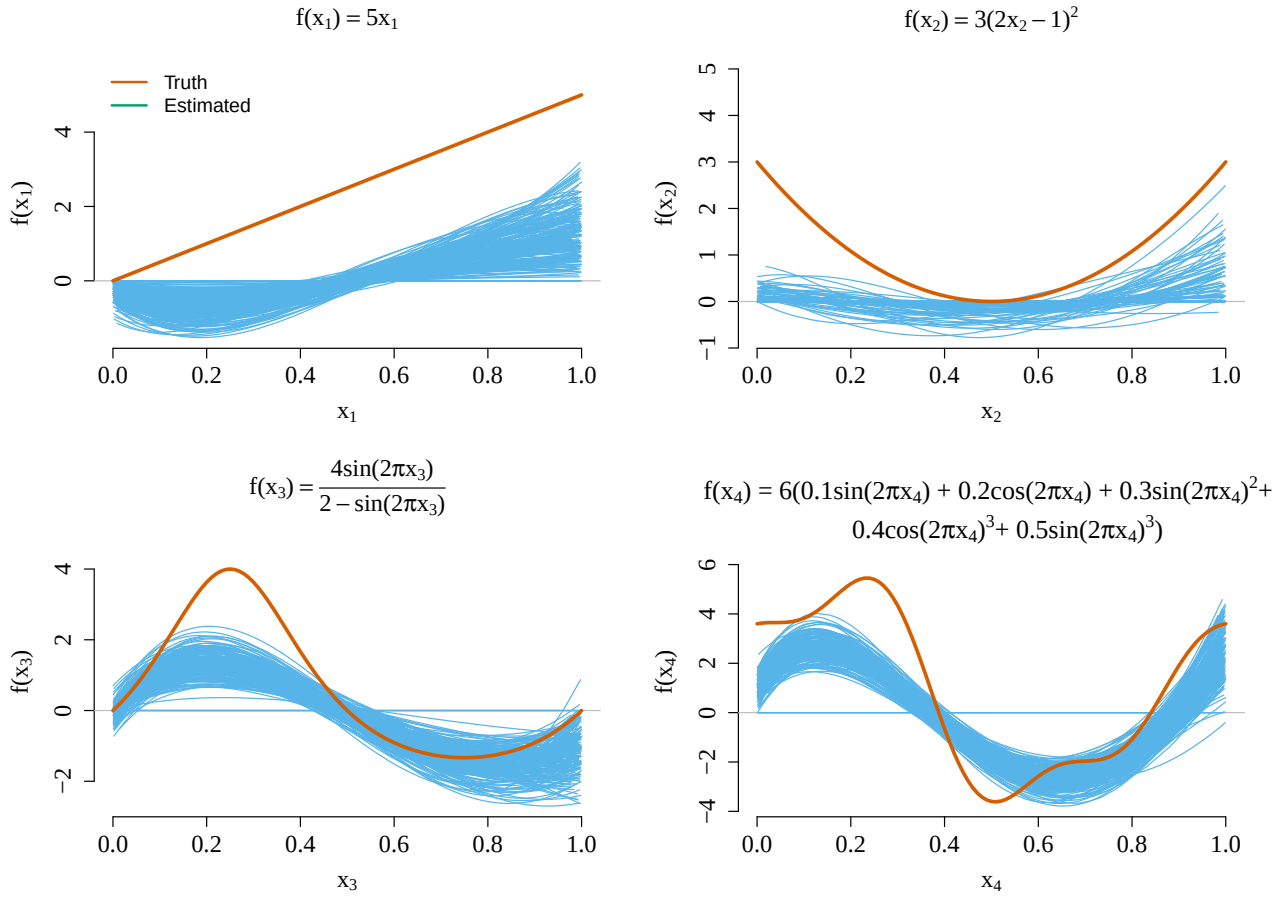


Figure 3.5: True and estimated main effect component functions for scenario 1a). The estimated curves represent the results from each one of the 200 simulations conducted.

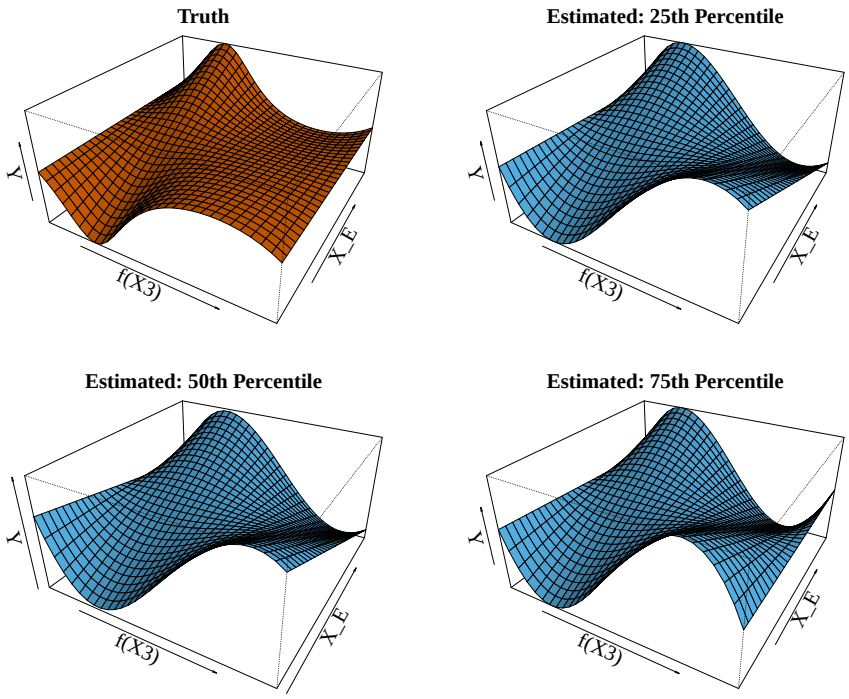


Figure 3.6: True and estimated interaction effects for $X_E \cdot f_3(X_3)$ in simulation scenario 1a).

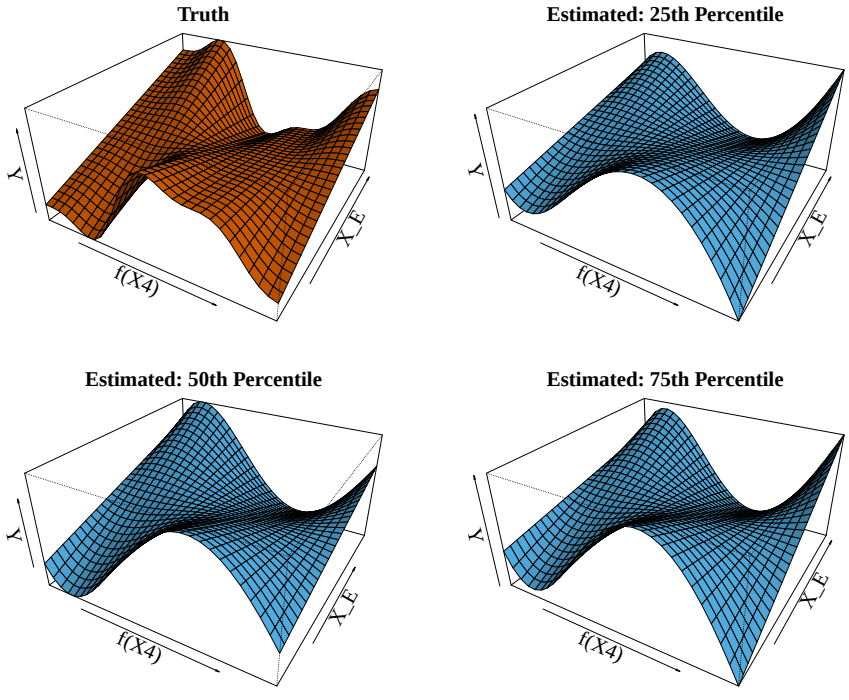


Figure 3.7: True and estimated interaction effects for $X_E \cdot f_4(X_4)$ in simulation scenario 1a).

3.4 Real Data Application

In this section we illustrate `sail` on several real data examples.

3.4.1 Alzheimer's Disease Neuroimaging Initiative

Alzheimer's is an irreversible neurodegenerative disease that results in a loss of mental function due to the deterioration of brain tissue. The overall goal of the Alzheimer's Disease Neuroimaging Initiative (ADNI) is to validate biomarkers for use in Alzheimer's disease clinical treatment trials (Petersen et al., 2010). The patients were selected into the study based on their clinical diagnosis: controls, mild cognitive impairment (MCI) or Alzheimer's disease (AD). PET amyloid imaging was used to asses amyloid beta (Aβ) protein load in 96 brain regions. The response we use here is general cognitive decline, as measured by a continuous mini-mental state examination score. We applied `sail` to this data to see if there were any non-linear interactions between clinical diagnosis and Aβ protein in the 96 brain regions on mini-mental state examination.

There were a total of 343 patients who we divided randomly into equal sized training/validation/test splits. We ran the strong heredity `sail` with cubic B-splines and $\alpha = 0.1$. We also applied the `lasso, lassoBT, HierBasis` and `GLinternet` to this data. Using the same default settings and strategy as the simulation study, we ran each method on the training data, determined the optimal tuning parameter on the validation data, and assessed MSE on the test data. We repeated this process 200 times.

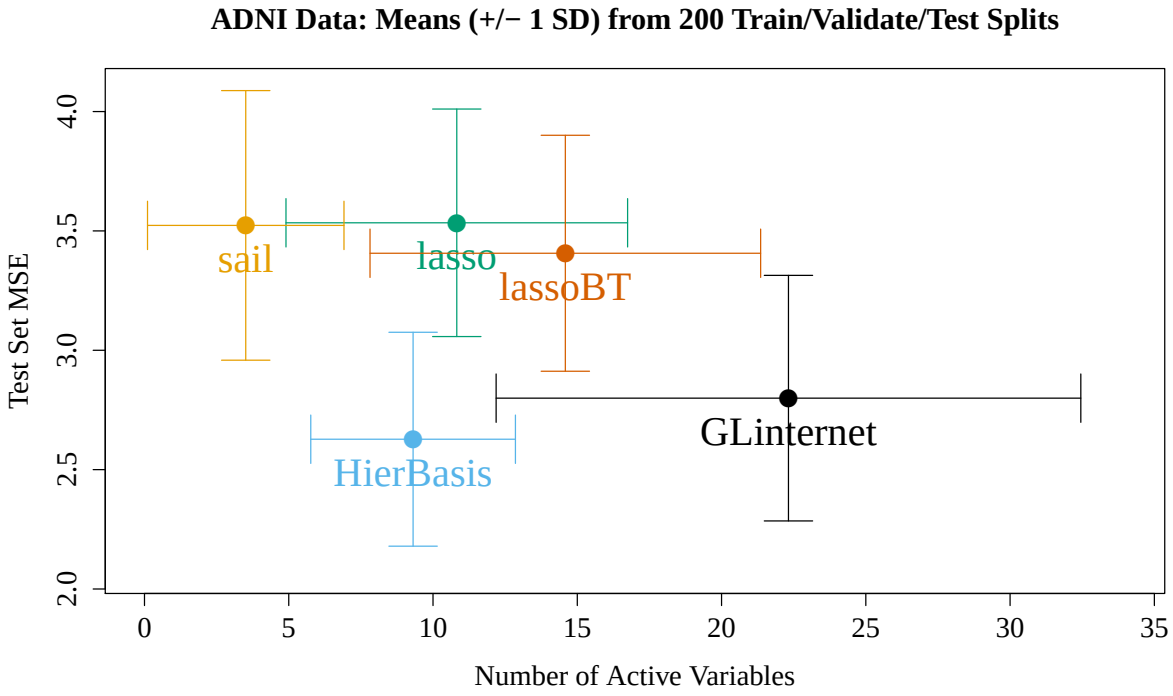


Figure 3.8: Mean test set MSE vs. mean number of active variables (± 1 SD) for ADNI data based on 200 train/validation/test splits.

In Figure 3.8 we plot the mean test set MSE vs. the mean number of active variables ± 1 SD. We see that `sail` produces the sparsest models but doesn't perform as well as `HierBasis` and `GLinternet` in terms of MSE. `sail` achieves a similar MSE to both the `lasso` and `lassoBT` with fewer variables on average. `GLinternet` produces the largest models and seems to be sensitive to the train/validate/test split as evidenced by the large standard deviations.

To visualize the results from the `sail` method, we chose the train/validate/test split which led to the best test set MSE, and then plotted the interaction effects in Figure 3.9. The left panel shows the middle occipital gyrus left region in the occipital lobe known for visual object perception. We see that more Aβ protein loads leads to a worse cognitive score for the MCI and AD group but not for the controls. The right panel shows the cuneus region which is known to be involved in basic visual processing, and we see that more Aβ proteins leads to better cognitive scores for the MCI and AD group and poorer scores for the controls.

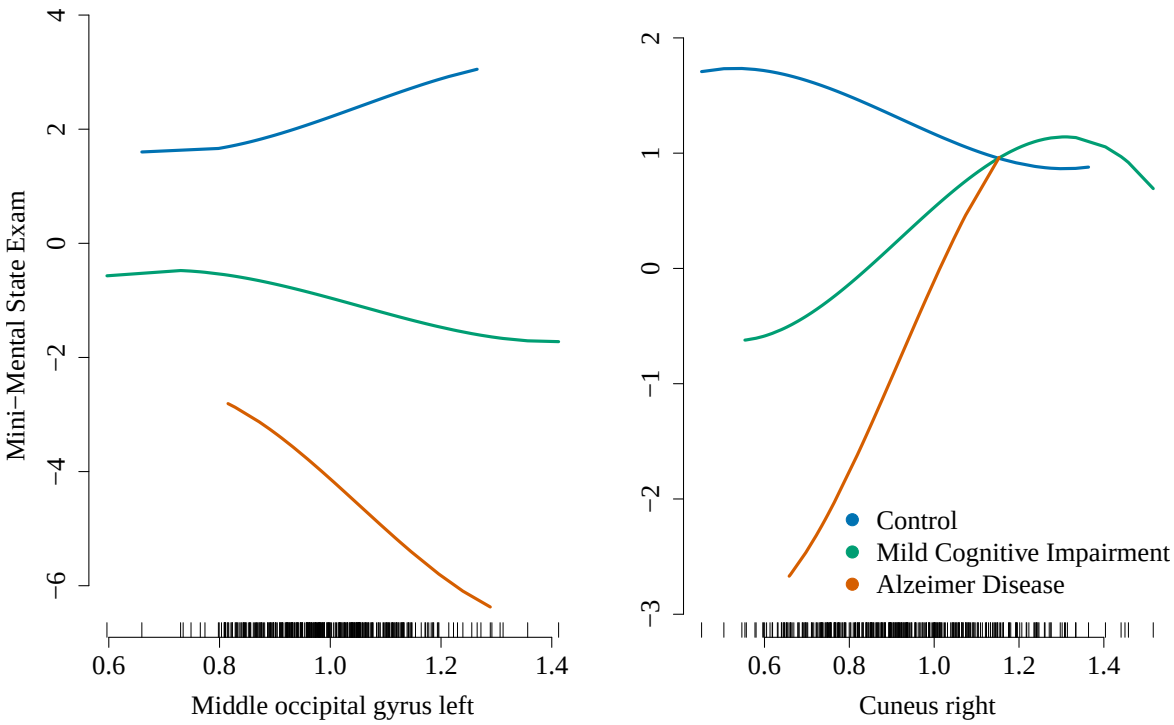


Figure 3.9: Estimated interaction effects by `sail` for the ADNI data.

3.5 Discussion

In this article we have introduced the sparse additive interaction learning model `sail` for detecting non-linear interactions with a key environmental or exposure variable in high-dimensional settings. Using a simple reparametrization, we are able to achieve either the weak or strong heredity property without using a complex penalty function. We developed a blockwise coordinate descent algorithm to solve the `sail` objective function for the least-squares loss function. All our algorithms are implemented in a computationally efficient, well-documented and freely available `R` package. Furthermore, our method is flexible enough to handle any type of basis expansion including the identity map, which allows for linear interactions. Our implementation allows the user to selectively apply the basis expansions to the predictors, allowing for example, a combination of continuous and categorical predictors. An extensive simulation study shows that `sail`, `adaptive sail` and `sail weak`

outperform existing penalized regression methods in terms of prediction error, sensitivity and specificity when there are non-linear main effects only, as well as interactions with an exposure variable.

Our method however does have its limitations. `sail`can currently only handle $X_E \cdot f(X)$ or $f(X_E) \cdot X$ and does not allow for $f(X, X_E)$, i.e., only one of the variables in the interaction can have a non-linear effect and we do not consider the tensor product. The reparametrization leads to a non-convex optimization problem which makes convergence rates difficult to assess, though we did not experience any major convergence issues in our simulations and real data analysis. The memory footprint can also be an issue depending on the degree of the basis expansion and the number of variables.

To our knowledge, our proposal is the first to allow for non-linear interactions with a key exposure variable following the weak or strong heredity property in high-dimensional settings. We also provide a first software implementation for these models.

Chapter 4

A General Framework for Variable Selection in Linear Mixed Models with Applications to Genetic Studies with Structured Populations

Sahir Rai Bhatnagar[1,2], Karim Oualkacha[3], Yi Yang[4], Marie Forest[2], Celia MT Greenwood[1,2]

[1]Department of Epidemiology, Biostatistics and Occupational Health, McGill University

[2]Lady Davis Institute, Jewish General Hospital, Montréal, QC

[3]Département de Mathématiques, Université de Québec À Montréal

[4]Department of Mathematics and Statistics, McGill University

Abstract

Complex traits are known to be influenced by a combination of environmental factors and rare and common genetic variants. However, detection of such multivariate associations can be compromised by low statistical power and confounding by population structure. Linear mixed effect models (LMM) can account for correlations due to relatedness but have not been applicable in high-dimensional (HD) settings where the number of fixed effect predictors greatly exceeds the number of samples. False positives can result from two-stage approaches, where the residuals estimated from a null model adjusted for the subjects' relationship structure are subsequently used as the response in a standard penalized regression model. To overcome these challenges, we develop a general penalized LMM framework called `ggmix` that simultaneously, in one step, selects variables and estimates their effects, while accounting for between individual correlations. Our method can accommodate several sparsity-inducing penalties such as the lasso, elastic net and group lasso, and also readily handles prior annotation information in the form of weights. We develop a blockwise coordinate descent algorithm which is highly scalable, computationally efficient and has theoretical guarantees of convergence. Through simulations, we show that `ggmix` leads to correct Type 1 error control and improved variance component estimation compared to the two-stage approach or principal component adjustment. `ggmix` is also robust to different kinship structures and heritability proportions. Our algorithms are available in an R package (https://github.com/greenwoodlab).

4.1 Introduction

Genome-wide association studies (GWAS) have become the standard method for analyzing genetic datasets owing to their success in identifying thousands of genetic variants associated with complex diseases (https://www.genome.gov/gwastudies/). Despite these impressive findings, the discovered markers have only been able to explain a small proportion of the phenotypic variance; this is known as the missing heritability problem (Manolio et al., 2009). One plausible explanation is that there are many causal variants that each explain a small amount of variation with small effect sizes (J. Yang et al., 2010). Methods such GWAS, which test each variant or single nucleotide polymorphism (SNP) independently, may miss these true associations due to the stringent significance thresholds required to reduce the number of false positives (Manolio et al., 2009). Another major issue to overcome is that of confounding due to geographic population structure, family and/or cryptic relatedness which can lead to spurious associations (Astle et al., 2009). For example, there may be subpopulations within a study that differ with respect to their genotype frequencies at a particular locus due to geographical location or their ancestry. This heterogeneity in genotype frequency can cause correlations with other loci and consequently mimic the signal of association even though there is no biological association (Marchini et al., 2004; Song et al., 2015). Studies that separate their sample by ethnicity to address this confounding suffer from a loss in statistical power.

To address the first problem, multivariable regression methods have been proposed which simultaneously fit many SNPs in a single model (Hoggart et al., 2008; Li et al., 2010). Indeed, the power to detect an association for a given SNP may be increased when other causal SNPs have been accounted for. Conversely, a stronger signal from a causal SNP may weaken false signals when modeled jointly (Hoggart et al., 2008).

Solutions for confounding by population structure have also received significant attention in the literature (Eu-Ahsunthornwattana et al., 2014; Kang et al., 2010; Lippert et al., 2011;

Yu et al., 2006). There are two main approaches to account for the relatedness between subjects: 1) the principal component (PC) adjustment method and 2) the linear mixed model (LMM). The PC adjustment method includes the top PCs of genome-wide SNP genotypes as additional covariates in the model (Price et al., 2006). The LMM uses an estimated covariance matrix from the individuals' genotypes and includes this information in the form of a random effect Astle et al. (2009).

While these problems have been addressed in isolation, there has been relatively little progress towards addressing them jointly at a large scale. Region-based tests of association have been developed where a linear combination of p variants is regressed on the response variable in a mixed model framework (Oualkacha et al., 2013). In case-control data, a stepwise logistic-regression procedure was used to evaluate the relative importance of variants within a small genetic region (Cordell & Clayton, 2002). These methods however are not applicable in the high-dimensional setting, i.e., when the number of variables p is much larger than the sample size n, as is often the case in genetic studies where millions of variants are measured on thousands of individuals.

There has been recent interest in using penalized linear mixed models, which place a constraint on the magnitude of the effect sizes while controlling for confounding factors such as population structure. For example, the LMM-lasso (Rakitsch et al., 2013) places a Laplace prior on all main effects while the adaptive mixed lasso (Wang et al., 2011) uses the L_1 penalty (Tibshirani, 1996) with adaptively chosen weights (Zou, 2006) to allow for differential shrinkage amongst the variables in the model. Another method applied a combination of both the lasso and group lasso penalties in order to select variants within a gene most associated with the response (Ding et al., 2014). However, these methods are normally performed in two steps. First, the variance components are estimated once from a LMM with a single random effect. These LMMs normally use the estimated covariance matrix from the individuals' genotypes to account for the relatedness but assumes no SNP main effects

(i.e. a null model). The residuals from this null model with a single random effect can be treated as independent observations because the relatedness has been effectively removed from the original response. In the second step, these residuals are used as the response in any high-dimensional model that assumes uncorrelated errors. This approach has both computational and practical advantages since existing penalized regression software such as `glmnet` (J. Friedman et al., 2010) and `gglasso` (Y. Yang & Zou, 2015), which assume independent observations, can be applied directly to the residuals. However, recent work has shown that there can be a loss in power if a causal variant is included in the calculation of the covariance matrix as its effect will have been removed in the first step (Oualkacha et al., 2013; J. Yang et al., 2014).

In this paper we develop a general penalized LMM framework called `ggmix` that simultaneously selects variables and estimates their effects, accounting for between-individual correlations. Our method can accommodate several sparsity inducing penalties such as the lasso (Tibshirani, 1996), elastic net (Zou & Hastie, 2005) and group lasso (Yuan & Lin, 2006). `ggmix` also readily handles prior annotation information in the form of a penalty factor, which can be useful, for example, when dealing with rare variants. We develop a blockwise coordinate descent algorithm which is highly scalable and has theoretical guarantees of convergence to a stationary point. All of our algorithms are implemented in the `ggmix` R package hosted on GitHub with extensive documentation (`http://sahirbhatnagar.com/ggmix/`). We provide a brief demonstration of the `ggmix` package in Appendix B.3.

The rest of the paper is organized as follows. Section 2 describes the `ggmix` model. Section 3 contains the optimization procedure and the algorithm used to fit the `ggmix` model. In Section 4, we compare the performance of our proposed approach and demonstrate the scenarios where it can be advantageous to use over existing methods through simulation studies. Section 5 discusses some limitations and future directions.

4.2 Penalized Linear Mixed Models

4.2.1 Model Set-up

Let $i = 1, \ldots, N$ be a grouping index, $j = 1, \ldots, n_i$ the observation index within a group and $N_T = \sum_{i=1}^{N} n_i$ the total number of observations. For each group let $\boldsymbol{y}_i = (y_1, \ldots, y_{n_i})$ be the observed vector of responses or phenotypes, $\mathbf{X}_i$ an $n_i \times (p+1)$ design matrix (with the column of 1s for the intercept), $\boldsymbol{b}_i$ a group-specific random effect vector of length n_i and $\boldsymbol{\varepsilon}_i = (\varepsilon_{i1}, \ldots, \varepsilon_{in_i})$ the individual error terms. Denote the stacked vectors $\mathbf{Y} = (\boldsymbol{y}_i, \ldots, \boldsymbol{y}_N)^T \in \mathbb{R}^{N_T \times 1}$, $\boldsymbol{b} = (\boldsymbol{b}_i, \ldots, \boldsymbol{b}_N)^T \in \mathbb{R}^{N_T \times 1}$, $\boldsymbol{\varepsilon} = (\boldsymbol{\varepsilon}_i, \ldots, \boldsymbol{\varepsilon}_N)^T \in \mathbb{R}^{N_T \times 1}$, and the stacked matrix
$\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_N)^T \in \mathbb{R}^{N_T \times (p+1)}$. Furthermore, let $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^T \in \mathbb{R}^{(p+1)\times 1}$ be a vector of fixed effects regression coefficients corresponding to $\mathbf{X}$. We consider the following linear mixed model with a single random effect (Pirinen et al., 2013):

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{b} + \boldsymbol{\varepsilon} \tag{4.1}$$

where the random effect $\boldsymbol{b}$ and the error variance $\boldsymbol{\varepsilon}$ are assigned the distributions

$$\boldsymbol{b} \sim \mathcal{N}(0, \eta\sigma^2\boldsymbol{\Phi}) \qquad \boldsymbol{\varepsilon} \sim \mathcal{N}(0, (1-\eta)\sigma^2\mathbf{I}) \tag{4.2}$$

Here, $\boldsymbol{\Phi}_{N_T \times N_T}$ is a known positive semi-definite and symmetric covariance or kinship matrix calculated from SNPs sampled across the genome, $\mathbf{I}_{N_T \times N_T}$ is the identity matrix and parameters σ^2 and $\eta \in [0, 1]$ determine how the variance is divided between $\boldsymbol{b}$ and $\boldsymbol{\varepsilon}$. Note that η is also the narrow-sense heritability (h^2), defined as the proportion of phenotypic variance attributable to the additive genetic factors (Manolio et al., 2009). The joint density of $\mathbf{Y}$ is

therefore multivariate normal:

$$\mathbf{Y}|(\boldsymbol{\beta}, \eta, \sigma^2) \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \eta\sigma^2\boldsymbol{\Phi} + (1-\eta)\sigma^2\mathbf{I}) \tag{4.3}$$

The LMM-Lasso method (Rakitsch et al., 2013) considers an alternative but equivalent parameterization given by:

$$\mathbf{Y}|(\boldsymbol{\beta}, \delta, \sigma_g^2) \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma_g^2(\boldsymbol{\Phi} + \delta\mathbf{I})) \tag{4.4}$$

where $\delta = \sigma_e^2/\sigma_g^2$, σ_g^2 is the genetic variance and σ_e^2 is the residual variance. We instead consider the parameterization in (4.3) since maximization is easier over the compact set $\eta \in [0, 1]$ than over the unbounded interval $\delta \in [0, \infty)$ (Pirinen et al., 2013). We define the complete parameter vector as $\boldsymbol{\Theta} := (\boldsymbol{\beta}, \eta, \sigma^2)$. The negative log-likelihood for (4.3) is given by

$$-\ell(\boldsymbol{\Theta}) \propto \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2}\log\left(\det(\mathbf{V})\right) + \frac{1}{2\sigma^2}\left(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\right)^T \mathbf{V}^{-1}\left(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\right) \tag{4.5}$$

where $\mathbf{V} = \eta\boldsymbol{\Phi} + (1-\eta)\mathbf{I}$ and $\det(\mathbf{V})$ is the determinant of $\mathbf{V}$.

Let $\boldsymbol{\Phi} = \mathbf{U}\mathbf{D}\mathbf{U}^T$ be the eigen (spectral) decomposition of the kinship matrix $\boldsymbol{\Phi}$, where $\mathbf{U}_{N_T \times N_T}$ is an orthonormal matrix of eigenvectors (i.e. $\mathbf{U}\mathbf{U}^T = \mathbf{I}$) and $\mathbf{D}_{N_T \times N_T}$ is a diagonal matrix of eigenvalues Λ_i. $\mathbf{V}$ can then be further simplified (Pirinen et al., 2013)

$$\begin{aligned}
\mathbf{V} &= \eta\boldsymbol{\Phi} + (1-\eta)\mathbf{I} \\
&= \eta\mathbf{U}\mathbf{D}\mathbf{U}^T + (1-\eta)\mathbf{U}\mathbf{I}\mathbf{U}^T \\
&= \mathbf{U}\eta\mathbf{D}\mathbf{U}^T + \mathbf{U}(1-\eta)\mathbf{I}\mathbf{U}^T \\
&= \mathbf{U}\left(\eta\mathbf{D} + (1-\eta)\mathbf{I}\right)\mathbf{U}^T \\
&= \mathbf{U}\widetilde{\mathbf{D}}\mathbf{U}^T
\end{aligned} \tag{4.6}$$

where

$$\widetilde{\mathbf{D}} = \eta\mathbf{D} + (1-\eta)\mathbf{I} \tag{4.7}$$

$$= \eta \begin{bmatrix} \Lambda_1 & & & \\ & \Lambda_2 & & \\ & & \ddots & \\ & & & \Lambda_{N_T} \end{bmatrix} + (1-\eta) \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1+\eta(\Lambda_1-1) & & & \\ & 1+\eta(\Lambda_2-1) & & \\ & & \ddots & \\ & & & 1+\eta(\Lambda_{N_T}-1) \end{bmatrix}$$

$$= \operatorname{diag}\left\{1+\eta(\Lambda_1-1), 1+\eta(\Lambda_2-1), \ldots, 1+\eta(\Lambda_{N_T}-1)\right\} \tag{4.8}$$

Since (4.7) is a diagonal matrix, its inverse is also a diagonal matrix:

$$\widetilde{\mathbf{D}}^{-1} = \operatorname{diag}\left\{\frac{1}{1+\eta(\Lambda_1-1)}, \frac{1}{1+\eta(\Lambda_2-1)}, \ldots, \frac{1}{1+\eta(\Lambda_{N_T}-1)}\right\} \tag{4.9}$$

From (4.6) and (4.8), $\log(\det(\mathbf{V}))$ simplifies to

$$\begin{aligned} \log(\det(\mathbf{V})) &= \log\left(\det(\mathbf{U})\det\left(\widetilde{\mathbf{D}}\right)\det(\mathbf{U}^T)\right) \\ &= \log\left\{\prod_{i=1}^{N_T}(1+\eta(\Lambda_i-1))\right\} \\ &= \sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i-1)) \end{aligned} \tag{4.10}$$

since $\det(\mathbf{U}) = 1$. It also follows from (4.6) that

$$\begin{aligned}\mathbf{V}^{-1} &= \left(\mathbf{U}\widetilde{\mathbf{D}}\mathbf{U}^T\right)^{-1} \\ &= \left(\mathbf{U}^T\right)^{-1}\left(\widetilde{\mathbf{D}}\right)^{-1}\mathbf{U}^{-1} \\ &= \mathbf{U}\widetilde{\mathbf{D}}^{-1}\mathbf{U}^T \end{aligned} \tag{4.11}$$

since for an orthonormal matrix $\mathbf{U}^{-1} = \mathbf{U}^T$. Substituting (4.9), (4.10) and (4.11) into (4.5) the negative log-likelihood becomes

$$-\ell(\boldsymbol{\Theta}) \propto \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2}\sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i - 1)) + \frac{1}{2\sigma^2}\left(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\right)^T \mathbf{U}\widetilde{\mathbf{D}}^{-1}\mathbf{U}^T\left(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\right) \tag{4.12}$$

$$\begin{aligned} &= \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2}\sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i - 1)) + \frac{1}{2\sigma^2}\left(\mathbf{U}^T\mathbf{Y} - \mathbf{U}^T\mathbf{X}\boldsymbol{\beta}\right)^T \widetilde{\mathbf{D}}^{-1}\left(\mathbf{U}^T\mathbf{Y} - \mathbf{U}^T\mathbf{X}\boldsymbol{\beta}\right) \\ &= \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2}\sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i - 1)) + \frac{1}{2\sigma^2}\left(\widetilde{\mathbf{Y}} - \widetilde{\mathbf{X}}\boldsymbol{\beta}\right)^T \widetilde{\mathbf{D}}^{-1}\left(\widetilde{\mathbf{Y}} - \widetilde{\mathbf{X}}\boldsymbol{\beta}\right) \\ &= \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2}\sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i - 1)) + \frac{1}{2\sigma^2}\sum_{i=1}^{N_T}\frac{\left(\widetilde{Y}_i - \sum_{j=0}^{p}\widetilde{X}_{ij+1}\beta_j\right)^2}{1+\eta(\Lambda_i - 1)} \end{aligned} \tag{4.13}$$

where $\widetilde{\mathbf{Y}} = \mathbf{U}^T\mathbf{Y}$, $\widetilde{\mathbf{X}} = \mathbf{U}^T\mathbf{X}$, $\widetilde{Y}_i$ denotes the i^{th} element of $\widetilde{\mathbf{Y}}$, $\widetilde{X}_{ij}$ is the i, j^{th} entry of $\widetilde{\mathbf{X}}$ and $\mathbf{1}$ is a column vector of N_T ones.

4.2.2 Penalized Maximum Likelihood Estimator

We define the $p+3$ length vector of parameters $\boldsymbol{\Theta} := (\Theta_0, \Theta_1, \ldots, \Theta_{p+1}, \Theta_{p+2}, \Theta_{p+3}) = (\boldsymbol{\beta}, \eta, \sigma^2)$ where $\boldsymbol{\beta} \in \mathbb{R}^{p+1}, \eta \in [0,1], \sigma^2 > 0$. In what follows, $p+2$ and $p+3$ are the indices in $\boldsymbol{\Theta}$ for η and σ^2, respectively. In light of our goals to select variables associated with the response in high-dimensional data, we propose to place a constraint on the magnitude of the regression coefficients. This can be achieved by adding a penalty term to the likelihood

function (4.13). The penalty term is a necessary constraint because in our applications, the sample size is much smaller than the number of predictors. We define the following objective function:

$$Q_\lambda(\boldsymbol{\Theta}) = f(\boldsymbol{\Theta}) + \lambda \sum_{j \neq 0} v_j P_j(\beta_j) \tag{4.14}$$

where $f(\boldsymbol{\Theta}) \coloneqq -\ell(\boldsymbol{\Theta})$ is defined in (4.13), $P_j(\cdot)$ is a penalty term on the fixed regression coefficients $\beta_1, \ldots, \beta_{p+1}$ (we do not penalize the intercept) controlled by the nonnegative regularization parameter λ, and v_j is the penalty factor for jth covariate. These penalty factors serve as a way of allowing parameters to be penalized differently. Note that we do not penalize η or σ^2. An estimate of the regression parameters $\widehat{\boldsymbol{\Theta}}_\lambda$ is obtained by

$$\widehat{\boldsymbol{\Theta}}_\lambda = \arg\min_{\boldsymbol{\Theta}} Q_\lambda(\boldsymbol{\Theta}) \tag{4.15}$$

This is the general set-up for our model. In Section 4.3 we provide more specific details on how we solve (4.15).

4.3 Computational Algorithm

We use a general purpose block coordinate gradient descent algorithm (CGD) (Tseng & Yun, 2009) to solve (4.15). At each iteration, we cycle through the coordinates and minimize the objective function with respect to one coordinate only. For continuously differentiable $f(\cdot)$ and convex and block-separable $P(\cdot)$ (i.e. $P(\boldsymbol{\beta}) = \sum_i P_i(\beta_i)$), Tseng and Yun Tseng & Yun (2009) show that the solution generated by the CGD method is a stationary point of $Q_\lambda(\cdot)$ if the coordinates are updated in a Gauss-Seidel manner i.e. $Q_\lambda(\cdot)$ is minimized with respect to one parameter while holding all others fixed. The CGD algorithm has been successfully applied in fixed effects models (e.g. Meier et al. (2008), J. Friedman et al. (2010)) and linear mixed models with an ℓ_1 penalty Schelldorfer et al. (2011). In the next section we provide some brief details about Algorithm 6. A more thorough treatment of the algorithm is given

in Appendix B.1.

We emphasize here that previously developed methods such as the LMM-lasso (Rakitsch et al., 2013) use a two-stage fitting procedure without any convergence details. From a practical point of view, there is currently no implementation that provides a principled way of determining the sequence of tuning parameters to fit, nor a procedure that automatically selects the optimal value of λ. To our knowledge, we are the first to develop a CGD algorithm in the specific context of fitting a penalized LMM for population structure correction with theoretical guarantees of convergence. Furthermore, we develop a principled method for automatic tuning parameter selection and provide an easy-to-use software implementation in order to promote wider uptake of these more complex methods by applied practitioners.

Algorithm 6 Block Coordinate Gradient Descent

Set the iteration counter $k \leftarrow 0$, initial values for the parameter vector $\boldsymbol{\Theta}^{(0)}$ and convergence threshold ϵ
for $\lambda \in \{\lambda_{max}, \ldots, \lambda_{min}\}$ **do**
  repeat
    $\boldsymbol{\beta}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{\beta}} Q_\lambda\left(\boldsymbol{\beta}, \eta^{(k)}, \sigma^{2\ (k)}\right)$
    $\eta^{(k+1)} \leftarrow \arg\min_{\eta} Q_\lambda\left(\boldsymbol{\beta}^{(k+1)}, \eta, \sigma^{2\ (k)}\right)$
    $\sigma^{2\ (k+1)} \leftarrow \arg\min_{\sigma^2} Q_\lambda\left(\boldsymbol{\beta}^{(k+1)}, \eta^{(k+1)}, \sigma^2\right)$
    $k \leftarrow k + 1$
  until convergence criterion is satisfied: $\left\|\boldsymbol{\Theta}^{(k+1)} - \boldsymbol{\Theta}^{(k)}\right\|_2 < \epsilon$

4.3.1 Updates for the β parameter

Recall that the part of the objective function that depends on $\boldsymbol{\beta}$ has the form

$$Q_\lambda(\boldsymbol{\Theta}) = \frac{1}{2}\sum_{i=1}^{N_T} w_i \left(\widetilde{Y}_i - \sum_{j=0}^{p} \widetilde{X}_{ij+1}\beta_j\right)^2 + \lambda \sum_{j=1}^{p} v_j|\beta_j| \tag{4.16}$$

where

$$w_i \coloneqq \frac{1}{\sigma^2\left(1+\eta(\Lambda_i-1)\right)} \tag{4.17}$$

Conditional on $\eta^{(k)}$ and $\sigma^{2\,(k)}$, it can be shown that the solution for β_j, $j=1,\ldots,p$ is given by

$$\beta_j^{(k+1)} \leftarrow \frac{\mathcal{S}_\lambda\left(\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}\left(\widetilde{Y}_i - \sum_{\ell\neq j}\widetilde{X}_{i\ell}\beta_\ell^{(k)}\right)\right)}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} \tag{4.18}$$

where $\mathcal{S}_\lambda(x)$ is the soft-thresholding operator

$$\mathcal{S}_\lambda(x) = \text{sign}(x)(|x|-\lambda)_+$$

$\text{sign}(x)$ is the signum function

$$\text{sign}(x) = \begin{cases} -1 & x<0 \\ 0 & x=0 \\ 1 & x>0 \end{cases}$$

and $(x)_+ = \max(x,0)$. We provide the full derivation in Appendix B.1.1.

4.3.2 Updates for the η paramter

Given $\boldsymbol{\beta}^{(k+1)}$ and $\sigma^{2\,(k)}$, solving for $\eta^{(k+1)}$ becomes a univariate optimization problem:

$$\eta^{(k+1)} \leftarrow \arg\min_{\eta} \frac{1}{2}\sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i-1)) + \frac{1}{2\sigma^{2\,(k)}}\sum_{i=1}^{N_T}\frac{\left(\widetilde{Y}_i - \sum_{j=0}^{p}\widetilde{X}_{ij+1}\beta_j^{(k+1)}\right)^2}{1+\eta(\Lambda_i-1)} \tag{4.19}$$

We use a bound constrained optimization algorithm (Byrd et al., 1995) implemented in the `optim` function in `R` and set the lower and upper bounds to be 0.01 and 0.99, respec-

tively.

4.3.3 Updates for the σ^2 parameter

Conditional on $\boldsymbol{\beta}^{(k+1)}$ and $\eta^{(k+1)}$, ${\sigma^2}^{(k+1)}$ can be solved for using the following equation:

$$ {\sigma^2}^{(k+1)} \leftarrow \arg\min_{\sigma^2} \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2\sigma^2}\sum_{i=1}^{N_T}\frac{\left(\widetilde{Y}_i - \sum_{j=0}^{p}\widetilde{X}_{ij+1}\beta_j\right)^2}{1+\eta(\Lambda_i - 1)} \tag{4.20} $$

There exists an analytic solution for (4.20) given by:

$$ {\sigma^2}^{(k+1)} \leftarrow \frac{1}{N_T}\sum_{i=1}^{N_T}\frac{\left(\widetilde{Y}_i - \sum_{j=0}^{p}\widetilde{X}_{ij+1}\beta_j^{(k+1)}\right)^2}{1+\eta^{(k+1)}(\Lambda_i - 1)} \tag{4.21} $$

4.3.4 Regularization path

In this section we describe how determine the sequence of tuning parameters λ at which to fit the model. Recall that our objective function has the form

$$ Q_\lambda(\boldsymbol{\Theta}) = \frac{N_T}{2}\log(\sigma^2) + \frac{1}{2}\sum_{i=1}^{N_T}\log(1+\eta(\Lambda_i - 1)) + \frac{1}{2}\sum_{i=1}^{N_T} w_i\left(\widetilde{Y}_i - \sum_{j=0}^{p}\widetilde{X}_{ij+1}\beta_j\right)^2 + \lambda\sum_{j=1}^{p} v_j|\beta_j| \tag{4.22} $$

The Karush-Kuhn-Tucker (KKT) optimality conditions for (4.22) are given by:

$$ \begin{aligned} \frac{\partial}{\partial\beta_1,\ldots,\beta_p}Q_\lambda(\boldsymbol{\Theta}) &= \mathbf{0}_p \\ \frac{\partial}{\partial\beta_0}Q_\lambda(\boldsymbol{\Theta}) &= 0 \\ \frac{\partial}{\partial\eta}Q_\lambda(\boldsymbol{\Theta}) &= 0 \\ \frac{\partial}{\partial\sigma^2}Q_\lambda(\boldsymbol{\Theta}) &= 0 \end{aligned} \tag{4.23} $$

The equations in (4.23) are equivalent to

$$\begin{aligned}
\sum_{i=1}^{N_T} w_i \widetilde{X}_{i1} \left(\widetilde{Y}_i - \sum_{j=0}^{p} \widetilde{X}_{ij+1} \beta_j \right) &= 0 \\
\frac{1}{v_j} \sum_{i=1}^{N_T} w_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \sum_{j=0}^{p} \widetilde{X}_{ij+1} \beta_j \right) &= \lambda \gamma_j, \\
\gamma_j \in \begin{cases} \text{sign}(\hat{\beta}_j) & \text{if} \quad \hat{\beta}_j \neq 0 \\ [-1, 1] & \text{if} \quad \hat{\beta}_j = 0 \end{cases}, &\qquad \text{for } j = 1, \ldots, p \\
\frac{1}{2} \sum_{i=1}^{N_T} \frac{\Lambda_i - 1}{1 + \eta(\Lambda_i - 1)} \left(1 - \frac{\left(\widetilde{Y}_i - \sum_{j=0}^{p} \widetilde{X}_{ij+1} \beta_j \right)^2}{\sigma^2 (1 + \eta(\Lambda_i - 1))} \right) &= 0 \\
\sigma^2 - \frac{1}{N_T} \sum_{i=1}^{N_T} \frac{\left(\widetilde{Y}_i - \sum_{j=0}^{p} \widetilde{X}_{ij+1} \beta_j \right)^2}{1 + \eta(\Lambda_i - 1)} &= 0
\end{aligned} \tag{4.24}$$

where w_i is given by (4.17), $\widetilde{\mathbf{X}}_{-1}^T$ is $\widetilde{\mathbf{X}}^T$ with the first column removed, $\widetilde{\mathbf{X}}_1^T$ is the first column of $\widetilde{\mathbf{X}}^T$, and $\boldsymbol{\gamma} \in \mathbb{R}^p$ is the subgradient function of the ℓ_1 norm evaluated at $(\hat{\beta}_1, \ldots, \hat{\beta}_p)$. Therefore $\widehat{\boldsymbol{\Theta}}$ is a solution in (4.15) if and only if $\widehat{\boldsymbol{\Theta}}$ satisfies (4.24) for some γ. We can determine a decreasing sequence of tuning parameters by starting at a maximal value for $\lambda = \lambda_{max}$ for which $\hat{\beta}_j = 0$ for $j = 1, \ldots, p$. In this case, the KKT conditions in (4.24) are equivalent to

$$\begin{aligned}
\frac{1}{v_j} \sum_{i=1}^{N_T} \left| w_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \widetilde{X}_{i1} \beta_0 \right) \right| &\leq \lambda, \quad \forall j = 1, \ldots, p \\
\beta_0 &= \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{i1} \widetilde{Y}_i}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{i1}^2} \\
\frac{1}{2} \sum_{i=1}^{N_T} \frac{\Lambda_i - 1}{1 + \eta(\Lambda_i - 1)} \left(1 - \frac{\left(\widetilde{Y}_i - \widetilde{X}_{i1} \beta_0 \right)^2}{\sigma^2 (1 + \eta(\Lambda_i - 1))} \right) &= 0 \\
\sigma^2 &= \frac{1}{N_T} \sum_{i=1}^{N_T} \frac{\left(\widetilde{Y}_i - \widetilde{X}_{i1} \beta_0 \right)^2}{1 + \eta(\Lambda_i - 1)}
\end{aligned} \tag{4.25}$$

We can solve the KKT system of equations in (4.25) (with a numerical solution for η) in

order to have an explicit form of the stationary point $\widehat{\boldsymbol{\Theta}}_0 = \left\{\hat{\beta}_0, \mathbf{0}_p, \hat{\eta}, \widehat{\sigma}^2\right\}$. Once we have $\widehat{\boldsymbol{\Theta}}_0$, we can solve for the smallest value of λ such that the entire vector $(\hat{\beta}_1, \ldots, \hat{\beta}_p)$ is 0:

$$\lambda_{max} = \max_j \left\{ \left| \frac{1}{v_j} \sum_{i=1}^{N_T} \hat{w}_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \widetilde{X}_{i1} \hat{\beta}_0 \right) \right| \right\}, \quad j = 1, \ldots, p \tag{4.26}$$

Following Friedman et al. J. Friedman et al. (2010), we choose $\tau\lambda_{max}$ to be the smallest value of tuning parameters λ_{min}, and construct a sequence of K values decreasing from λ_{max} to λ_{min} on the log scale. The defaults are set to $K = 100$, $\tau = 0.01$ if $n < p$ and $\tau = 0.001$ if $n \geq p$.

4.3.5 Warm Starts

The way in which we have derived the sequence of tuning parameters using the KKT conditions, allows us to implement warm starts. That is, the solution $\widehat{\boldsymbol{\Theta}}$ for λ_k is used as the initial value $\boldsymbol{\Theta}^{(0)}$ for λ_{k+1}. This strategy leads to computational speedups and has been implemented in the `ggmix` R package.

4.3.6 Prediction of the random effects

We use an empirical Bayes approach (e.g. Wakefield (2013)) to predict the random effects $\boldsymbol{b}$. Let the maximum a posteriori (MAP) estimate be defined as

$$\widehat{\boldsymbol{b}} = \arg\max_{\boldsymbol{b}} f(\boldsymbol{b}|\mathbf{Y}, \boldsymbol{\beta}, \eta, \sigma^2) \tag{4.27}$$

where, by using Bayes rule, $f(\boldsymbol{b}|\mathbf{Y}, \boldsymbol{\beta}, \eta, \sigma^2)$ can be expressed as

$$\begin{aligned} f(\boldsymbol{b}|\mathbf{Y}, \boldsymbol{\beta}, \eta, \sigma^2) &= \frac{f(\mathbf{Y}|\boldsymbol{b}, \boldsymbol{\beta}, \eta, \sigma^2)\pi(\boldsymbol{b}|\eta, \sigma^2)}{f(\mathbf{Y}|\boldsymbol{\beta}, \eta, \sigma^2)} \\ &\propto f(\mathbf{Y}|\boldsymbol{b}, \boldsymbol{\beta}, \eta, \sigma^2)\pi(\boldsymbol{b}|\eta, \sigma^2) \\ &\propto \exp\left\{-\frac{1}{2\sigma^2}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b})^T\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b}) - \frac{1}{2\eta\sigma^2}\boldsymbol{b}^T\boldsymbol{\Phi}^{-1}\boldsymbol{b}\right\} \\ &= \exp\left\{-\frac{1}{2\sigma^2}\left[(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b})^T\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b}) + \frac{1}{\eta}\boldsymbol{b}^T\boldsymbol{\Phi}^{-1}\boldsymbol{b}\right]\right\} \end{aligned} \tag{4.28}$$

Solving for (4.27) is equivalent to minimizing the exponent in (4.28):

$$\widehat{\boldsymbol{b}} = \arg\min_{\boldsymbol{b}}\left\{(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b})^T\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b}) + \frac{1}{\eta}\boldsymbol{b}^T\boldsymbol{\Phi}^{-1}\boldsymbol{b}\right\} \tag{4.29}$$

Taking the derivative of (4.29) with respect to $\boldsymbol{b}$ and setting it to 0 we get:

$$\begin{aligned} 0 &= -2\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta} - \boldsymbol{b}) + \frac{2}{\eta}\boldsymbol{\Phi}^{-1}\boldsymbol{b} \\ &= -\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) + \left(\mathbf{V}^{-1} + \frac{1}{\eta}\boldsymbol{\Phi}^{-1}\right)\boldsymbol{b} \\ \widehat{\boldsymbol{b}} &= \left(\mathbf{V}^{-1} + \frac{1}{\widehat{\eta}}\boldsymbol{\Phi}^{-1}\right)^{-1}\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) \\ &= \left(\mathbf{U}\widetilde{\mathbf{D}}^{-1}\mathbf{U}^T + \frac{1}{\widehat{\eta}}\mathbf{U}\mathbf{D}^{-1}\mathbf{U}^T\right)^{-1}\mathbf{U}\widetilde{\mathbf{D}}^{-1}\mathbf{U}^T(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) \\ &= \left(\mathbf{U}\left[\widetilde{\mathbf{D}}^{-1} + \frac{1}{\widehat{\eta}}\mathbf{D}^{-1}\right]\mathbf{U}^T\right)^{-1}\mathbf{U}\widetilde{\mathbf{D}}^{-1}(\widetilde{\mathbf{Y}} - \widetilde{\mathbf{X}}\widehat{\boldsymbol{\beta}}) \\ &= \mathbf{U}\left[\widetilde{\mathbf{D}}^{-1} + \frac{1}{\widehat{\eta}}\mathbf{D}^{-1}\right]^{-1}\mathbf{U}^T\mathbf{U}\widetilde{\mathbf{D}}^{-1}(\widetilde{\mathbf{Y}} - \widetilde{\mathbf{X}}\widehat{\boldsymbol{\beta}}) \end{aligned}$$

where $\mathbf{V}^{-1}$ is given by (4.11), and $(\widehat{\boldsymbol{\beta}}, \widehat{\eta})$ are the estimates obtained from Algorithm 6.

4.3.7 Choice of the optimal tuning parameter

In order to choose the optimal value of the tuning parameter λ, we use the generalized information criterion (Nishii, 1984) (GIC):

$$GIC_\lambda = -2\ell(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}^2, \widehat{\eta}) + a_n \cdot \widehat{df}_\lambda \tag{4.30}$$

where $\widehat{df}_\lambda$ is the number of non-zero elements in $\widehat{\boldsymbol{\beta}}_\lambda$ (Zou et al., 2007) plus two (representing the variance parameters η and σ^2). Several authors have used this criterion for variable selection in mixed models with $a_n = \log N_T$ (Bondell et al., 2010; Schelldorfer et al., 2011), which corresponds to the BIC. We instead choose the high-dimensional BIC (Y. Fan & Tang, 2013) given by $a_n = \log(\log(N_T)) * \log(p)$. This is the default choice in our `ggmix` R package, though the interface is flexible to allow the user to select their choice of a_n.

4.4 Simulation Study

To assess the performance of `ggmix`, we simulated random genotypes from the BN-PSD admixture model using the `bnpsd` package (Ochoa & Storey, 2016a,b). We used a block diagonal kinship structure with 5 subpopulations. In Figure 4.1, we plot an estimated kinship matrix ($\boldsymbol{\Phi}$), based on a single simulated dataset, in the form of a heatmap. Each block represents a subpopulation, and a darker color indicates a closer genetic relationship.

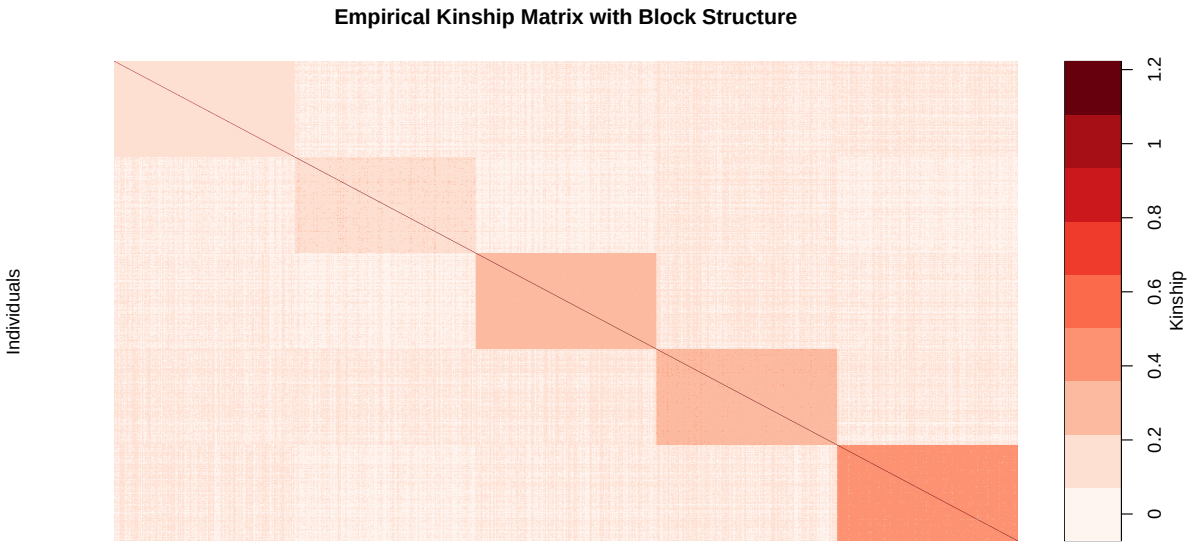


Figure 4.1: Empirical kinship matrix with block diagnoal structure used in simulation studies. Each block represents a subpopulation.

In Figure 4.2 we plot the first two principal component scores calculated from the block diagonal kinship matrix in Figure 4.1, and color each point by subpopulation membership. We can see that the PCs can identify the subpopulations which is why including them as additional covariates in a regression model has been considered a reasonable approach to control for confounding.

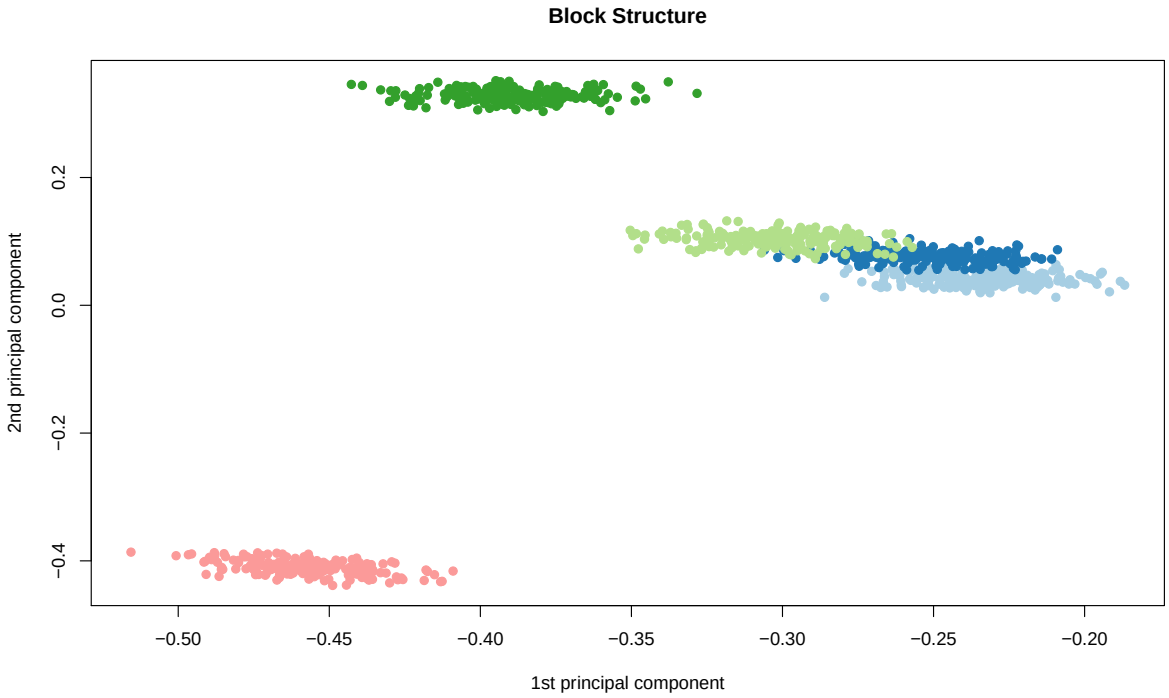


Figure 4.2: First two principal component scores of the block diagonal kinship matrix where each color represents one of the 5 simulated subpopulations.

For other parameters in our simulation study, we define the following quantities:

- c: percentage of causal SNPs
- $\mathbf{X}^{(fixed)}$: $n \times p_{fixed}$ matrix of SNPs that will be included as fixed effects in our model.
- $\mathbf{X}^{(causal)}$: $n \times (c * p_{fixed})$ matrix of SNPs that will be truly associated with the simulated phenotype, where $\mathbf{X}^{(causal)} \subseteq \mathbf{X}^{(fixed)}$
- $\mathbf{X}^{(other)}$: $n \times p_{other}$ matrix of SNPs that will be used in the construction of the kinship matrix. Some of these $\mathbf{X}^{(other)}$ SNPs, in conjunction with some of the SNPs in $\mathbf{X}^{(test)}$ will be used in construction of the kinship matrix. We will alter the balance between these two contributors and with the proportion of causal SNPs used to calculate kinship.
- $\mathbf{X}^{(kinship)}$: $n \times k$ matrix of SNPs used to construct the kinship matrix.

- β_j: effect size for the j^{th} SNP, simulated from a $Uniform(0.3, 0.7)$ distribution for $j = 1, \ldots, (c * p_{fixed})$

We simulate data from the model

$$\mathbf{Y} = \mathbf{X}^{(fixed)}\boldsymbol{\beta} + \mathbf{P} + \boldsymbol{\varepsilon} \tag{4.31}$$

where $\mathbf{P} \sim \mathcal{N}(0, \eta\sigma^2\mathbf{\Phi})$ and $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, (1-\eta)\sigma^2\mathbf{I})$. The values of the parameters that we used were as follows: narrow sense heritability $\eta = \{0.1, 0.5\}$, sample size $n = 1000$, number of fixed effects $p_{fixed} = 5000$, number of SNPs used to calculate the kinship matrix $k = 10000$, percentage of causal SNPs $c = \{0\%, 1\%\}$ and $\sigma^2 = 1$. In addition to these parameters, we also varied the amount of overlap between the causal SNPs and the SNPs used to generate the kinship matrix. We considered two main scenarios:

1. None of the causal SNPs are included in the calculation of the kinship matrix:

$$\mathbf{X}^{(kinship)} = \left[\mathbf{X}^{(other)}\right]$$

2. All the causal SNPs are included in the calculation of the kinship matrix:

$$\mathbf{X}^{(kinship)} = \left[\mathbf{X}^{(other)}; \mathbf{X}^{(causal)}\right].$$

Both kinship matrices are meant to contrast the model behavior when the causal SNPs are included in both the main effects and random effects versus when the causal SNPs are only included in the main effects. These scenarios are motivated by the current standard of practice in GWAS where the candidate marker is excluded from the calculation of the kinship matrix (Lippert et al., 2011). This approach becomes much more difficult to apply in large-scale multivariable models where there is likely to be overlap between the variables in the design matrix and kinship matrix.

We compare `ggmix` to the lasso and the twostep method. For the lasso, we include the first 10 principal components of the estimated kinship as unpenalized predictors in the design matrix. For the twostep method, we first fit an intercept only model with a single random effect using the average information restricted maximum likelihood (AIREML) algorithm (Gilmour et al., 1995) as implemented in the `gaston` R package (Dandine-Roulland, 2018). The residuals from this model are then used as the response in a regular lasso model. Note that in the twostep method, we have removed the kinship effect in the first step and therefore do not need to make any further adjustments when fitting the penalized model. We fit the lasso using the default settings in the `glmnet` package (J. Friedman et al., 2010) and select the optimal value of the regularization parameter using 10-fold cross-validation.

Let $\hat{\lambda}$ be the estimated value of the optimal regularization parameter selected via cross-validation or GIC, $\widehat{\boldsymbol{\beta}}_{\hat{\lambda}}$ the estimate of $\boldsymbol{\beta}$ at regularization parameter $\hat{\lambda}$, $S_0 = \{j; (\boldsymbol{\beta})_j \neq 0\}$ the index of the true active set, $\widehat{S}_{\hat{\lambda}} = \left\{j; (\widehat{\boldsymbol{\beta}}_{\hat{\lambda}})_j \neq 0\right\}$ the index of the set of non-zero estimated coefficients, and $|A|$ the cardinality of set A.

We evaluate the methods based on correct sparsity defined as $\frac{1}{p}\sum_{j=1}^{p} A_j$, where

$$A_j = \begin{cases} 1 & \text{if } (\widehat{\boldsymbol{\beta}}_{\hat{\lambda}})_j = (\boldsymbol{\beta})_j = 0 \\ 1 & \text{if } (\widehat{\boldsymbol{\beta}}_{\hat{\lambda}})_j \neq 0, (\boldsymbol{\beta})_j \neq 0 \\ 0 & \text{if else.} \end{cases}$$

We also compare the model error ($\|\mathbf{X}\boldsymbol{\beta} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\hat{\lambda}}\|_2$), true positive rate ($|\widehat{S}_{\hat{\lambda}} \in S_0|/|S_0|$), false positive rate ($|\widehat{S}_{\hat{\lambda}} \notin S_0|/|j \notin S_0|$), and the variance components for the random effect and error term. The following estimator is used for the error variance of the lasso (Reid et al., 2016):

$$\frac{1}{n - \widehat{S}_{\hat{\lambda}}} \left\| \mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}_{\hat{\lambda}} \right\|_2^2 \tag{4.32}$$

4.4.1 Results

We first plot the correct sparsity results for the null model ($c = 0$) and the model with 1% causal SNPs ($c = 0.01$) in Figures 4.3 and 4.4, respectively. When the true model has no causal SNPs, we see that `ggmix` has perfect Type 1 error control across all 200 replications while both the twostep and lasso methods sometimes estimate a model with a large number of false positives. When the true model contains some causal SNPs, `ggmix` again outperforms the other two methods in terms of correct sparsity. The distribution of $\widehat{S}_{\hat{\lambda}}$ for each of the three methods is shown in Figure B.1 for $c = 0$ and Figure B.2 for $c = 0.01$ of Supplemental Section B.2.

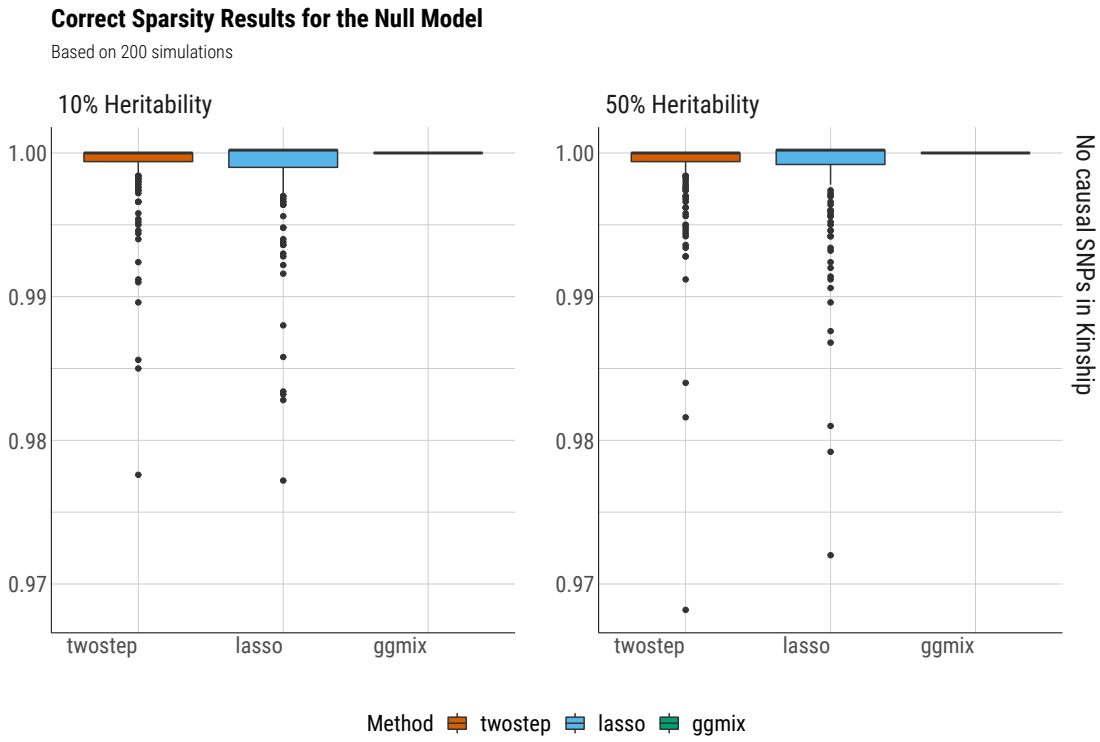


Figure 4.3: Boxplots of the correct sparsity from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ for the null model ($c = 0$).

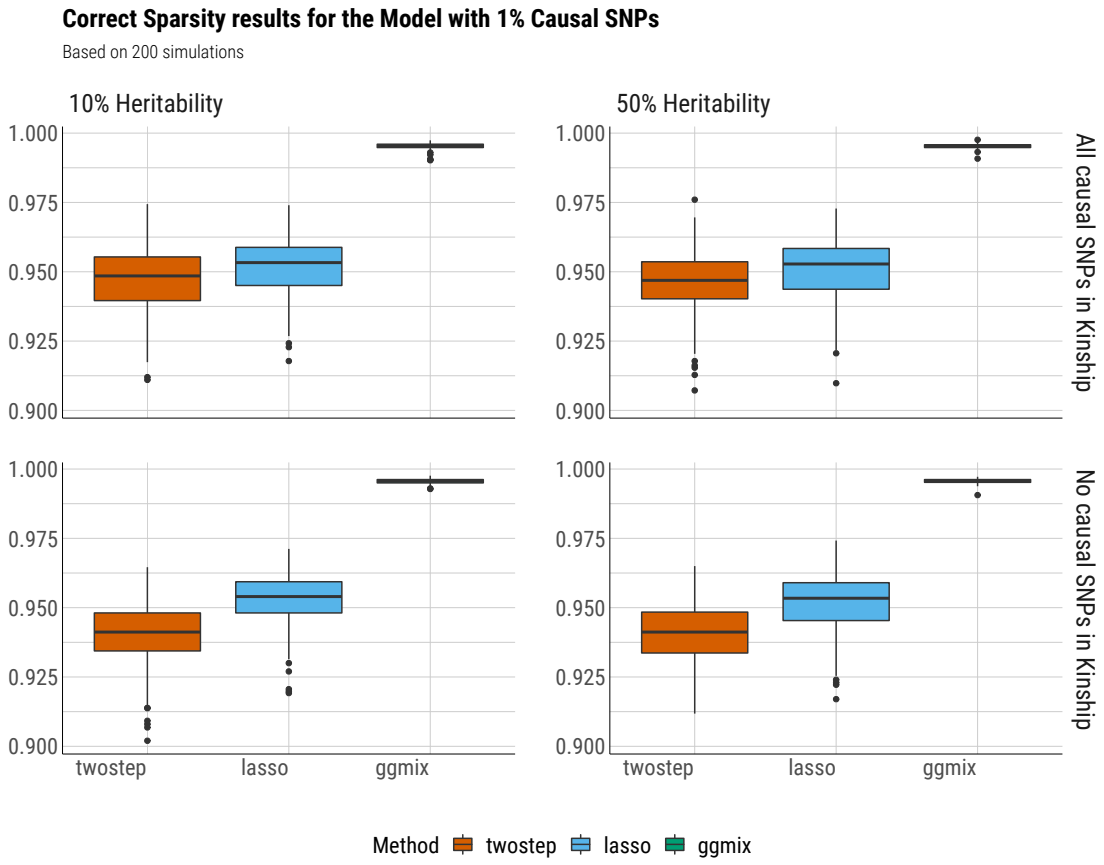


Figure 4.4: Boxplots of the correct sparsity from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ and number of causal SNPs that were included in the calculation of the kinship matrix for the model with 1% causal SNPs ($c = 0.01$).

The true positive vs. false positive rate for the model with 1% causal SNPs ($c = 0.01$) is shown in Figure 4.5. Both the lasso and twostep outperform `ggmix` in terms of identifying the true model. This accuracy however, comes at the cost of a very high false positive rate compared to `ggmix`.

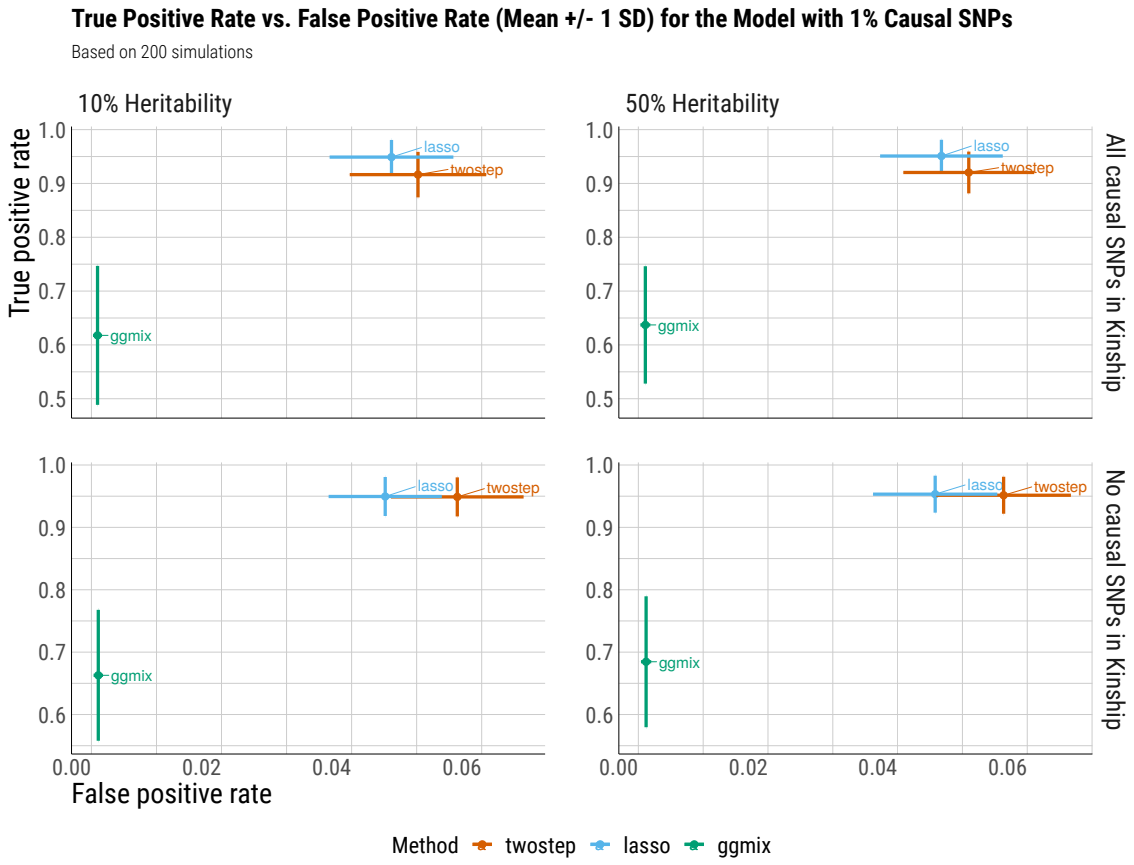


Figure 4.5: Means ± 1 standard deviation of true positive rate vs. false positive rate from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ and number of causal SNPs that were included in the calculation of the kinship matrix for the model with 1% causal SNPs ($c = 0.01$).

We plot the twostep and `ggmix` heritability estimates for $c = 0$ (Figure B.3, Supplemental Section B.2) and $c = 0.01$ (Figure 4.6). We see that both methods correctly estimate the heritability in the null model. When all of the causal SNPs are in the kinship matrix, both methods overestimate η though `ggmix` is closer to the true value. When none of the causal SNPs are in the kinship, both methods tend to overestimate the truth when $\eta = 10\%$ and underestimate when $\eta = 50\%$.

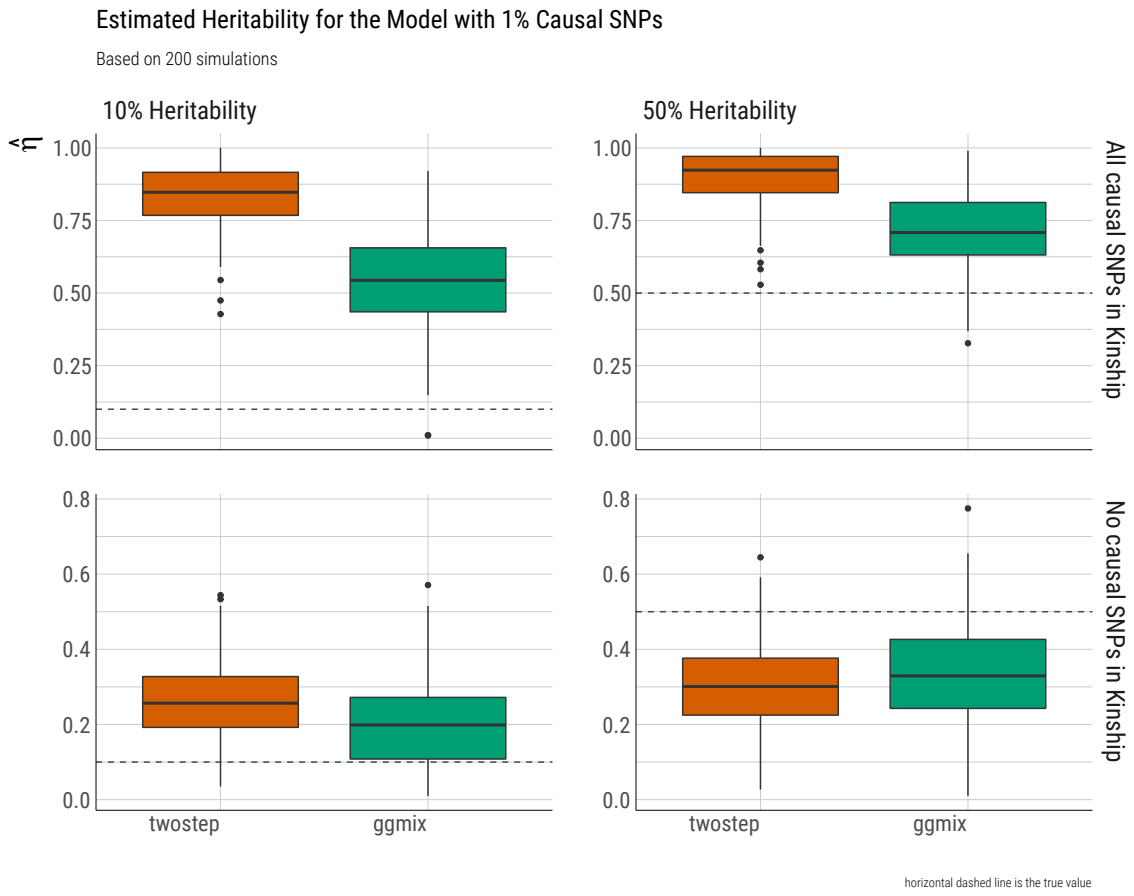


Figure 4.6: Boxplots of the heritability estimate $\hat{\eta}$ from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ and number of causal SNPs that were included in the calculation of the kinship matrix for the model with 1% causal SNPs ($c = 0.01$).

In Figures B.4 (Supplemental Section B.2) and 4.7, we plot the error variance for $c = 0$ and $c = 0.01$, respectively. The twostep and `ggmix` methods correctly estimate the error variance while the lasso overestimates it for the null model and for when 1% of the causal SNPs are in the kinship matrix. We see an inflated estimated error variance across all three methods when $c = 0.01$ and none of the causal SNPs are in the kinship matrix with the lasso and `ggmix` performing similarly.

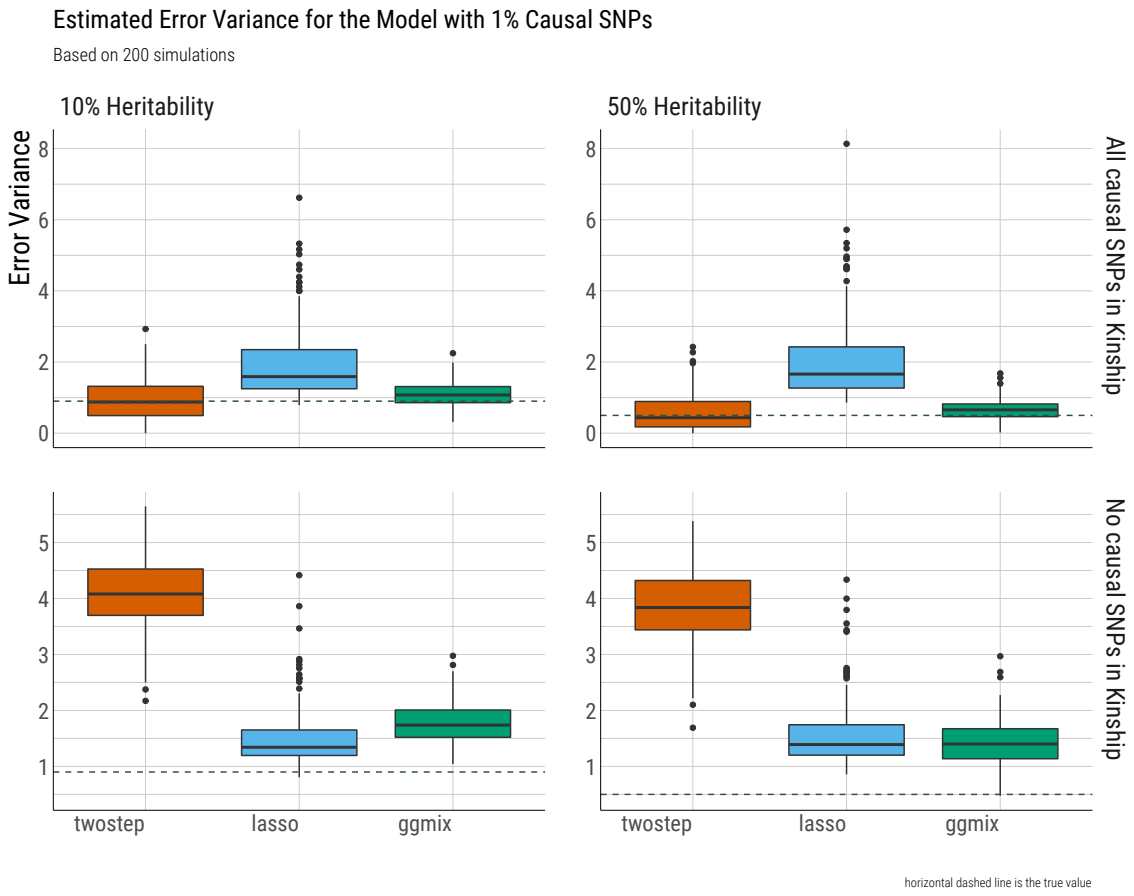


Figure 4.7: Boxplots of the estimated error variance from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ and number of causal SNPs that were included in the calculation of the kinship matrix for the model with 1% causal SNPs ($c = 0.01$).

We compare the model error as a function of $\widehat{S}_{\hat{\lambda}}$ in Figures B.5 (Supplemental Section B.2) and 4.8 for $c = 0$ and $c = 0.01$, respectively. Lasso achieves the smallest model error across all scenarios (for $c = 0.01$), albeit with a large number of active variables. `ggmix` has a smaller model error compared to twostep when all causal SNPs are in the kinship matrix and similar performance when none of the causal SNPs are in the kinship matrix.

Figure 4.8: Means ± 1 standard deviation of the model error vs. the number of active variables by the true heritability $\eta = \{10\%, 50\%\}$ and number of causal SNPs that were included in the calculation of the kinship matrix for the model with 1% causal SNPs ($c = 0.01$).

Overall, we observe that variable selection results for `ggmix` are similar regardless of whether the causal SNPs are in the kinship matrix or not. This result is encouraging since in practice the kinship matrix is constructed from a random sample of SNPs across the genome, some of which are likely to be causal. `ggmix` has very good Type 1 error control, while both the lasso and twostep have a very high false positive rate. Inclusion of the causal SNPs in the kinship calculation has a strong impact on the variance component estimation with the heritabilty and error variance estimates working in opposite directions. That is, when all causal SNPs are in the kinship matrix, the heritability estimates are biased towards 1

while the error variance is correctly estimated. Conversely, when none of the causal SNPs are included in the kinship matrix, the estimated heritability is closer to the true value, while the error variance is inflated. Both the lasso and twostep methods have better signal recovery as compared to `ggmix`. However, this signal is being spread across many variables leading to many Type 1 errors.

4.5 Discussion

We develop a general penalized LMM framework for population structure correction that simultaneously selects and estimates variables, accounting for between individual correlations, in one step. Our CGD algorithm is computationally efficient and has theoretical guarantees of convergence. We provide an easy-to-use software implementation of our algorithm along with a principled method for automatic tuning parameter selection. Through simulation studies, we show that existing approaches such as a two-stage approach or the lasso with a principal component adjustment lead to a large number of false positives. Our proposed method has excellent Type 1 error control and is robust to the inclusion of causal SNPs in the kinship matrix. This feature is important since in practice the kinship matrix is constructed from a random sample of SNPs across the genome, some of which are likely to be causal.

Although we derive a CGD algorithm for the ℓ_1 penalty, our approach can also be easily extended to other penalties such as the elastic net and group lasso with the same guarantees of convergence.

A limitation of `ggmix` is that it first requires computing the covariance matrix with a computation time of $\mathcal{O}(n^2k)$ followed by a spectral decomposition of this matrix in $\mathcal{O}(n^3)$ time where k is the number of SNP genotypes used to construct the covariance matrix. This computation becomes prohibitive for large cohorts such as the UK Biobank (Allen et al.,

2012) which have collected genetic information on half a million individuals. When the matrix of genotypes used to construct the covariance matrix is low rank, there are additional computational speedups that can be implemented. While this has been developed for the univariate case (Lippert et al., 2011), to our knowledge, this has not been explored in the multivariable case. We are currently developing a low rank version of the penalized LMM developed here, which reduces the time complexity from $\mathcal{O}(n^2k)$ to $\mathcal{O}(nk^2)$.

While the predominant motivation for our approach has been association testing, we believe that there are other applications in which it can be used as well. For example, in the most recent Genetic Analysis Workship 20 (GAW20), the causal modeling group investigated causal relationships between DNA methylation (exposure) within some genes and the change in high-density lipoproteins ΔHDL (outcome) using Mendelian randomization (MR) (Davey Smith & Ebrahim, 2003). Penalized regression methods could be used to select SNPs strongly associated with the exposure in order to be used as an instrumental variable (IV). However, since GAW20 data consisted of families, two step methods were used which could have resulted in a large number of false positives. `ggmix` is an alternative approach that could be used for selecting the IV while accounting for the familial structure of the data. Our method is also suitable for fine mapping SNP association signals in genomic regions, where the goal is to pinpoint individual variants most likely to impact the undelying biological mechanisms of disease (Spain & Barrett, 2015).

Chapter 5

An analytic approach for interpretable predictive models in high dimensional data, in the presence of interactions with exposures

Sahir Rai Bhatnagar[1,2], Yi Yang[3], Budhachandra Khundrakpam[4], Alan C Evans[4], Mathieu Blanchette[5], Luigi Bouchard[6], Celia MT Greenwood[1,2]

[1]Department of Epidemiology, Biostatistics and Occupational Health, McGill University

[2]Lady Davis Institute, Jewish General Hospital, Montréal, QC

[3]Department of Mathematics and Statistics, McGill University

[4]Montreal Neurological Institute, McGill University

[5]Department of Computer Science, McGill University

[6]Department of Biochemistry, Université de Sherbrooke

Paper published in *Genetic Epidemiology* (2018). Apr 1;42(3):233-49. DOI 10.1101/102475.

Abstract

Predicting a phenotype and understanding which variables improve that prediction are two very challenging and overlapping problems in analysis of high-dimensional data such as those arising from genomic and brain imaging studies. It is often believed that the number of truly important predictors is small relative to the total number of variables, making computational approaches to variable selection and dimension reduction extremely important. To reduce dimensionality, commonly-used two-step methods first cluster the data in some way, and build models using cluster summaries to predict the phenotype.

It is known that important exposure variables can alter correlation patterns between clusters of high-dimensional variables, i.e., alter network properties of the variables. However, it is not well understood whether such altered clustering is informative in prediction. Here, assuming there is a binary exposure with such network-altering effects, we explore whether use of exposure-dependent clustering relationships in dimension reduction can improve predictive modelling in a two-step framework. Hence, we propose a modelling framework called ECLUST to test this hypothesis, and evaluate its performance through extensive simulations.

With ECLUST, we found improved prediction and variable selection performance compared to methods that do not consider the environment in the clustering step, or to methods that use the original data as features. We further illustrate this modelling framework through the analysis of three data sets from very different fields, each with high dimensional data, a binary exposure, and a phenotype of interest. Our method is available in the *eclust* CRAN package.

5.1 Introduction

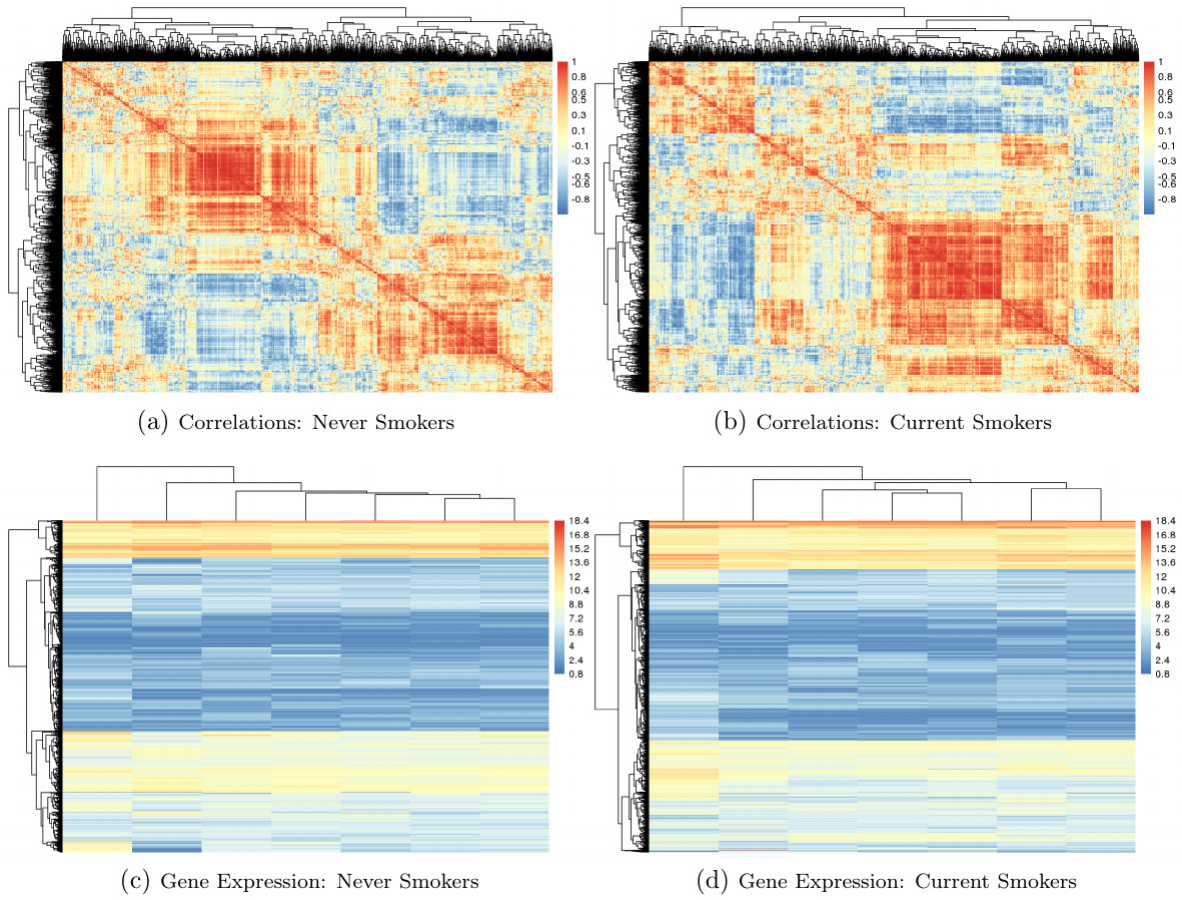
In this article, we consider the prediction of an outcome variable y observed on n individuals from p variables, where p is much larger than n. Challenges in this high-dimensional context include not only building a good predictor which will perform well in an independent dataset, but also being able to interpret the factors that contribute to the predictions. This latter issue can be very challenging in ultra-high dimensional predictor sets. For example, multiple different sets of covariates may provide equivalent measures of goodness of fit (J. Fan et al., 2014), and therefore how does one decide which are important? If many variables are highly correlated, interpretation may be improved by acknowledging the existence of an underlying or latent factor generating these patterns. In consequence, many authors have suggested a two-step procedure where the first step is to cluster or group variables in the design matrix in an interpretable way, and then to perform model fitting in the second step using a summary measure of each group of variables.

There are several advantages to these two-step methods. Through the reduction of the dimension of the model, the results are often more stable with smaller prediction variance, and through identification of sets of correlated variables, the resulting clusters can provide an easier route to interpretation. From a practical point of view, two-step approaches are both flexible and easy to implement because efficient algorithms exist for both clustering (e.g. (Müllner, 2013)) and model fitting (e.g. (J. Friedman et al., 2010; Kuhn, 2008; Y. Yang & Zou, 2014)), particularly in the case when the outcome variable is continuous.

This two-step idea dates back to 1957 when Kendall first proposed using principal components in regression (Kendall, 1957). Hierarchical clustering based on the correlation of the design matrix has also been used to create groups of genes in microarray studies. For example, at each level of a hierarchy, cluster averages have been used as new sets of potential predictors in both forward-backward selection (Hastie et al., 2001) or the lasso (Park et al., 2007). Bühlmann *et al.* proposed a bottom-up agglomerative clustering algorithm based on

canonical correlations and used the group lasso on the derived clusters (Bühlmann et al., 2013). A more recent proposal performs sparse regression on cluster prototypes (Reid & Tibshirani, 2016), i.e., extracting the most representative gene in a cluster instead of averaging them.

These two-step approaches usually group variables based on a matrix of correlations or some transformation of the correlations. However, when there are external factors, such as exposures, that can alter correlation patterns, a dimension reduction step that ignores this information may be suboptimal. Many of the high-dimensional genomic data sets currently being generated capture a possibly dynamic view of how a tissue is functioning, and demonstrate differential patterns of coregulation or correlation under different conditions. We illustrate this critical point with an example of a microarray gene expression dataset available in the *COPDSexualDimorphism.data* package (Sathirapongsasuti, 2013) from Bioconductor. This study measured gene expression in Chronic Obstructive Pulmonary Disease (COPD) patients and controls in addition to their age, gender and smoking status. To see if there was any effect of smoking status on gene expression, we plotted the expression profiles separately for current and never smokers. To balance the covariate profiles, we matched subjects from each group on age, gender and COPD case status, resulting in a sample size of 7 in each group. Heatmaps in Figure 5.1 show gene expression levels and the corresponding gene-gene correlation matrices as a function of dichotomized smoking status for 2,900 genes with large variability. Evidently, there are substantial differences in correlation patterns between the smoking groups (Figures 1a and 1b). However, it is difficult to discern any patterns or major differences between the groups when examining the gene expression levels directly (Figures 1c and 1d). This example highlights two key points; 1) environmental exposures can have a widespread effect on regulatory networks and 2) this effect may be more easily discerned by looking at a measure for gene similarity, relative to analyzing raw expression data.



(a) Correlations: Never Smokers

(b) Correlations: Current Smokers

(c) Gene Expression: Never Smokers

(d) Gene Expression: Current Smokers

Figure 5.1: Heatmaps of correlations between genes (top) and gene expression data (bottom - rows are genes and columns are subjects), stratified by smoking status from a microarray study of COPD (Sathirapongsasuti, 2013). The 20% most variable genes are displayed (2,900 genes). There are 7 subjects in each group, matched on COPD case status, gender and age. Data available on Bioconductor in the `COPDSexualDimorphism.data` package.

Many other examples of altered co-regulation and phenotype associations can be found. For instance, in a pediatric brain development study, very different correlation patterns of cortical thickness within brain regions were observed across age groups, consistent with a process of fine-tuning an immature brain system into a mature one (Khundrakpam et al., 2013). A comparison of gene expression levels in bone marrow from 327 children with acute leukemia found several differentially *coexpressed* genes in philadelphia positive leukemias compared to the cytogenetically normal group (Kostka & Spang, 2004). To give a third example, an analysis of RNA-sequencing data from The Cancer Genome Atlas (TCGA) revealed very

different correlation patterns among sets of genes in tumors grouped according to their missense or null mutations in the TP53 tumor suppressor gene (Klein et al., 2016).

Therefore, in this paper, we pose the question whether clustering or dimension reduction that incorporates known covariate or exposure information can improve prediction models in high dimensional genomic data settings. Substantial evidence of dysregulation of genomic coregulation has been observed in a variety of contexts, however we are not aware of any work that carefully examines how this might impact the performance of prediction models. We propose a conceptual analytic strategy called ECLUST, for prediction of a continuous or binary outcome in high dimensional contexts while exploiting exposure-sensitive data clusters. We restrict our attention to two-step algorithms in order to implement a covariate-driven clustering.

Specifically, we hypothesize that within two-step methods, variable grouping that considers exposure information can lead to improved predictive accuracy and interpretability. We use simulations to compare our proposed method to comparable approaches that combine data reduction with predictive modelling. We are focusing our attention primarily on the performance of alternative dimension reduction strategies within the first step of a two-step method. Therefore, performance of each strategy is compared for several appropriate step 2 predictive models. We then illustrate these concepts more concretely by analyzing three data sets. Our method and the functions used to conduct the simulation studies have been implemented in the R package `eclust` (Bhatnagar, 2017), available on CRAN. Extensive documentation of the package is available at `http://sahirbhatnagar.com/eclust/`.

5.2 Methods

Assume there is a single binary environmental factor E of importance, and an $n \times p$ high dimensional (HD) data set $\mathbf{X}$ (n observations, p features) of relevance. This could be genome-

wide epigenetic data, gene expression data, or brain imaging data, for example. Assume there is a continuous or binary phenotype of interest Y and that the environment has a widespread effect on the HD data, i.e., affects many elements of the HD data. The primary goal is to improve prediction of Y by identifying interactions between E and $\mathbf{X}$ through a carefully constructed data reduction strategy that exploits E dependent correlation patterns. The secondary goal is to improve identification of the elements of $\mathbf{X}$ that are involved; we denote this subset by S_0. We hypothesize that a systems-based perspective will be informative when exploring the factors that are associated with a phenotype of interest, and in particular we hypothesize that incorporation of environmental factors into predictive models in a way that retains a high dimensional perspective will improve results and interpretation.

5.2.1 Potential impacts of covariate-dependent coregulation

Motivated by real world examples of differential coexpression, we first demonstrate that environment-dependent correlations in $\boldsymbol{x}$ can induce an interaction model. Without loss of generality, let $p = 2$ and the relationship between X_1 and X_2 depend on the environment such that

$$X_{i2} = \psi X_{i1} E_i + \varepsilon_i \tag{5.1}$$

where ε_i is an error term and ψ is a slope parameter, that is:

$$X_{i2} = \begin{cases} \psi X_{i1} + \varepsilon_i & \text{when } E_i = 1 \\ \varepsilon_i & \text{when } E_i = 0 \end{cases}$$

Consider the 3-predictor regression model

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \beta_3 E_i + \varepsilon_i^* \tag{5.2}$$

where ε_i^* is another error term which is independent of ε_i. At first glance (5.2) does not contain any interaction terms. However, substituting (5.1) for X_{i2} in (5.2) we get

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 \psi \left(X_{i1} E_i\right) + \beta_3 E_i + \varepsilon_i \beta_2 + \varepsilon_i^* \tag{5.3}$$

The third term in (5.3) resembles an interaction model, with $\beta_2 \psi$ being the interaction parameter. We present a second illustration showing how non-linearity can induce interactions. Suppose

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \beta_3 E_i + \beta_3 \max_{j \in \{1,2\}} \left(X_{ij} - \bar{X}_i\right)^2 + \varepsilon_i^* \tag{5.4}$$

Substituting (5.1) for X_{i2} in (5.4) we obtain a non-linear interaction term. Equation (5.4) provided partial motivation for the model used in our third simulation scenario. Some motivation for this model and a graphical representation are presented below in the Simulation Studies section.

5.2.2 Proposed framework and algorithm

We restrict attention to methods containing two phases as illustrated in Figure 5.2: 1a) a clustering stage where variables are clustered based on some measure of similarity, 1b) a dimension reduction stage where a summary measure is created for each of the clusters, and 2) a simultaneous variable selection and regression stage on the summarized cluster measures. Although this framework appears very similar to any two-step approach, our hypothesis is that allowing the clustering in Step 1a to depend on the environment variable can lead to improvements in prediction after Step 2. Hence, methods in Step 1a are adapted to this end, as decribed in the following sections. Our focus in this manuscript is on the clustering and cluster representation steps. Therefore, we compare several well known methods for variable selection and regression that are best adapted to our simulation designs and data sets.

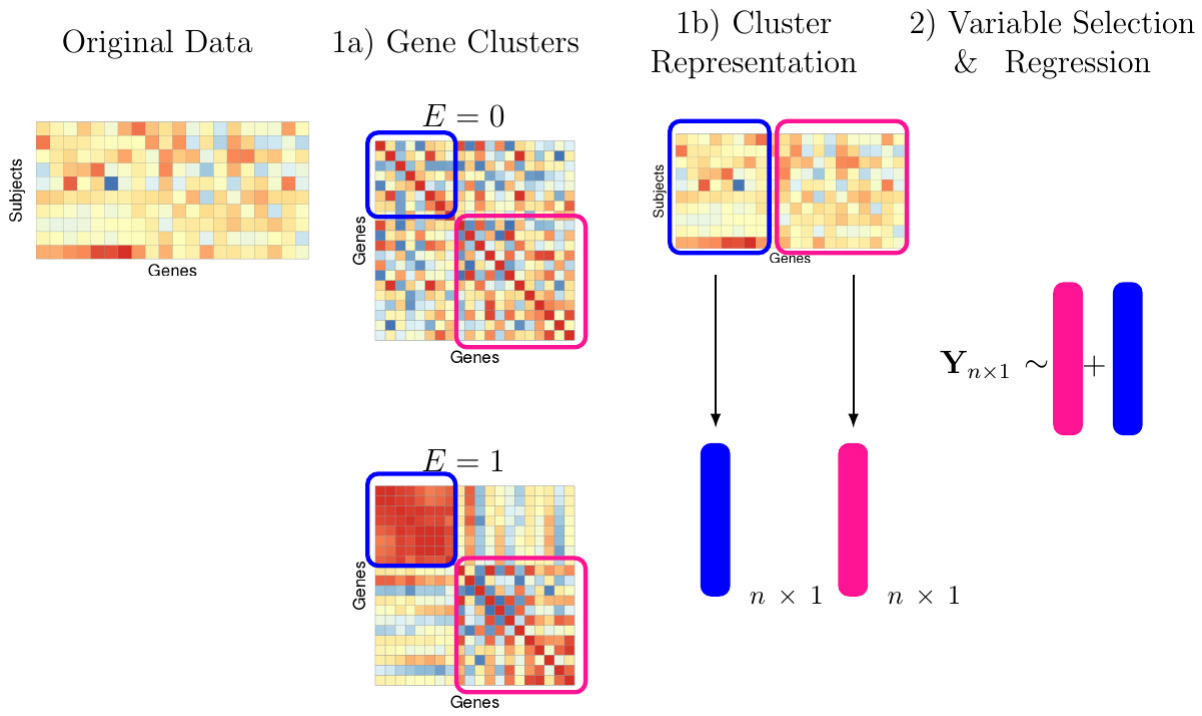


Figure 5.2: Overview of our proposed method. 1a) A measure of similarity is calculated separately for both groups and clustering is performed on a linear combination of these two matrices. 1b) We reduce the dimension of each cluster by taking a summary measure. 2) Variable selection and regression is performed on the cluster representatives, E and their interaction with E.

Step 1a: Clustering using co-expression networks that are influenced by the environment

In agglomerative clustering, a measure of similarity between sets of observations is required in order to decide which clusters should be combined. Common choices include Euclidean, maximum and absolute distance. A more natural choice in genomic or brain imaging data is to use Pearson correlation (or its absolute value) because the derived clusters are biologically interpretable. Indeed, genes that cluster together are correlated and thus likely to be involved in the same cellular process. Similarly, cortical thickness measures of the brain tend to be correlated within pre-defined regions such as the left and right hemisphere, or frontal and temporal regions (Sato et al., 2013). However, the information on the con-

nection between two variables, as measured by the Pearson correlation for example, may be noisy or incomplete. Thus it is of interest to consider alternative measures of pairwise interconnectedness. Gene co-expression networks are being used to explore the system-level function of genes, where nodes represent genes and are connected if they are significantly co-expressed (B. Zhang & Horvath, 2005), and here we use their overlap measure (Ravasz et al., 2002) to capture connectnedness between two X variables within each environmental condition. As was discussed earlier, genes can exhibit very different patterns of correlation in one environment versus the other (e.g. Figure 5.1). Furthermore, measures of similarity that go beyond pairwise correlations and consider the shared connectedness between nodes can be useful in elucidating networks that are biologically meaningful. Therefore, we propose to first look at the topological overlap matrix (TOM) separately for exposed ($E = 1$) and unexposed ($E = 0$) individuals (see Supplemental Section C.1 for details on the TOM). We then seek to identify nodes that are very different between environments. We determine differential coexpression using the absolute difference $TOM(X_{\text{diff}}) = |TOM_{E=1} - TOM_{E=0}|$ (Klein et al., 2016). We then use hierarchical clustering with average linkage on the derived difference matrix to identify these differentially co-expressed variables. Clusters are automatically chosen using the `dynamicTreeCut` (Langfelder et al., 2008) algorithm. Of course, there could be other clusters which are not sensitive to the environment. For this reason we also create a set of clusters based on the TOM for all subjects denoted $TOM(X_{\text{all}})$. This will lead to each covariate appearing in two clusters. In the sequel we denote the clusters derived from $TOM(X_{\text{all}})$ as the set $C_{\text{all}} = \{C_1, \ldots, C_k\}$, and those derived from $TOM(X_{\text{diff}})$ as the set $C_{\text{diff}} = \{C_{k+1}, \ldots, C_\ell\}$ where $k < \ell < p$.

Step 1b: Dimension reduction via cluster representative

Once the clusters have been identified in phase 1, we proceed to reduce the dimensionality of the overall problem by creating a summary measure for each cluster. A low-dimensional structure, i.e. grouping when captured in a regression model, improves predictive perfor-

mance and facilitates a model's interpretability. We propose to summarize a cluster by a single representative number. Specifically, we chose the average values across all measures (Bühlmann et al., 2013; Park et al., 2007), and the first principal component (Langfelder & Horvath, 2007). These representative measures are indexed by their cluster, i.e., the variables to be used in our predictive models are $\widetilde{\mathbf{X}}_{\text{all}} = \left\{\widetilde{X}_{C_1}, \ldots, \widetilde{X}_{C_k}\right\}$ for clusters that do not consider E, as well as $\widetilde{\mathbf{X}}_{\text{diff}} = \left\{\widetilde{X}_{C_{k+1}}, \ldots, \widetilde{X}_{C_\ell}\right\}$ for E-derived clusters. The tilde notation on the X is to emphasize that these variables are different from the separate variables in the original data.

Step 2: Variable Selection and Regression

Because the clustering in phase 1 is unsupervised, it is possible that the derived latent representations from phase 2 will not be associated with the response. We therefore use penalized methods for supervised variable selection, including the lasso (Tibshirani, 1996) and elasticnet (Zou & Hastie, 2005) for linear models, and multivariate adaptive regression splines (MARS) (J. H. Friedman, 1991) for nonlinear models. We argue that the selected non-zero predictors in this model will represent clusters of genes that interact with the environment and are associated with the phenotype. Such an additive model might be insufficient for predicting the outcome. In this case we may directly include the environment variable, the summary measures and their interaction. In the light of our goals to improve prediction and interpretability, we consider the following model

$$g(\boldsymbol{\mu}) = \beta_0 + \sum_{j=1}^{\ell} \beta_j \widetilde{X}_{C_j} + \beta_E E + \sum_{j=1}^{\ell} \alpha_j \left(\widetilde{X}_{C_j} E\right) + \varepsilon \tag{5.5}$$

where $g(\cdot)$ is a known link function, $\boldsymbol{\mu} = \mathsf{E}[Y|\mathbf{X}, E, \boldsymbol{\beta}, \boldsymbol{\alpha}]$ and $\widetilde{X}_{C_j}$ are linear combinations of $\mathbf{X}$ (from Step 1b). The primary comparison is models with $\widetilde{\mathbf{X}}_{\text{all}}$ only versus models with $\widetilde{\mathbf{X}}_{\text{all}}$ and $\widetilde{\mathbf{X}}_{\text{diff}}$. Given the context of either the simulation or the data set, we use either linear models or non linear models. Our general approach, ECLUST, can therefore be summarized

by the algorithm in Table 5.1.

Table 5.1: Details of ECLUST algorithm

Step	Description, Software[a] and Reference
1a)	i) Calculate TOM separately for observations with $E = 0$ and $E = 1$ using `WGCNA::TOMsimilarityFromExpr` (Langfelder & Horvath, 2008) ii) Euclidean distance matrix of $\lvert TOM_{E=1} - TOM_{E=0} \rvert$ using `stats::dist` iii) Run the `dynamicTreeCut` algorithm (Langfelder et al., 2008, 2016) on the distance matrix to determine the number of clusters and cluster membership using `dynamicTreeCut::cutreeDynamic` with `minClusterSize = 50`
1b)	i) 1st PC or average for each cluster using `stat::prcomp` or `base::mean` ii) Penalized regression model: create a design matrix of the derived cluster representatives and their interactions with E using `stats::model.matrix` iii) MARS model: create a design matrix of the derived cluster representatives and E
2)	i) For linear models, run penalized regression on design matrix from step 1b) using `glmnet::cv.glmnet` (J. Friedman et al., 2010). Elasticnet mixing parameter `alpha=1` corresponds to the lasso and `alpha=0.5` corresponds to the value we used in our simulations for elasticnet. The tuning parameter `alpha` is selected by minimizing 10 fold cross-validated mean squared error (MSE). ii) For non-linear effects, run MARS on the design matrix from step 1b) using `earth::earth` (Milborrow. Derived from mda:mars by T. Hastie and R. Tibshirani., 2011) with pruning method `pmethod = "backward"` and maximum number of model terms `nk = 1000`. The `degree=1,2` is chosen using 10 fold cross validation (CV), and within each fold the number of terms in the model is the one that minimizes the generalized cross validated (GCV) error.

[a]All functions are implemented in `R` (R Core Team, 2016). The naming convention is as follows: `package_name::package_function`. Default settings used for all functions unless indicated otherwise.

5.3 Simulation Studies

We have evaluated the performance of our ECLUST method in a variety of simulated scenarios. For each simulation scenario we compared ECLUST to the following analytic approaches 1) regression and variable selection is performed on the model which consists of the original variables, E and their interaction with E (SEPARATE), and 2) clustering is performed without considering the environmental exposure followed by regression and variable selection on the cluster representations, E, and their interaction with E (CLUST). A detailed

description of the methods being compared is summarized in Table 5.2. We have designed 6 simulation scenarios that illustrate different kinds of relationships between the variables and the response. For all scenarios, we have created high dimensional data sets with p predictors ($p = 5000$), and sample sizes of $n = 200$. We also assume that we have two data sets for each simulation - a training data set where the parameters are estimated, and a testing data set where prediction performance is evaluated, each of size $n_{train} = n_{test} = 200$. The number of subjects who were exposed ($n_{E=1} = 100$) and unexposed ($n_{E=0} = 100$) and the number of truly associated parameters ($|S_0| = 500$) remain fixed across the 6 simulation scenarios. Let

$$Y = Y^* + k \cdot \varepsilon \tag{5.6}$$

where Y^* is the linear predictor, the error term ε is generated from a standard normal distribution, and k is chosen such that the signal-to-noise ratio $SNR = (Var(Y^*)/Var(\varepsilon))$ is 0.2, 1 and 2 (e.g. the variance of the response variable Y due to ε is $1/SNR$ of the variance of Y due to Y^*).

5.3.1 The Design Matrix

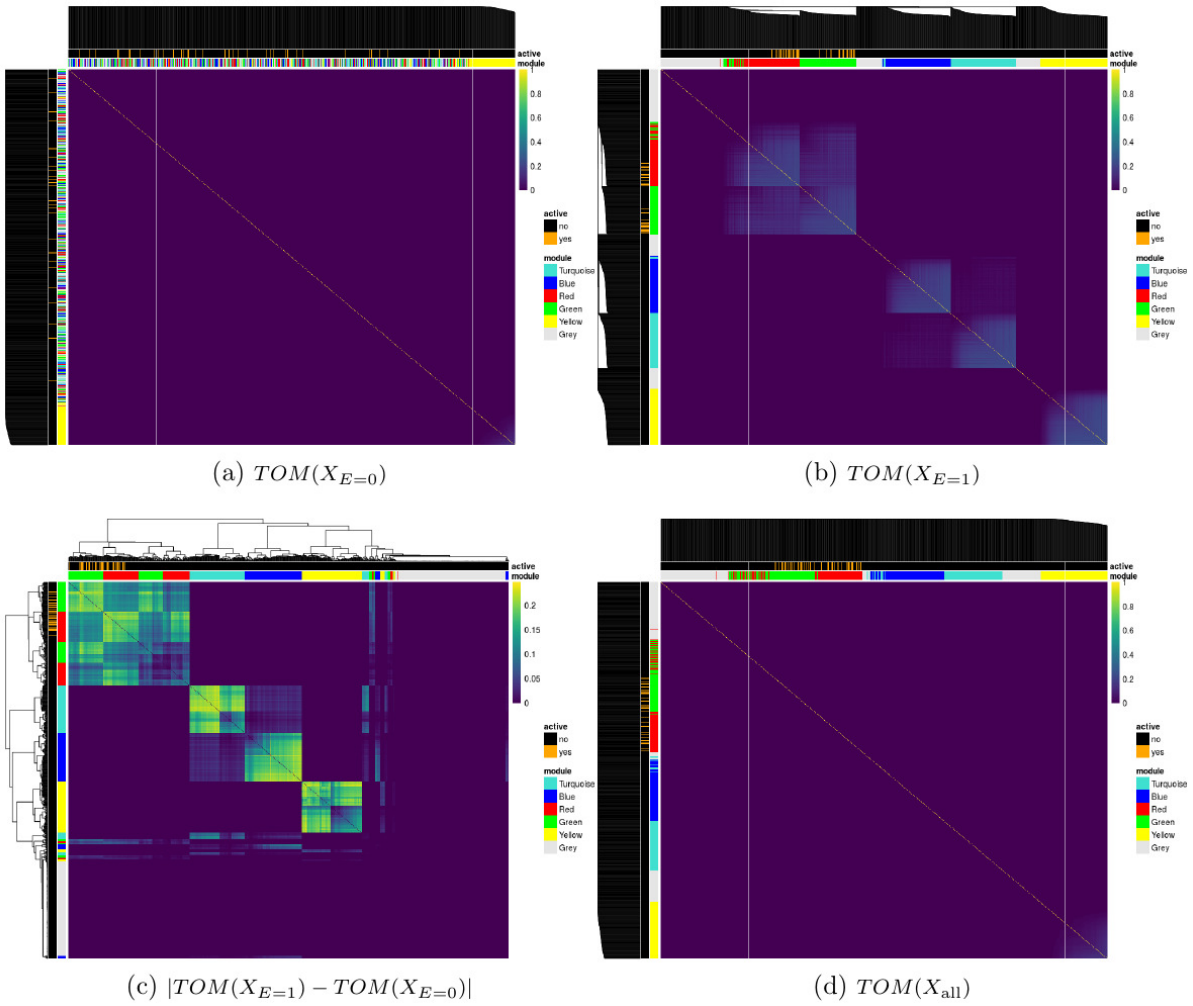
We generated covariate data in blocks using the `simulateDatExpr` function from the `WGCNA` package in `R` (version 1.51). This generates data from a latent vector: first a seed vector is simulated, then covariates are generated with varying degree of correlation with the seed vector in a given block. We simulated five clusters (blocks), each of size 750 variables, and labeled them by colour (turquoise, blue, red, green and yellow), while the remaining 1250 variables were simulated as independent standard normal vectors (grey) (Figure 5.3). For the unexposed observations ($E = 0$), only the predictors in the yellow block were simulated with correlation, while all other covariates were independent within and between blocks. The TOM values are very small for the yellow cluster because it is not correlated with any of its neighbors. For the exposed observations ($E = 1$), all 5 blocks contained predictors

Table 5.2: Summary of methods used in simulation study

General Approach	Summary Measure of Feature Clusters	Description[a,b]
SEPARATE	NA	Regression of the original predictors $\{X_1, \ldots, X_p\}$ on the response i.e. no transformation of the predictors is being done here
CLUST	1st PC, average	Create clusters of predictors without using the environment variable $\{C_1, \ldots, C_k\}$. Use the summary measure of each cluster as inputs of the regression model.
ECLUST	1st PC, average	Create clusters of predictors using the environment variable $\{C_{k+1}, \ldots, C_\ell\}$ where $k < \ell < p$, as well as clusters without the environment variable $\{C_1, \ldots, C_k\}$. Use summary measures of $\{C_1, \ldots, C_\ell\}$ as inputs of the regression model.

[a]Simulations 1 and 2 used lasso and elasticnet for the linear models, and simulation 3 used MARS for estimating non-linear effects [b]Simulations 4, 5 and 6 convert the continuous response generated in simulations 1, 2 and 3, respectively, into a binary response [c]PC: principal component

that are correlated. The blue and turquoise blocks are set to have an average correlation of 0.6. The average correlation was varied for both green and red clusters $\rho = \{0.2, 0.9\}$ and the active set S_0, that are directly associated with y, was distributed evenly between these two blocks. Heatmaps of the TOM for this environment dependent correlation structure are shown in Figure 5.3 with annotations for the true clusters and active variables. This design matrix shows widespread changes in gene networks in the exposed environment, and this subsequently affects the phenotype through the two associated clusters. There are also pathways that respond to changes in the environment but are not associated with the response (blue and turquoise), while others that are neither active in the disease nor affected by the environment (yellow).



(a) $TOM(X_{E=0})$ (b) $TOM(X_{E=1})$

(c) $|TOM(X_{E=1}) - TOM(X_{E=0})|$ (d) $TOM(X_{\text{all}})$

Figure 5.3: Topological overlap matrices (TOM) of simulated predictors based on subjects with (a) $E = 0$, (b) $E = 1$, (c) their absolute difference and (d) all subjects. Dendrograms are from hierarchical clustering (average linkage) of one minus the TOM for a, b, and d and the euclidean distance for c. Some variables in the red and green clusters are associated with the outcome variable. The *module* annotation represents the true cluster membership for each predictor, and the *active* annotation represents the truly associated predictors with the response.

5.3.2 The response

The first three simulation scenarios differ in how the linear predictor Y^* in (5.6) is defined, and also in the choice of regression model used to fit the data. In simulations 1 and 2 we use lasso (Tibshirani, 1996) and elasticnet (Zou & Hastie, 2005) to fit linear models; then we use

MARS (J. H. Friedman, 1991) in simulation 3 to estimate non-linear effects. Simulations 4, 5 and 6 use the GLM version of these models, respectively, since the responses are binary.

Simulation 1

Simulation 1 was designed to evaluate performance when there are no explicit interactions between X and E (see Equation (5.3)). We generated the linear predictor from

$$Y^* = \sum_{\substack{j\in\{1,\ldots,250\} \\ j\in \text{ red, green block}}} \beta_j X_j + \beta_E E \tag{5.7}$$

where $\beta_j \sim \text{Unif}\,[0.9, 1.1]$ and $\beta_E = 2$. That is, only the first 250 predictors of both the red and green blocks are active. In this setting, only the main effects model is being fit to the simulated data.

Simulation 2

In the second scenario we explicitly simulated interactions. All non-zero main effects also had a corresponding non-zero interaction effect with E. We generated the linear predictor from

$$Y^* = \sum_{\substack{j\in\{1,\ldots,125\} \\ j\in \text{ red, green block}}} (\beta_j X_j + \alpha_j X_j E) + \beta_E E \tag{5.8}$$

where $\beta_j \sim \text{Unif}\,[0.9, 1.1]$, $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$ or $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$, and $\beta_E = 2$. In this setting, both the main effects and their interactions with E are being fit to the simulated data.

Simulation 3

In the third simulation we investigated the performance of the ECLUST approach in the presence of non-linear effects of the predictors on the phenotype:

$$Y_i^* = \sum_{\substack{j\in\{1,\ldots,250\} \\ j\in \text{ red, green block}}} \beta_j X_{ij} + \beta_E E_i + \alpha_Q E_i \cdot f(Q_i) \tag{5.9}$$

where

$$Q_i = -\max_{\substack{j\in\{1,\ldots,250\} \\ j\in \text{ red, green block}}} \left(X_{ij} - \bar{X}_i\right)^2 \tag{5.10}$$

$$f(u_i) = \frac{u_i - \min\limits_{i\in\{1,\ldots,n\}} u_i}{-\min\limits_{i\in\{1,\ldots,n\}} u_i} \tag{5.11}$$

$$\bar{X}_i = \frac{1}{500} \sum_{\substack{j\in\{1,\ldots,250\} \\ j\in \text{ red, green block}}} X_{ij}$$

The design of this simulation was partially motivated by considering the idea of canalization, where systems operate within appropriate parameters until sufficient perturbations accumulate (e.g. Gibson (2009)). In this third simulation, we set $\beta_j \sim \text{Unif}\,[0.9, 1.1]$, $\beta_E = 2$ and $\alpha_Q = 1$. We assume the data has been appropriately normalized, and that the correlation between any two features is greater than or equal to 0. In simulation 3, we tried to capture the idea that an exposure could lead to coregulation or disregulation of a cluster of X's, which in itself directly impacts Y. Hence, we defined coregulation as the X's being similar in magnitude and disregulation as the X's being very different. The Q_i term in (5.10) is defined such that higher values would correspond to strong coregulation whereas lower values correspond to disregulation. For example, suppose Q_i ranges from -5 to 0. It will be -5 when there is lots of variability (disregulation) and 0 when there is none (strong coregulation). The function $f(\cdot)$ in (5.11) simply maps Q_i to the [0,1] range. In order to get an idea of the

relationship in (5.9), Figure 5.4 displays the response Y as a function of the first principal component of $\sum_j \beta_j X_{ij}$ (denoted by `1st PC`) and $f(Q_i)$. We see that lower values of $f(Q_i)$ (which implies disregulation of the features) leads to a lower Y. In this setting, although the clusters do not explicitly include interactions between the X variables, the MARS algorithm allows for the possibility of two way interactions between any of the variables.

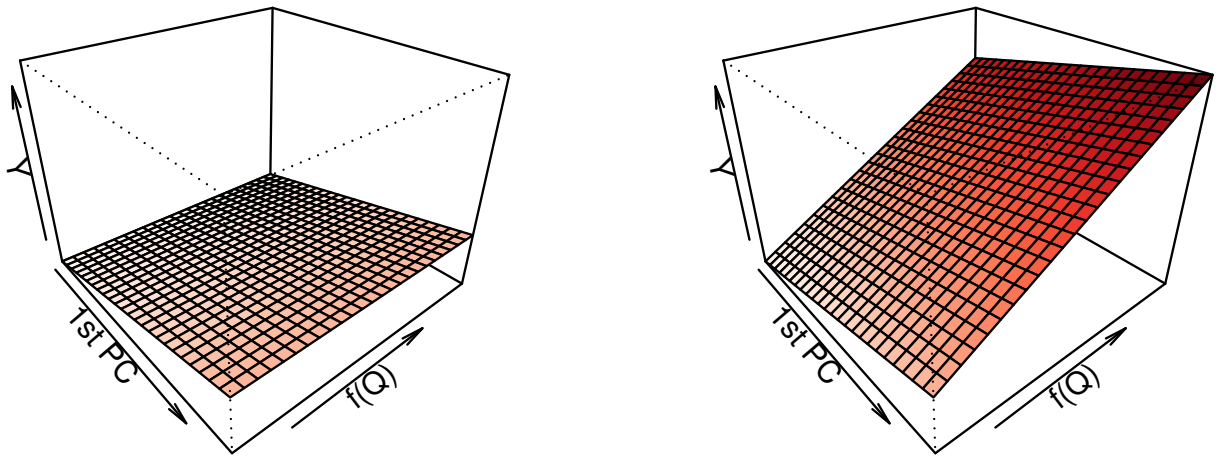


Figure 5.4: Visualization of the relationship between the response, the first principal component of the main effects and $f(Q_i)$ in (5.9) for $E = 0$ (left) and $E = 1$ (right) in simulation scenario 3. This graphic also depicts the intuition behind model (5.4).

Simulations 4, 5 and 6

We used the same simulation setup as above, except that we took the continuous outcome Y, defined $p = 1/(1 + exp(-Y))$ and used this to generate a two-class outcome z with $\Pr(z = 1) = p$ and $\Pr(z = 0) = 1 - p$. The true parameters were simulated as $\beta_j \sim$ Unif$[\log(0.9), \log(1.1)]$, $\beta_E = \log(2)$, $\alpha_j \sim$ Unif$[\log(0.4), \log(0.6)]$ or $\alpha_j \sim$ Unif$[\log(1.9), \log(2.1)]$. Simulations 4, 5 and 6 are the binary response versions of simulations 1, 2 and 3, respectively. The larger odds ratio for E compared to the odds ratio for X is motivated by certain environmental factors which are well known to have substantial impacts on disease risks and phenotypes. For example, body mass index has been estimated to explain a large proportion of variation in bone mineral density (BMD) in women (10-20%) (Felson et al., 1993). This

can be converted to a slope of 0.31-0.44 assuming variables are standardized, i.e., changes of 0.3-0.4 standard deviations in BMD per standard deviation change in weight. In contrast, the majority of SNPs and rare variants have effect sizes under 0.10 standard deviations on BMD (Kemp et al., 2017).

5.3.3 Measures of Performance

Simulation performance was assessed with measures of model fit, prediction accuracy and feature stability. Several measures for each of these categories, and the specific formulae used are provided in Table 5.3. We simulated both a training data set and a test data set for each simulation: all tuning parameters for model selection were selected using the training sets only. Although most of the measures of model fit were calculated on the test data sets, true positive rate, false positive rate and correct sparsity were calculated on the training set only. The root mean squared error is determined by predicting the response for the test set using the fitted model on the training set. The area under the curve is determined using the trapezoidal rule (Robin et al., 2011). The stability of feature importance is defined as the variability of feature weights under perturbations of the training set, i.e., small modifications in the training set should not lead to considerable changes in the set of important covariates (Tološi & Lengauer, 2011). A feature selection algorithm produces a weight (e.g. $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$), a ranking (e.g. $rank(\boldsymbol{\beta}) : \mathbf{r} = (r_1, \ldots, r_m)$) and a subset of features (e.g. $\mathbf{s} = (s_1, \ldots, s_p), s_j = \mathbb{I}\{\beta_j \neq 0\}$ where $\mathbb{I}\{\cdot\}$ is the indicator function). In the CLUST and ECLUST methods, we defined a predictor to be non-zero if its corresponding cluster representative weight was non-zero. Using 10-fold cross validation (CV), we evaluated the similarity between two features and their rankings using Pearson and Spearman correlation, respectively. For each CV fold we re-ran the models and took the average Pearson/Spearman correlations of the $\binom{10}{2}$ combinations of estimated coefficients vectors. To measure the similarity between two subsets of features we took the average of the

Jaccard distance in each fold. A Jaccard distance of 1 indicates perfect agreement between two sets while no agreement will result in a distance of 0. For MARS models we do not report the Pearson/Spearman stability rankings due to the adaptive and functional nature of the model (there are many possible combinations of predictors, each of which are linear basis functions).

Table 5.3: Measures of Performance

Measure	Formula
Model Fit	
True Positive Rate (TPR)	$\lvert\widehat{S} \in S_0\rvert / \lvert S_0\rvert$
False Positive Rate (TPR)	$\lvert\widehat{S} \notin S_0\rvert / \lvert j \notin S_0\rvert$
Correct Sparsity (Witten et al., 2014)	$\frac{1}{p}\sum_{j=1}^{p} A_j$ $A_j = \begin{cases} 1 & \text{if } \hat{\beta}_j = \beta_j = 0 \\ 1 & \text{if } \hat{\beta}_j \neq 0, \beta_j \neq 0 \\ 0 & \text{if else} \end{cases}$
Prediction Accuracy	
Root Mean Squared Error (RMSE)	$\lVert \mathbf{Y}_{test} - \widehat{\boldsymbol{\mu}}(\mathbf{X}_{test}) \rVert_2$
Area Under the Curve (AUC)	Trapezoidal rule
Hosmer-Lemeshow Test ($G = 10$)	χ^2 test statistic
Feature Stability using K-fold Cross-Validation on training set (Kalousis et al., 2007)	
Pearson Correlation (ρ) (Pearson, 1895)	$\binom{K}{2}^{-1} \sum_{i,j \in \{1,\ldots,K\}, i \neq j} \frac{cov\left(\widehat{\beta}_{(i)}, \widehat{\beta}_{(j)}\right)}{\sigma_{\widehat{\beta}_{(i)}} \sigma_{\widehat{\beta}_{(j)}}}$
Spearman Correlation (r) (Spearman, 1904)	$\binom{K}{2}^{-1} \sum_{i,j \in \{1,\ldots,K\}, i \neq j} \left[1 - 6 \sum_{m} \frac{\left(r_{m_{(i)}} - r_{m_{(j)}}\right)^2}{p(p^2-1)} \right]$
Jaccard Distance (Jaccard, 1912)	$\frac{\lvert\widehat{S}_{(i)} \cap \widehat{S}_{(j)}\rvert}{\lvert\widehat{S}_{(i)} \cup \widehat{S}_{(j)}\rvert}$

[a] $\widehat{\boldsymbol{\mu}}$: fitting procedure on the training set [b] S_0: index of active set $= \{j; \beta_j^0 \neq 0\}$ [c] $\widehat{S}$: index of the set of non-zero estimated coefficients $= \left\{j; \widehat{\beta}_j \neq 0\right\}$ [d] $|A|$: is the cardinality of set A

5.3.4 Results

All reported results are based on 200 simulation runs. We graphically summarized the results across simulations 1-3 for model fit (Figure 5.5) and feature stability (Figure 5.6). The results for simulations 4-6 are shown in the Supplemental Section C.2, Figures A.1-A.6. We restrict our attention to $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. The model names are labeled as `summary measure_model` (e.g. `avg_lasso` corresponds using the average of the features in a cluster as inputs into a lasso regression model). When there is no summary measure appearing in the model name, that indicates that the original variables were used (e.g. `enet` means all separate features were used in the elasticnet model). In panel A of Figure 5.5, we plot the true positive rate against the false positive rate for each of the 200 simulations. We see that across all simulation scenarios, the SEPARATE method has extremely poor sensitivity compared to both CLUST and ECLUST, which do much better at identifying the active variables, though the resulting models are not always sparse. The relatively few number of green points in panel A is due to the small number of estimated clusters (Supplemental Section C.3, Figure A.7) leading to very little variability in performance across simulations. The better performance of ECLUST over CLUST is noticeable as more points lie in the top left part of the plot. The horizontal banding in panel A reflects the stability of the TOM-based clustering approach. ECLUST also does better than CLUST in correctly determining whether a feature is zero or nonzero (Figure 5.5, panel B). Importantly, across all three simulation scenarios, ECLUST outperforms the competing methods in terms of RMSE (Figure 5.5, panel C), regardless of the summary measure and modeling procedure. We present the distribution for the effective number of variables selected in the supplemental material (Figures A.8 and A.9). We see that the median number of variables selected from ECLUST is less than the median number of variables selected from CLUST, though ECLUST has more variability.

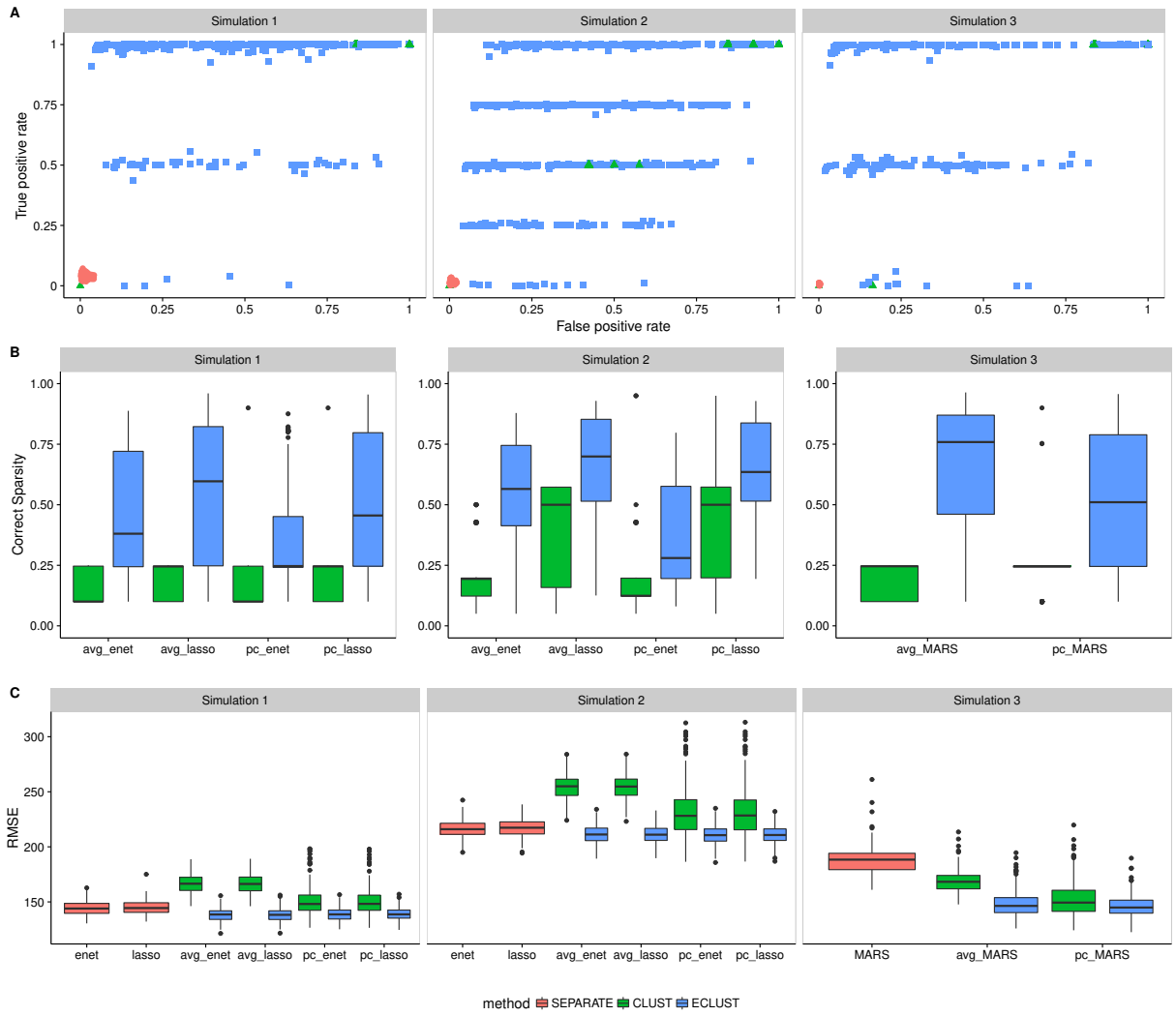


Figure 5.5: Model fit results from simulations 1, 2 and 3 with $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

While the approach using all separate original variables (SEPARATE) produce sparse models, they are sensitive to small perturbations of the data across all stability measures (Figure 5.6), i.e, similar datasets produce very different models. Although the median for the CLUST approach is always slightly better than the median for ECLUST across all stability measures, CLUST results can be much more variable, particularly when stability is measured by the agreement between the value and the ranking of the estimated coefficients across CV folds (Figure 5.6, panels B and C). The number of estimated clusters, and therefore the number of features in the regression model, tends to be much smaller in CLUST compared to

ECLUST, and this explains its poorer performance using the stability measures in Figure 5.6, since there are more coefficients to estimate. Overall, we observe that the relative performance of ECLUST versus CLUST in terms of stability is consistent across the two summary measures (average or principal component) and across the penalization procedures. The complete results for different values of ρ, SNR and α_j (when applicable) are available in the Supplemental Section C.4, Figures A.10-A.15 for Simulation 1, Figures A.16 - A.21 for Simulation 2, and Figures A.22 - A.25 for Simulation 3. They show that these conclusions are not sensitive to the SNR, ρ or α_j. Similar conclusions are made for a binary outcome using logistic regression versions of the lasso, elasticnet and MARS. ECLUST and CLUST also have better calibration than the SEPARATE method for both linear and non-linear models (Supplemental Section C.2, Figures A.3-A.6). The distributions of Hosmer-Lemeshow (HL) p-values do not follow uniformity. This is in part due to the fact that the HL test has low power in the presence of continuous-dichotomous variable interactions (Hosmer et al., 1997). Upon inspection of the Q-Q plots, we see that the models have difficulty predicting risks at the boundaries which is a known issue in most models. We also have a small sample size of 200, which means there are on average only 20 subjects in each of the 10 bins. Furthermore, the HL test is sensitive to the choice of bins and method of computing quantiles. Nevertheless, the improved fit relative to the SEPARATE analysis is quite clear.

We also ran all our simulations using the Pearson correlation matrix as a measure of similarity in order to compare its performance against the TOM. The complete results are in the Supplemental Section C.5, Figures A.26-A.31 for Simulation 1, Figures A.32 - A.37 for Simulation 2, and Figures A.38 - A.41 for Simulation 3. In general, we see slightly better performance of CLUST over ECLUST when using Pearson correlations. This result is probably due to the imprecision in the estimated correlations. The exposure dependent similarity matrices are quite noisy, and the variability is even larger when we examine the differences between two correlation matrices. Such large levels of variability have a negative impact on the clustering algorithm's ability to detecting the true clusters.

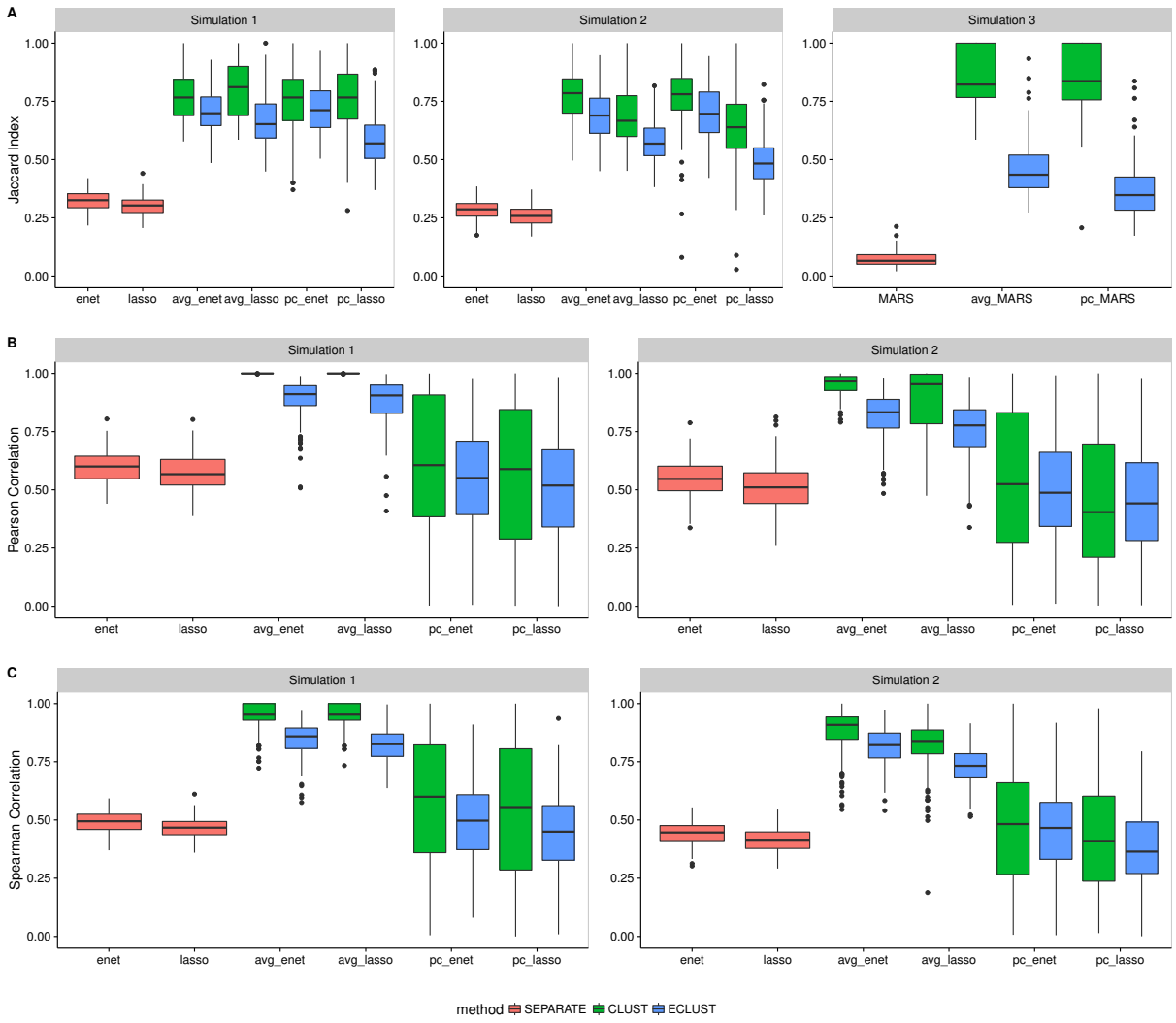


Figure 5.6: Stability results from simulations 1, 2 and 3 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

5.4 Analysis of three data sets

In this section we demonstrate the performance of ECLUST on three high dimensional datasets with contrasting motivations and features. In the first data set, normal brain development is examined in conjunction with intelligence scores. In the second data set we aim to identify molecular subtypes of ovarian cancer using gene expression data. The investigators' goal in the third data set is to examine the impact of gestational diabetes mellitus (GDM)

on childhood obesity in a sample of mother-child pairs from a prospective birth cohort. The datasets comprise a range of sample sizes, and both the amount of clustering in the HD data and the strength of the effects of the designated exposure variables vary substantially. Due to the complex nature of these datasets, we decided to use MARS models for step 2 of our algorithm for all 3 datasets, as outlined in Table 5.1. In order to assess performance in these data sets, we have computed the 0.632 estimator (Efron, 1983) and the 95% confidence interval of the R^2 and RMSE from 100 bootstrap samples. The R^2 reported here is defined as the squared Pearson correlation coefficient between the observed and predicted response (Kvålseth, 1985), and the RMSE is defined as in Table 5.3. Because MARS models can result in unstable predictors (Kuhn, 2008), we also report the results of bagged MARS from $B = 50$ bootstrap samples, where bagging (Breiman, 1996) refers to averaging the predictions from each of the MARS models fit on the B bootstrap samples.

5.4.1 NIH MRI Study of Normal Brain Development

The NIH MRI Study of Normal Brain Development, started in 2001, was a 7 year longitudinal multi-site project that used magnetic resonance technologies to characterize brain maturation in 433 medically healthy, psychiatrically normal children aged 4.5-18 years (Evans et al., 2006). The goal of this study was to provide researchers with a representative and reliable source of healthy control subject data as a basis for understanding atypical brain development associated with a variety of developmental, neurological, and neuropsychiatric disorders affecting children and adults. Brain imaging data (e.g. cortical surface thickness, intra-cranial volume), behavioural measures (e.g. IQ scores, psychiatric interviews, behavioral ratings) and demographics (e.g. socioeconomic status) were collected at two year intervals for three time points and are publicly available upon request. Previous research using these data found that level of intelligence and age correlate with cortical thickness (Khundrakpam et al., 2013; Shaw et al., 2006), but to our knowledge no such relation between income and

cortical thickness has been observed. We therefore used this data to see the performance of ECLUST in the presence (age) and absence (income) of an effect on the correlations in the HD data. We analyzed the 10,000 most variable regions on the cortical surface from brain scans corresponding to the first sampled time point only. We used binary age (166 age ≤ 11.3 and 172 > 11.3) and binary income (142 high and 133 low income) indicator as the environment variables and standardized IQ scores as the response. We identified 22 clusters from $TOM(X_{\text{all}})$ and 57 clusters from $TOM(X_{\text{diff}})$ when using age as the environment, and 86 clusters from $TOM(X_{\text{all}})$ and 49 clusters from $TOM(X_{\text{diff}})$ when using income as the environment. Results are shown in Figure 5.7, panels C and D. The method which uses all individual variables as predictors (pink), has better R^2 but also worse RMSE compared to CLUST and ECLUST, likely due to over-fitting. There is a slight benefit in performance for ECLUST over CLUST when using age as the environment (panel D). Importantly, we observe very similar performance between CLUST and ECLUST across all models (panel C), suggesting very little impact on the prediction performance when including features derived both with and without the E variable, in a situation where they are unlikely to be relevant.

5.4.2 Gene Expression Study of Ovarian Cancer

Differences in gene expression profiles have led to the identification of robust molecular subtypes of ovarian cancer; these are of biological and clinical importance because they have been shown to correlate with overall survival (Tothill et al., 2008). Improving prediction of survival time based on gene expression signatures can lead to targeted therapeutic interventions (Helland et al., 2011). The proposed ECLUST algorithm was applied to gene expression data from 511 ovarian cancer patients profiled by the Affymetrix Human Genome U133A 2.0 Array. The data were obtained from the TCGA Research Network: http://cancergenome.nih.gov/ and downloaded via the TCGA2STAT R library (Wan et al.,

2015). Using the 881 signature genes from Helland et al. (2011) we grouped subjects into two groups based on the results in this paper, to create a "positive control" environmental variable expected to have a strong effect. Specifically, we defined an environment variable in our framework as: $E = 0$ for subtypes C1 and C2 ($n = 253$), and $E = 1$ for subtypes C4 and C5 ($n = 258$). Overall survival time (log transformed) was used as the response variable. Since these genes were ascertained on survival time, we expected the method using all genes without clustering to have the best performance, and hence one goal of this analysis was to see if ECLUST performed significantly worse as a result of summarizing the data into a lower dimension. We found 3 clusters from $TOM(X_{\text{all}})$ and 3 clusters from $TOM(X_{\text{diff}})$; results are shown in Figure 5.7, panel C. Across all models, ECLUST performs slightly better than CLUST. Furthermore it performs almost as well as the separate variable method, with the added advantage of dealing with a much smaller number of predictors (881 with SEPARATE compared to 6 with ECLUST).

5.4.3 Gestational diabetes, epigenetics and metabolic disease

Events during pregnancy are suspected to play a role in childhood obesity development but only little is known about the mechanisms involved. Indeed, children born to women who had GDM in pregnancy are more likely to be overweight and obese (Wendland et al., 2012), and evidence suggests epigenetic factors are important piece of the puzzle (Bouchard et al., 2012, 2010). Recently, methylation changes in placenta and cord blood were associated with GDM (Ruchat et al., 2013), and here we explore how these changes are associated with obesity in the children at the age of about 5 years old. DNA methylation in placenta was measured with the Infinium HumanMethylation450 BeadChip (Illumina, Inc (Bibikova et al., 2011)) microarray, in a sample of 28 women, 20 of whom had a GDM-affected pregnancy, and here, we used GDM status as our E variable, assuming that this has widespread effects on DNA methylation and on its correlation patterns. Our response, Y, is the standardized body

mass index (BMI) in the offspring at the age of 5. In contrast to the previous two examples, here we had no particular expectation of how ECLUST would perform. Using the 10,000 most variable probes, we found 2 clusters from $TOM(X_{\text{all}})$, and 75 clusters from $TOM(X_{\text{diff}})$. The predictive model results from a MARS analysis are shown in Figure 5.7, panel A. When using R^2 as the measure of performance, ECLUST outperforms both SEPARATE and CLUST methods. When using RMSE as the measure of model performance, performance tended to be better with CLUST rather than ECLUST perhaps in part due to the small number of clusters derived from $TOM(X_{\text{all}})$ relative to $TOM(X_{\text{diff}})$. Overall, the ECLUST algorithm with bagged MARS and the 1st PC of each cluster performed best, i.e., it had a better R^2 than CLUST with comparable RMSE. The sample size here is very small, and therefore the stability of the model fits is limited stability.

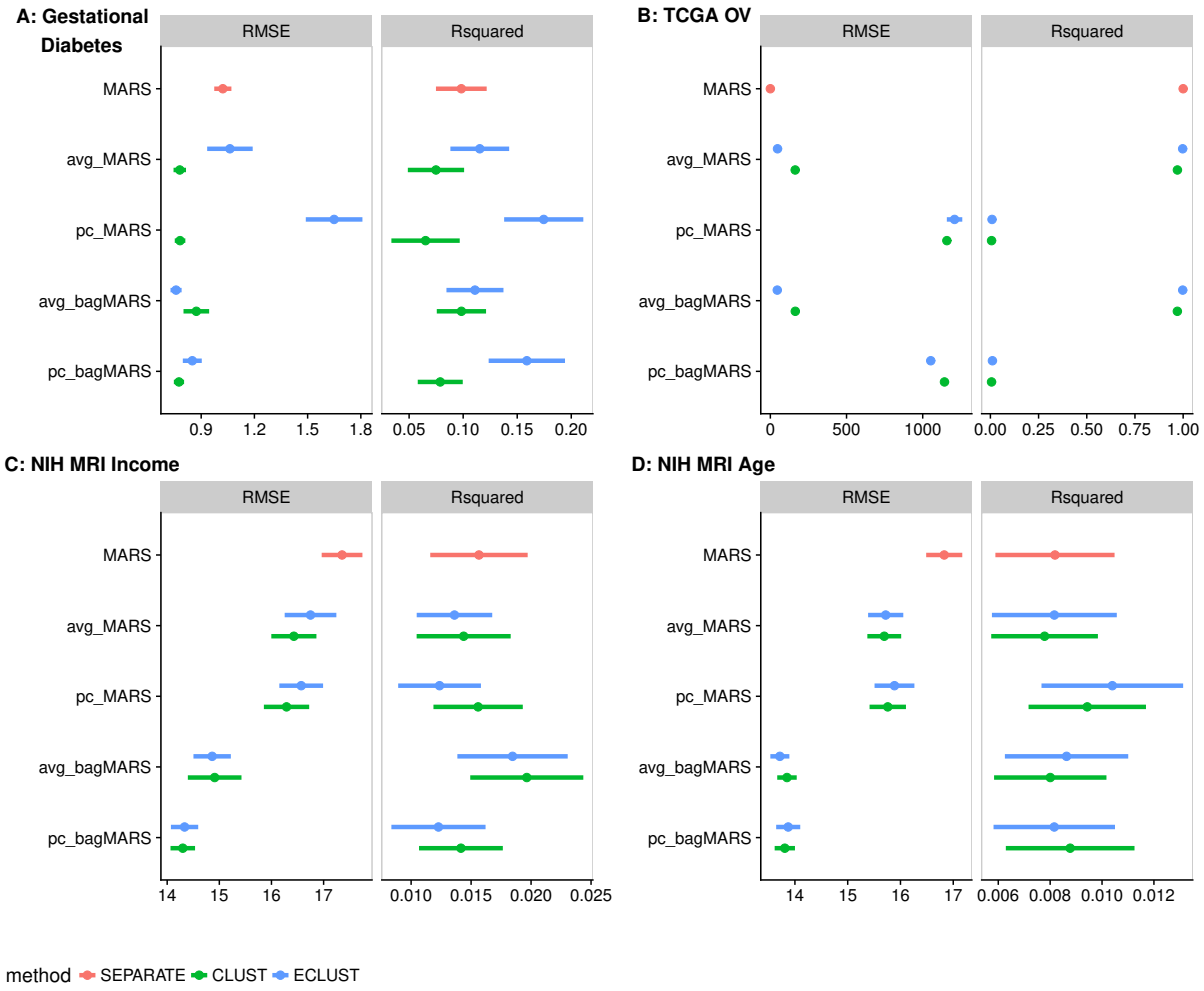


Figure 5.7: Model fit measures from analysis of three data sets: (A) Gestational diabetes birth-cohort (B) TCGA Ovarian Cancer study (C) NIH MRI Study with income as the environment variable (D) NIH MRI Study with age as the environment variable.

The probes in these clusters mapped to 164 genes and these genes were selected to conduct pathway analyses using the Ingenuity Pathway Analysis (IPA) software (Ingenuity System). IPA compares the selected genes to a reference list of genes included in many biological pathways using a hypergeometric test. Smaller p values are evidence for over-represented gene ontology categories in the input gene list. The results are summarized in Table 5.4 and provide some biological validation of our ECLUST method. For example, the Hepatic system is involved with the metabolism of glucose and lipids (Saltiel & Kahn, 2001), and behavior and neurodevelopment are associated with obesity (Epstein et al., 2004). Furthermore, it

is interesting that embryonic and organ development pathways are involved since GDM is associated with macrosomia (Ehrenberg et al., 2004).

Table 5.4: Ingenuity Pathway Analysis Results – top-ranked diseases and disorders, and physiological system development and function epigentically affected by gestational diabetes mellitus and associated with childhood body mass index

Category	Name	p values	n^a
Diseases and Disorders	Hepatic System Disease	[9.61e−7 – 5.17e−7]	75
Physiological System Development and Function			
	Behavior	[1.35e−2 – 7.82e−8]	33
	Embryonic Development	[1.35e−2 – 2.63e−8]	26
	Nervous System Development and Function	[1.35e−2 – 2.63e−8]	43
	Organ Development	[1.35e−2 – 2.63e−8]	20
	Organismal Development	[1.35e−2 – 2.63e−8]	34

[a]number of genes involved in each pathway

5.5 Discussion

The challenge of precision medicine is to appropriately fit treatments or recommendations to each individual. Data such as gene expression, DNA methylation levels, or magnetic resonance imaging (MRI) signals are examples of HD measurements that capture multiple aspects of how a tissue is functioning. These data often show patterns associated with disease, and major investments are being made in the genomics research community to generate such HD data. Analytic tools increasing prediction accuracy are needed to maximize the productivity of these investments. However, the effects of exposures have usually been overlooked, but these are crucial since they can lead to ways to intervene. Hence, it is essential to have a clear understanding of how exposures modify HD measures, and how the combination leads to disease. Existing methods for prediction (of disease), that are based on HD data and interactions with exposures, fall far short of being able to obtain this clear understanding. Most methods have low power and poor interpretability, and

furthermore, modelling and interpretation problems are exacerbated when there is interest in interactions. In general, power to estimate interactions is low, and the number of possible interactions could be enormous. Therefore, here we have proposed a strategy to leverage situations where a covariate (e.g. an exposure) has a wide-spread effect on one or more HD measures, e.g. GDM on methylation levels. We have shown that this expected pattern can be used to construct dimension-reduced predictor variables that inherently capture the systemic covariate effects. These dimension-reduced variables, constructed without using the phenotype, can then be used in predictive models of any type. In contrast to some common analysis strategies that model the effects of individual predictors on outcome, our approach makes a step towards a systems-based perspective that we believe will be more informative when exploring the factors that are associated with disease or a phenotype of interest. We have shown, through simulations and real data analysis, that incorporation of environmental factors into predictive models in a way that retains a high dimensional perspective can improve results and interpretation for both linear and non linear effects.

We proposed two key methodological steps necessary to maximize predictive model interpretability when using HD data and a binary exposure: (1) dimension reduction of HD data built on exposure sensitivity, and (2) implementation of penalized prediction models. In the first step, we proposed to identify exposure-sensitive HD pairs by contrasting the TOM between exposed and unexposed individuals; then we cluster the elements in these HD pairs to find exposure-sensitive co-regulated sets. New dimension-reduced variables that capture exposure-sensitive features (e.g. the first principal component of each cluster) were then defined. In the second step we implemented linear and non-linear variable selection methods using the dimension-reduced variables to ensure stability of the predictive model. The ECLUST method has been implemented in the `eclust` (Bhatnagar, 2017) R package publicly available on CRAN. Our method along with computationally efficient algorithms, allows for the analysis of up to 10,000 variables at a time on a laptop computer.

The methods that we have proposed here are currently only applicable when three data elements are available. Specifically a binary environmental exposure, a high dimensional dataset that can be affected by the exposure, and a single phenotype. When comparing the TOM and Pearson correlations as a measure of similarity, our simulations showed that the performance of ECLUST was worse with correlations. This speaks to the potential of developing a better measure than the difference of two matrices. For example, we are currently exploring ways in which to handle continuous exposures or multiple exposures. The best way to construct an exposure-sensitive distance matrix that can be used for clustering is not obvious in these situations. One possible solution relies on a non-parametric smoothing based approach where weighted correlations are calculated. These weights can be derived from a kernel-based summary of the exposure covariates (e.g. (Qiu et al., 2016)). Then, contrasting unweighted and weighted matrices will allow construction of covariate-sensitive clusters. The choice of summary measure for each cluster also warrants further study. While principal components and averages are well understood and easy to implement, the main shortcoming is that they involve all original variables in the group. As the size of the groups increase, the interpretability of these measures decreases. Non-negative matrix factorization (Lee & Seung, 2001) and sparse principal component analysis (SPCA) (Witten et al., 2009) are alternatives which find sparse and potentially interpretable factors. Furthermore, structured SPCA (Jenatton et al., 2009) goes beyond restricting the cardinality of the contributing factors by imposing some a priori structural constraints deemed relevant to model the data at hand.

We are all aware that our exposures and environments impact our health and risks of disease, however detecting how the environment acts is extremely difficult. Furthermore, it is very challenging to develop reliable and understandable ways of predicting the risk of disease in individuals, based on high dimensional data such as genomic or imaging measures, and this challenge is exacerbated when there are environmental exposures that lead to many subtle alterations in the genomic measurements. Hence, we have developed an algorithm

and an easy-to use software package to transform analysis of how environmental exposures impact human health, through an innovative signal-extracting approach for high dimensional measurements. Evidently, the model fitting here is performed using existing methods; our goal is to illustrate the potential of improved dimension reduction in two-stage methods, in order to generate discussion and new perspectives. If such an approach can lead to more interpretable results that identify gene-environment interactions and their effects on diseases and traits, the resulting understanding of how exposures influence the high-volume measurements now available in precision medicine will have important implications for health management and drug discovery.

Availability of data and material

1. NIH MRI Study of Normal Brain Development data are available in the Pediatric MRI Data Repository, https://pediatricmri.nih.gov/
2. Gene Expression Study of Ovarian Cancer data are available in the Genomic Data Commons repository, https://gdc.cancer.gov/, and were downloaded via the TCGA2STAT R library (Wan et al., 2015)
3. Gestational diabetes, epigenetics and metabolic disease: the clinical data, similarity matrices and cluster summaries are available at Zenodo [10.5281/zenodo.259222]. The raw analysed during the current study are not publicly available due to reasons of confidentiality, although specific collaborations with LB can be requested.

Competing interests

The authors declare that they have no competing interests.

Author's contributions

SRB, CMTG, YY, MB: conceptualization; SRB, LB, BK: data curation; SRB: formal analysis, software, visualization; SRB, CMTG: methodology; SRB, CMTG: writing-original draft; SRB, CMTG, YY, BK, ACE, MB, LB: writing-review & editing.

Acknowledgements

SRB was supported by the Ludmer Centre for Neuroinformatics and Mental Health.

Appendix C – Supplemental Methods and Simulation Results

Contains the following sections:

Appendix A

Supplemental Methods and Simulation Results for Chapter 3

A.1 Algorithm Details

In this section we provide more specific details about the algorithms used to solve the `sail` objective function.

A.1.1 Least-Squares `sail` with Strong Heredity

A more detailed algorithm for fitting the least-squares `sail` model with strong heredity is given in Algorithm 7.

Algorithm 7 Blockwise Coordinate Descent for Least-Squares `sail` with Strong Heredity

function `sail`$(\boldsymbol{X}, Y, X_E, \texttt{basis}, \lambda, \alpha, w_j, w_E, w_{jE}, \epsilon)$ ▷ Algorithm for solving (3.10)

$\Psi_j \leftarrow \texttt{basis}(X_j)$, $\widetilde{\Psi}_j \leftarrow X_E \circ \Psi_j$ for $j = 1, \ldots, p$

Initialize: $\beta_0^{(0)} \leftarrow \bar{Y}$, $\beta_E^{(0)} = \boldsymbol{\theta}_j^{(0)} = \gamma_j^{(0)} \leftarrow 0$ for $j = 1, \ldots, p$.

Set iteration counter $k \leftarrow 0$

$R^* \leftarrow Y - \beta_0^{(k)} - \beta_E^{(k)} X_E - \sum_j (\boldsymbol{\Psi}_j + \gamma_j^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_j) \boldsymbol{\theta}_j^{(k)}$

repeat

- To update $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$

  $\widetilde{X}_j \leftarrow \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_j \boldsymbol{\theta}_j^{(k)}$ for $j = 1, \ldots, p$

  $R \leftarrow R^* + \sum_{j=1}^p \gamma_j^{(k)} \widetilde{X}_j$

$$\boldsymbol{\gamma}^{(k)(new)} \leftarrow \arg\min_{\boldsymbol{\gamma}} \frac{1}{2n} \left\| R - \sum_j \gamma_j \widetilde{X}_j \right\|_2^2 + \lambda\alpha \sum_j w_{jE} |\gamma_j|$$

  $\Delta = \sum_j (\gamma_j^{(k)} - \gamma_j^{(k)(new)}) \widetilde{X}_j$

  $R^* \leftarrow R^* + \Delta$

- To update $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p)$

  $\widetilde{X}_j \leftarrow \boldsymbol{\Psi}_j + \gamma_j^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_j$ for $j = 1, \ldots, p$

  for $j = 1, \ldots, p$ **do**

    $R \leftarrow R^* + \widetilde{X}_j \boldsymbol{\theta}_j^{(k)}$

$$\boldsymbol{\theta}_j^{(k)(new)} \leftarrow \arg\min_{\boldsymbol{\theta}_j} \frac{1}{2n} \left\| R - \widetilde{X}_j \boldsymbol{\theta}_j \right\|_2^2 + \lambda(1-\alpha) w_j \|\theta_j\|_2$$

    $\Delta = \widetilde{X}_j (\boldsymbol{\theta}_j^{(k)} - \boldsymbol{\theta}_j^{(k)(new)})$

    $R^* \leftarrow R^* + \Delta$

- To update β_E

  $\widetilde{X}_E \leftarrow X_E + \sum_j \gamma_j^{(k)} \widetilde{\boldsymbol{\Psi}}_j \boldsymbol{\theta}_j^{(k)}$

  $R \leftarrow R^* + \beta_E^{(k)} \widetilde{X}_E$

$$\beta_E^{(k)(new)} \leftarrow S\left(\frac{1}{n \cdot w_E} \widetilde{X}_E^\top R, \lambda(1-\alpha) \right)$$

▷ $S(x,t) = \text{sign}(x)(|x| - t)_+$

  $\Delta = (\beta_E^{(k)} - \beta_E^{(k)(new)}) \widetilde{X}_E$

  $R^* \leftarrow R^* + \Delta$

- To update β_0

  $R \leftarrow R^* + \beta_0^{(k)}$

$$\beta_0^{(k)(new)} \leftarrow \frac{1}{n} R^* \cdot \mathbf{1}$$

  $\Delta = \beta_0^{(k)} - \beta_0^{(k)(new)}$

  $R^* \leftarrow R^* + \Delta$

$k \leftarrow k + 1$

until convergence criterion is satisfied: $\left| Q(\boldsymbol{\Theta}^{(k-1)}) - Q(\boldsymbol{\Theta}^{(k)}) \right| / Q(\boldsymbol{\Theta}^{(k-1)}) < \epsilon$

A.1.2 Details on Update for $\boldsymbol{\theta}$

Here we discuss a computational speedup in the updates for the $\boldsymbol{\theta}$ parameter. The partial residual (R_s) used for updating $\boldsymbol{\theta}_s$ ($s \in 1, \ldots, p$) at the kth iteration is given by

$$R_s = Y - \widetilde{Y}^{(k)}_{(-s)} \tag{A.1}$$

where $\widetilde{Y}^{(k)}_{(-s)}$ is the fitted value at the kth iteration excluding the contribution from $\boldsymbol{\Psi}_s$:

$$\widetilde{Y}^{(k)}_{(-s)} = \beta_0^{(k)} - \beta_E^{(k)} X_E - \sum_{\ell \neq s} \boldsymbol{\Psi}_\ell \boldsymbol{\theta}_\ell^{(k)} - \sum_{\ell \neq s} \gamma_\ell^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_\ell \boldsymbol{\theta}_\ell^{(k)} \tag{A.2}$$

Using (A.2), (A.1) can be re-written as

$$\begin{aligned} R_s &= Y - \beta_0^{(k)} - \beta_E^{(k)} X_E - \sum_{j=1}^{p} (\boldsymbol{\Psi}_j + \gamma_j^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_j) \boldsymbol{\theta}_j^{(k)} + (\boldsymbol{\Psi}_s + \gamma_s^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_s) \boldsymbol{\theta}_s^{(k)} \\ &= R^* + (\boldsymbol{\Psi}_s + \gamma_s^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_s) \boldsymbol{\theta}_s^{(k)} \end{aligned} \tag{A.3}$$

where

$$R^* = Y - \beta_0^{(k)} - \beta_E^{(k)} X_E - \sum_{j=1}^{p} (\boldsymbol{\Psi}_j + \gamma_j^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_j) \boldsymbol{\theta}_j^{(k)} \tag{A.4}$$

Denote $\boldsymbol{\theta}_s^{(k)(\text{new})}$ the solution for predictor s at the kth iteration, given by:

$$\boldsymbol{\theta}_s^{(k)(new)} = \arg\min_{\boldsymbol{\theta}_j} \frac{1}{2n} \left\| R_s - (\boldsymbol{\Psi}_s + \gamma_s^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_s) \boldsymbol{\theta}_j \right\|_2^2 + \lambda(1-\alpha) w_s \|\theta_j\|_2 \tag{A.5}$$

Now we want to update the parameters for the next predictor $\boldsymbol{\theta}_{s+1}$ ($s+1 \in 1, \ldots, p$) at the kth iteration. The partial residual used to update $\boldsymbol{\theta}_{s+1}$ is given by

$$R_{s+1} = R^* + (\boldsymbol{\Psi}_{s+1} + \gamma_{s+1}^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_{s+1}) \boldsymbol{\theta}_{s+1}^{(k)} + (\boldsymbol{\Psi}_s + \gamma_s^{(k)} \beta_E^{(k)} \widetilde{\boldsymbol{\Psi}}_s)(\boldsymbol{\theta}_s^{(k)} - \boldsymbol{\theta}_s^{(k)(new)}) \tag{A.6}$$

where R^* is given by (A.4), $\boldsymbol{\theta}_s^{(k)}$ is the parameter value prior to the update, and $\boldsymbol{\theta}_s^{(k)(new)}$ is the updated value given by (A.5). Taking the difference between (A.3) and (A.6) gives

$$\begin{aligned}\Delta &= R_t - R_s \\ &= (\boldsymbol{\Psi}_t + \gamma_t^{(k)}\beta_E^{(k)}\widetilde{\boldsymbol{\Psi}}_t)\boldsymbol{\theta}_t^{(k)} + (\boldsymbol{\Psi}_s + \gamma_s^{(k)}\beta_E^{(k)}\widetilde{\boldsymbol{\Psi}}_s)(\boldsymbol{\theta}_s^{(k)} - \boldsymbol{\theta}_s^{(k)(new)}) - (\boldsymbol{\Psi}_s + \gamma_s^{(k)}\beta_E^{(k)}\widetilde{\boldsymbol{\Psi}}_s)\boldsymbol{\theta}_s^{(k)} \\ &= (\boldsymbol{\Psi}_t + \gamma_t^{(k)}\beta_E^{(k)}\widetilde{\boldsymbol{\Psi}}_t)\boldsymbol{\theta}_t^{(k)} - (\boldsymbol{\Psi}_s + \gamma_s^{(k)}\beta_E^{(k)}\widetilde{\boldsymbol{\Psi}}_s)\boldsymbol{\theta}_s^{(k)(new)} \qquad \text{(A.7)}\end{aligned}$$

Therefore $R_t = R_s + \Delta$, and the partial residual for updating the next predictor can be computed by updating the previous partial residual by Δ, given by (A.7). This formulation can lead to computational speedups especially when $\Delta = 0$, meaning the partial residual does not need to be re-calculated.

A.1.3 Least-Squares `sail` with Weak Heredity

The least-squares `sail` model with weak heredity has the form

$$\hat{Y} = \beta_0 \cdot \mathbf{1} + \sum_{j=1}^{p} \boldsymbol{\Psi}_j\boldsymbol{\theta}_j + \beta_E X_E + \sum_{j=1}^{p} \gamma_j(X_E \circ \boldsymbol{\Psi}_j)(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j) \qquad \text{(A.8)}$$

The objective function is given by

$$Q(\boldsymbol{\Theta}) = \frac{1}{2n}\left\|Y - \hat{Y}\right\|_2^2 + \lambda(1-\alpha)\left(w_E|\beta_E| + \sum_{j=1}^{p} w_j\|\boldsymbol{\theta}_j\|_2\right) + \lambda\alpha\sum_{j=1}^{p} w_{jE}|\gamma_j| \qquad \text{(A.9)}$$

Denote the n-dimensional residual column vector $R = Y - \hat{Y}$. The subgradient equations are given by

$$\frac{\partial Q}{\partial \beta_0} = \frac{1}{n}\left(Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \boldsymbol{\theta}_j - \beta_E X_E - \sum_{j=1}^{p} \gamma_j (X_E \circ \boldsymbol{\Psi}_j)(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j)\right)^\top \mathbf{1} = 0 \tag{A.10}$$

$$\frac{\partial Q}{\partial \beta_E} = -\frac{1}{n}\left(X_E + \sum_{j=1}^{p} \gamma_j (X_E \circ \boldsymbol{\Psi}_j)\mathbf{1}_{m_j}\right)^\top R + \lambda(1-\alpha) w_E s_1 = 0 \tag{A.11}$$

$$\frac{\partial Q}{\partial \boldsymbol{\theta}_j} = -\frac{1}{n}\left(\boldsymbol{\Psi}_j + \gamma_j (X_E \circ \boldsymbol{\Psi}_j)\right)^\top R + \lambda(1-\alpha) w_j s_2 = \mathbf{0} \tag{A.12}$$

$$\frac{\partial Q}{\partial \gamma_j} = -\frac{1}{n}\left((X_E \circ \boldsymbol{\Psi}_j)(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j)\right)^\top R + \lambda\alpha w_{jE} s_3 = 0 \tag{A.13}$$

where s_1 is in the subgradient of the ℓ_1 norm:

$$s_1 \in \begin{cases} \operatorname{sign}(\beta_E) & \text{if } \beta_E \neq 0 \\ [-1, 1] & \text{if } \beta_E = 0, \end{cases}$$

s_2 is in the subgradient of the ℓ_2 norm:

$$s_2 \in \begin{cases} \dfrac{\boldsymbol{\theta}_j}{\|\boldsymbol{\theta}_j\|_2} & \text{if } \boldsymbol{\theta}_j \neq \mathbf{0} \\ u \in \mathbb{R}^{m_j} : \|u\|_2 \leq 1 & \text{if } \boldsymbol{\theta}_j = \mathbf{0}, \end{cases}$$

and s_3 is in the subgradient of the ℓ_1 norm:

$$s_3 \in \begin{cases} \operatorname{sign}(\gamma_j) & \text{if } \gamma_j \neq 0 \\ [-1, 1] & \text{if } \gamma_j = 0. \end{cases}$$

Define the partial residuals, without the jth predictor for $j = 1, \ldots, p$, as

$$R_{(-j)} = Y - \beta_0 \cdot \mathbf{1} - \sum_{\ell \neq j} \boldsymbol{\Psi}_\ell \boldsymbol{\theta}_\ell - \beta_E X_E - \sum_{\ell \neq j} \gamma_\ell (X_E \circ \boldsymbol{\Psi}_\ell)(\beta_E \cdot \mathbf{1}_{m_\ell} + \boldsymbol{\theta}_\ell)$$

the partial residual without X_E as

$$R_{(-E)} = Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \boldsymbol{\theta}_j - \sum_{j=1}^{p} \gamma_j (X_E \circ \boldsymbol{\Psi}_j) \boldsymbol{\theta}_j$$

and the partial residual without the jth interaction for $j = 1, \ldots, p$

$$R_{(-jE)} = Y - \beta_0 \cdot \mathbf{1} - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \boldsymbol{\theta}_j - \beta_E X_E - \sum_{\ell \neq j} \gamma_\ell (X_E \circ \boldsymbol{\Psi}_\ell)(\beta_E \cdot \mathbf{1}_{m_\ell} + \boldsymbol{\theta}_\ell)$$

From the subgradient Equation (A.11), we see that $\beta_E = 0$ is a solution if

$$\frac{1}{w_E} \left| \frac{1}{n} \left(X_E + \sum_{j=1}^{p} \gamma_j (X_E \circ \boldsymbol{\Psi}_j) \mathbf{1}_{m_j} \right)^\top R_{(-E)} \right| \leq \lambda(1 - \alpha) \tag{A.14}$$

From the subgradient Equation (A.12), we see that $\boldsymbol{\theta}_j = \mathbf{0}$ is a solution if

$$\frac{1}{w_j} \left\| \frac{1}{n} \left(\boldsymbol{\Psi}_j + \gamma_j (X_E \circ \boldsymbol{\Psi}_j) \right)^\top R_{(-j)} \right\|_2 \leq \lambda(1 - \alpha) \tag{A.15}$$

From the subgradient Equation (A.13), we see that $\gamma_j = 0$ is a solution if

$$\frac{1}{w_{jE}} \left| \frac{1}{n} \left((X_E \circ \boldsymbol{\Psi}_j)(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j) \right)^\top R_{(-jE)} \right| \leq \lambda\alpha \tag{A.16}$$

From the subgradient equations we see that

$$\hat{\beta}_0 = \left(Y - \sum_{j=1}^{p} \boldsymbol{\Psi}_j \hat{\boldsymbol{\theta}}_j - \hat{\beta}_E X_E - \sum_{j=1}^{p} \hat{\gamma}_j (X_E \circ \boldsymbol{\Psi}_j)(\hat{\beta}_E \cdot \mathbf{1}_{m_j} + \hat{\boldsymbol{\theta}}_j) \right)^\top \mathbf{1} \tag{A.17}$$

$$\hat{\beta}_E = S\left(\frac{1}{n \cdot w_E} \left(X_E + \sum_{j=1}^{p} \hat{\gamma}_j (X_E \circ \boldsymbol{\Psi}_j) \mathbf{1}_{m_j} \right)^\top R_{(-E)}, \lambda(1-\alpha) \right) \tag{A.18}$$

$$\lambda(1-\alpha) w_j \frac{\boldsymbol{\theta}_j}{\|\boldsymbol{\theta}_j\|_2} = \frac{1}{n} \left(\boldsymbol{\Psi}_j + \gamma_j (X_E \circ \boldsymbol{\Psi}_j) \right)^\top R_{(-j)} \tag{A.19}$$

$$\hat{\gamma}_j = S\left(\frac{1}{n \cdot w_{jE}} \left((X_E \circ \boldsymbol{\Psi}_j)(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j) \right)^\top R_{(-jE)}, \lambda\alpha \right) \tag{A.20}$$

where $S(x,t) = \text{sign}(x)(|x| - t)$ is the soft-thresholding operator. As was the case in the strong heredity `sail` model, there are closed form solutions for the intercept and β_E, each γ_j also has a closed form solution and can be solved efficiently for $j = 1, \ldots, p$ using the coordinate descent procedure implemented in the `glmnet` package (J. Friedman et al., 2010), while we use the quadratic majorization technique implemented in the `gglasso` package (Y. Yang & Zou, 2015) to solve (A.19). Algorithm 8 details the procedure used to fit the least-squares weak heredity `sail` model.

Lambda Max

The smallest value of λ for which the entire parameter vector $(\beta_E, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p, \gamma_1, \ldots, \gamma_p)$ is $\mathbf{0}$ is:

Algorithm 8 Coordinate descent for least-squares `sail` with weak heredity

function `sail`$(\boldsymbol{X}, Y, X_E, \texttt{basis}, \lambda, \alpha, w_j, w_E, w_{jE}, \epsilon)$ ▷ Algorithm for solving (A.9)

$\Psi_j \leftarrow \texttt{basis}(X_j)$, $\widetilde{\Psi}_j \leftarrow X_E \circ \Psi_j$ for $j = 1, \ldots, p$

Initialize: $\beta_0^{(0)} \leftarrow \bar{Y}$, $\beta_E^{(0)} = \boldsymbol{\theta}_j^{(0)} = \gamma_j^{(0)} \leftarrow 0$ for $j = 1, \ldots, p$.

Set iteration counter $k \leftarrow 0$

$R^* \leftarrow Y - \beta_0^{(k)} - \beta_E^{(k)} X_E - \sum_j \boldsymbol{\Psi}_j \boldsymbol{\theta}_j^{(k)} - \sum_j \gamma_j^{(k)} \widetilde{\boldsymbol{\Psi}}_j (\beta_E^{(k)} \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j^{(k)})$

repeat

- To update $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$

  $\widetilde{X}_j \leftarrow \widetilde{\boldsymbol{\Psi}}_j (\beta_E^{(k)} \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j^{(k)})$ for $j = 1, \ldots, p$

  $R \leftarrow R^* + \sum_{j=1}^p \gamma_j^{(k)} \widetilde{X}_j$

$$\boldsymbol{\gamma}^{(k)(new)} \leftarrow \arg\min_{\boldsymbol{\gamma}} \frac{1}{2n} \left\| R - \sum_j \gamma_j \widetilde{X}_j \right\|_2^2 + \lambda\alpha \sum_j w_{jE} |\gamma_j|$$

  $\Delta = \sum_j (\gamma_j^{(k)} - \gamma_j^{(k)(new)}) \widetilde{X}_j$

  $R^* \leftarrow R^* + \Delta$

- To update $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p)$

  $\widetilde{X}_j \leftarrow \boldsymbol{\Psi}_j + \gamma_j^{(k)} \widetilde{\boldsymbol{\Psi}}_j$ for $j = 1, \ldots, p$

  for $j = 1, \ldots, p$ **do**

    $R \leftarrow R^* + \widetilde{X}_j \boldsymbol{\theta}_j^{(k)}$

$$\boldsymbol{\theta}_j^{(k)(new)} \leftarrow \arg\min_{\boldsymbol{\theta}_j} \frac{1}{2n} \left\| R - \widetilde{X}_j \boldsymbol{\theta}_j \right\|_2^2 + \lambda(1-\alpha) w_j \|\theta_j\|_2$$

    $\Delta = \widetilde{X}_j (\boldsymbol{\theta}_j^{(k)} - \boldsymbol{\theta}_j^{(k)(new)})$

    $R^* \leftarrow R^* + \Delta$

- To update β_E

  $\widetilde{X}_E \leftarrow X_E + \sum_j \gamma_j^{(k)} \widetilde{\boldsymbol{\Psi}}_j \mathbf{1}_{m_j}$

  $R \leftarrow R^* + \beta_E^{(k)} \widetilde{X}_E$

$$\beta_E^{(k)(new)} \leftarrow S\left(\frac{1}{n \cdot w_E} \widetilde{X}_E^\top R, \lambda(1-\alpha) \right)$$

▷ $S(x,t) = \text{sign}(x)(|x| - t)_+$

  $\Delta = (\beta_E^{(k)} - \beta_E^{(k)(new)}) \widetilde{X}_E$

  $R^* \leftarrow R^* + \Delta$

- To update β_0

  $R \leftarrow R^* + \beta_0^{(k)}$

$$\beta_0^{(k)(new)} \leftarrow \frac{1}{n} R^* \cdot \mathbf{1}$$

  $\Delta = \beta_0^{(k)} - \beta_0^{(k)(new)}$

  $R^* \leftarrow R^* + \Delta$

$k \leftarrow k + 1$

until convergence criterion is satisfied: $\left| Q(\boldsymbol{\Theta}^{(k-1)}) - Q(\boldsymbol{\Theta}^{(k)}) \right| / Q(\boldsymbol{\Theta}^{(k-1)}) < \epsilon$

$$\lambda_{max} = \frac{1}{n} \max \left\{ \frac{1}{(1-\alpha)w_E} \left(X_E + \sum_{j=1}^{p} \gamma_j (X_E \circ \mathbf{\Psi}_j) \mathbf{1}_{m_j} \right)^\top R_{(-E)}, \right.$$
$$\max_j \frac{1}{(1-\alpha)w_j} \left\| (\mathbf{\Psi}_j + \gamma_j (X_E \circ \mathbf{\Psi}_j))^\top R_{(-j)} \right\|_2,$$
$$\left. \max_j \frac{1}{\alpha w_{jE}} \left((X_E \circ \mathbf{\Psi}_j)(\beta_E \cdot \mathbf{1}_{m_j} + \boldsymbol{\theta}_j) \right)^\top R_{(-jE)} \right\} \quad \text{(A.21)}$$

which reduces to

$$\lambda_{max} = \frac{1}{n(1-\alpha)} \max \left\{ \frac{1}{w_E} (X_E)^\top R_{(-E)}, \max_j \frac{1}{w_j} \left\| (\mathbf{\Psi}_j)^\top R_{(-j)} \right\|_2 \right\}$$

This is the same λ_{max} as the least-squares strong heredity `sail` model.

A.2 Simulation Results

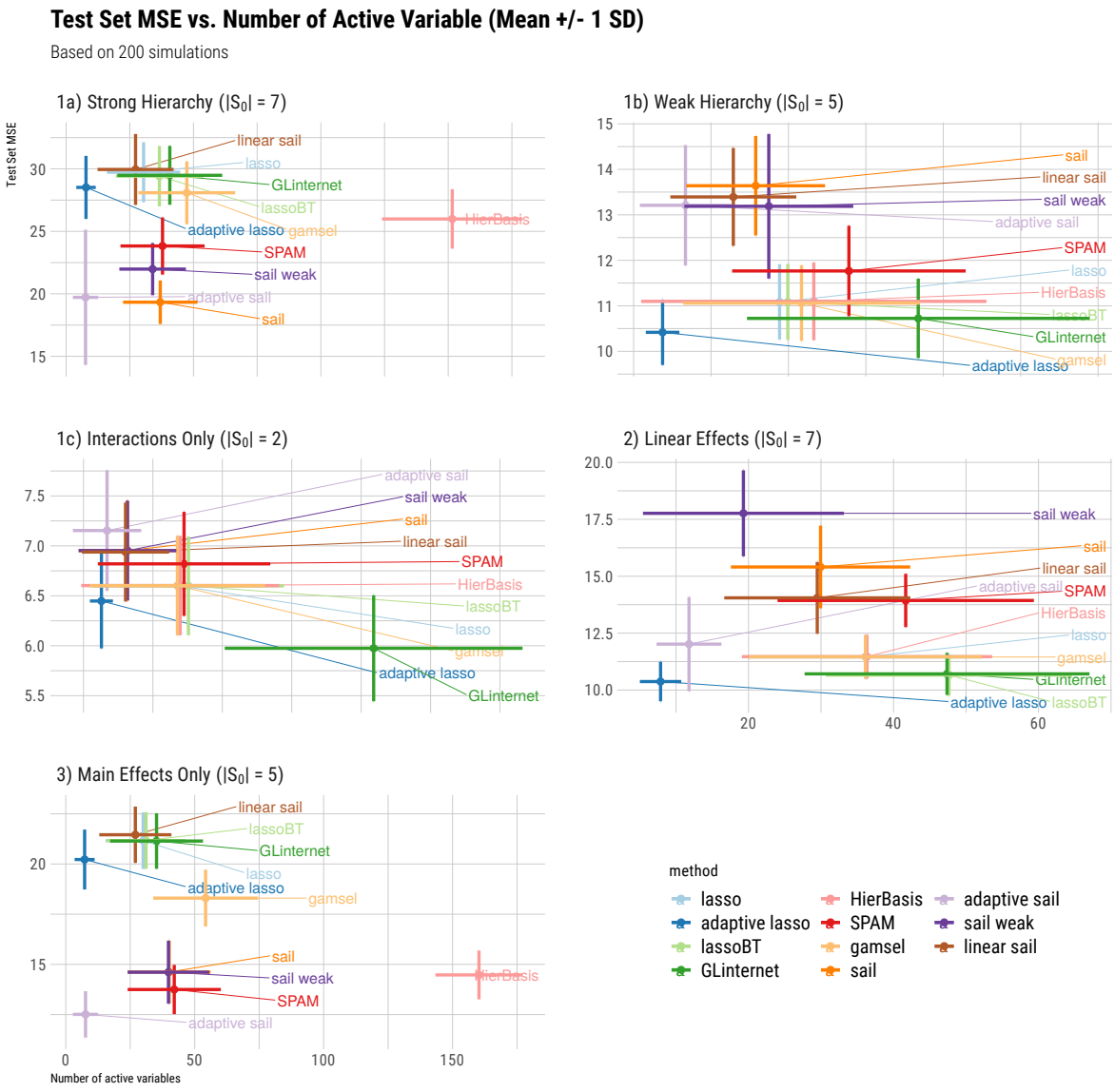


Figure A.1: Test set MSE vs number of active variables results.

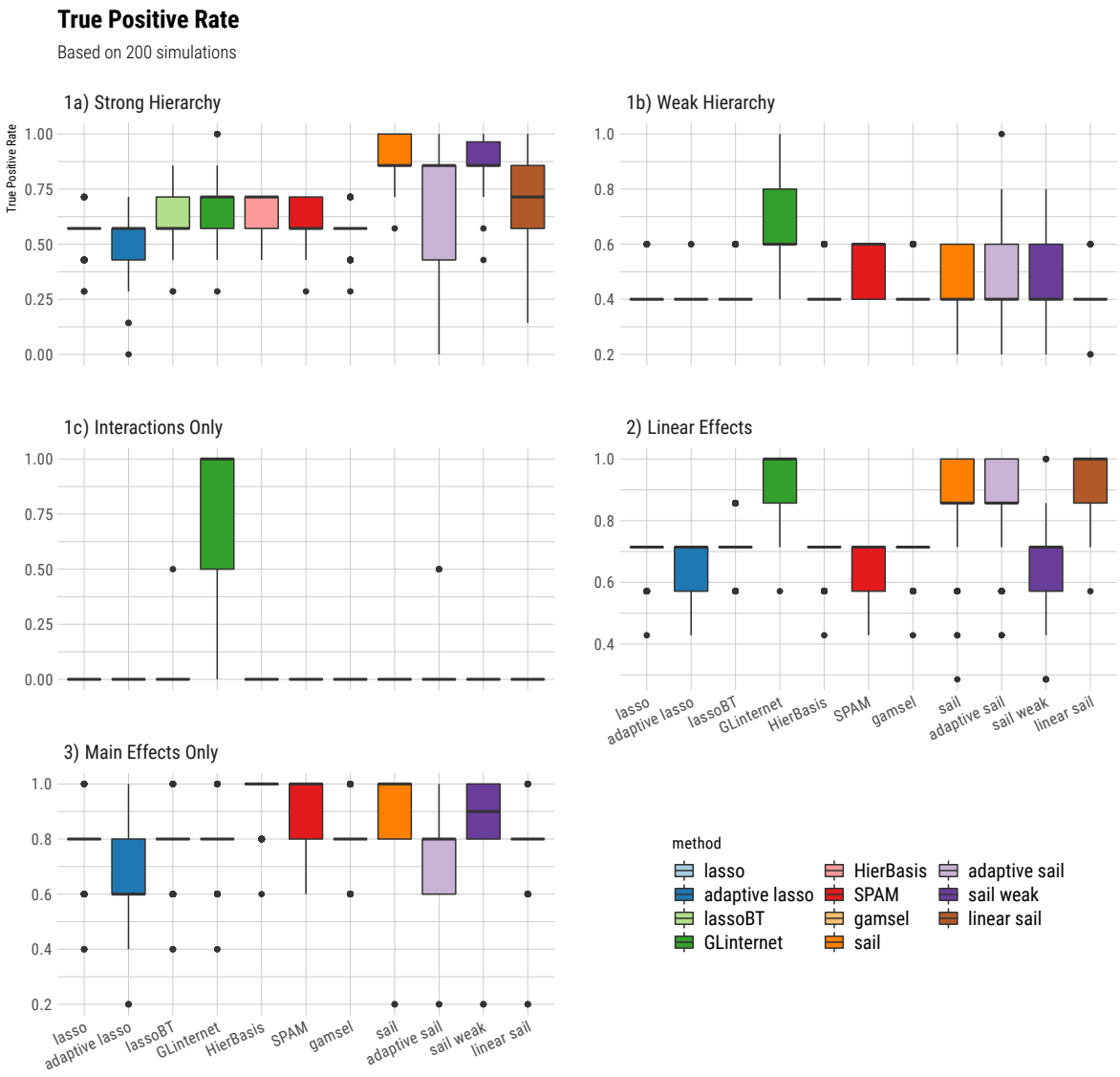


Figure A.2: True positive rate results.

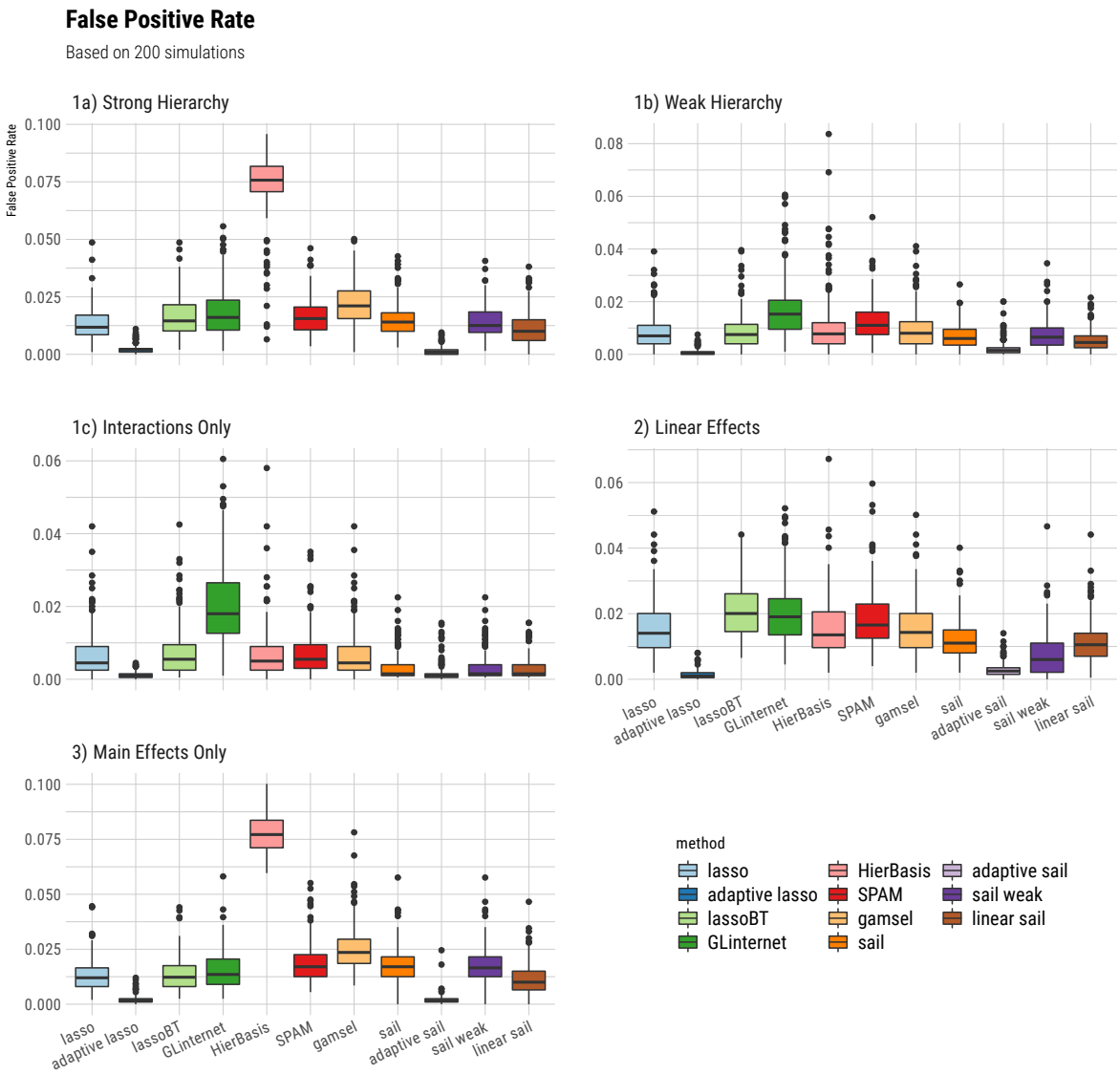


Figure A.3: False positive rate results.

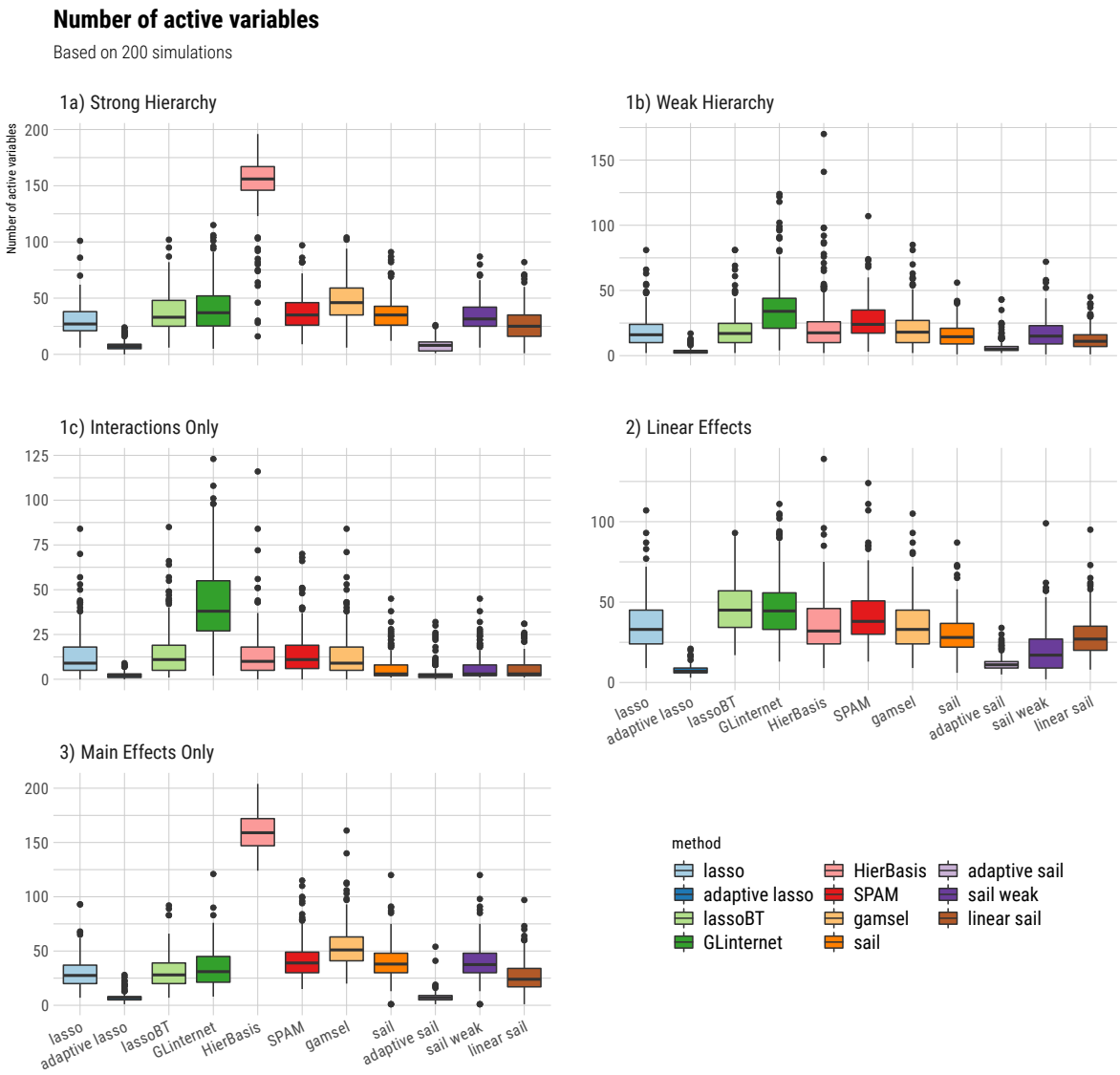


Figure A.4: Number of active variables results.

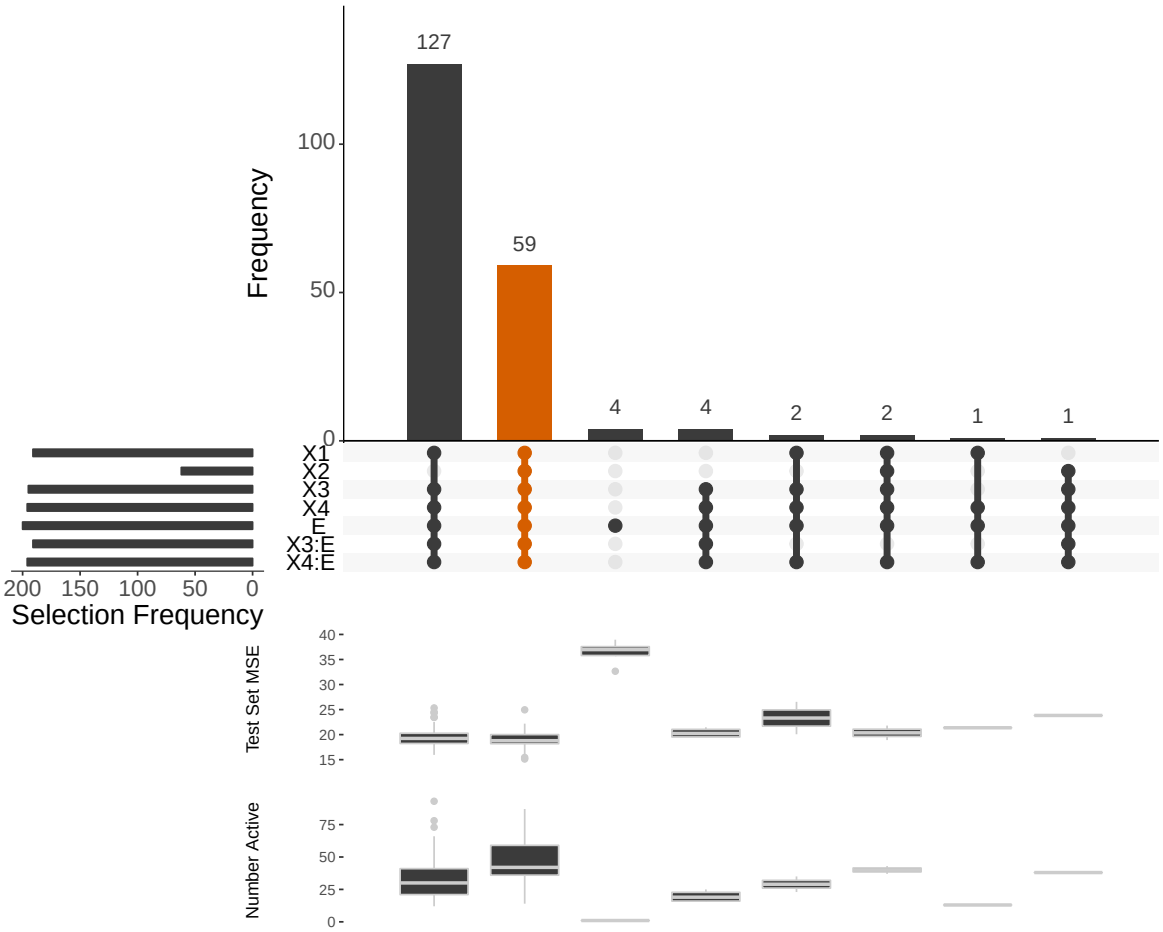


Figure A.5: Selection rates across 200 simulations of scenario 1a) for strong heredity `sail`.

A.3 `sail` Package Showcase

In this section we briefly introduce the freely available and open source `sail` package in `R`. More comprehensive documentation is available at https://sahirbhatnagar.com/sail. Note that this entire section is reproducible; the code and text are combined in an `.Rnw`[1] file and compiled using `knitr` (Xie, 2015).

[1]scripts available at https://github.com/sahirbhatnagar/sail/tree/master/manuscript

A.3.1 Installation

The package can be installed from GitHub via

```
install.packages("pacman")
pacman::p_load_gh('sahirbhatnagar/sail')
```

A.3.2 Quick Start

We give a quick overview of the main functions and go into details in other vignettes. We will use the simulated data which ships with the package and can be loaded via:

```
library(sail)
data("sailsim")
names(sailsim)

## [1] "x"         "y"         "e"         "f1"        "f2"        "f3"
## [7] "f4"        "f3.inter" "f4.inter"
```

We first define a basis expansion. In this example we use B-splines with degree 5.

```
library(splines)
f.basis <- function(x) splines::bs(x, degree = 5)
```

Next we fit the model using the most basic call to `sail`

```
fit <- sail(x = sailsim$x, y = sailsim$y, e = sailsim$e, basis = f.basis)
```

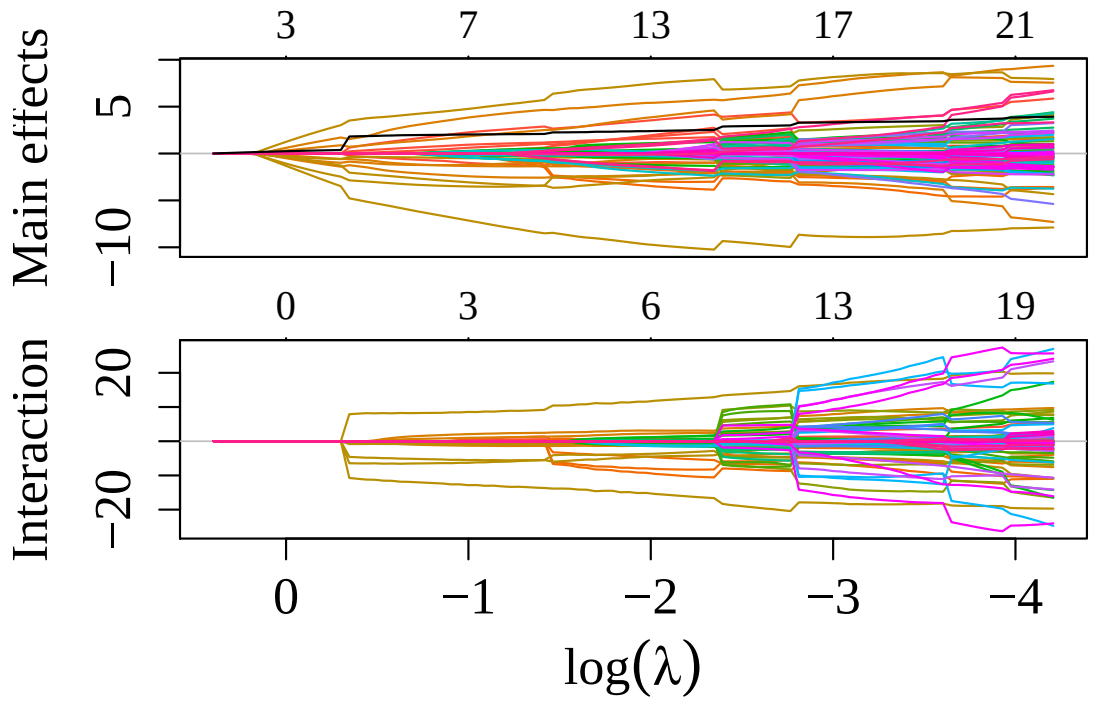
`fit` is an object of class `sail` that contains all the relevant information of the fitted model including the estimated coefficients at each value of λ (by default the program chooses its own decreasing sequence of 100 λ values). There are `print`, `plot`, `coef` and `predict` methods of objects of class `sail`.

When `expand = TRUE` (i.e. the user did not provide their own design matrix), the `df_main` and `df_interaction` columns correspond to the number of non-zero predictors present in the model before basis expansion. This does not correspond to the number of non-zero coefficients in the model, but rather the number of unique variables. In this example

we expanded each column of $\mathbf{X}$ to five columns. If `df_main=4`, `df_interaction=2` and `df_environment=1`, then the total number of non-zero coefficients would be $5 \times (4 + 2) + 1$.

The entire solution path can be plotted via the `plot` method for objects of class `sail`. The y-axis is the value of the coefficient and the x-axis is the $\log(\lambda)$. Each line represents a coefficient in the model, and each color represents a variable (i.e. in this example a given variable will have 5 lines when it is non-zero). The numbers at the top of the plot represent the number of non-zero variables in the model: top panel (`df_main` + `df_environment`), bottom panel (`df_interaction`). The black line is the coefficient path for the environment variable.

```
plot(fit)
```



The estimated coefficients at each value of lambda is given by (matrix partially printed here for brevity)

```
coef(fit)[1:6,50:55]
```

```
## 6 x 6 sparse Matrix of class "dgCMatrix"
##                     s50        s51        s52        s53        s54
## (Intercept)  5.2908242  5.2837492  5.2803715  5.2753572  5.2717869
## X1_1        -0.9792849 -0.9604046 -0.9449616 -0.9220738 -0.9171304
## X1_2         1.6903252  1.7894886  1.8924485  1.9952094  2.1042358
## X1_3         1.6463057  1.7049842  1.7722916  1.8251613  1.8951562
## X1_4         1.5224653  1.5433528  1.5663095  1.5854332  1.6102159
## X1_5         3.3386403  3.4183219  3.4908074  3.5763781  3.6633809
##                     s55
## (Intercept)  5.2695399
## X1_1        -0.9270958
## X1_2         2.2058453
## X1_3         1.9642875
## X1_4         1.6322047
## X1_5         3.7453708
```

The corresponding predicted response at each value of lambda (matrix partially printed here for brevity):

```
predict(fit)[1:5,50:55]
```

```
##            s50       s51       s52       s53       s54       s55
## [1,]  6.244693  6.199302  6.185402  6.177991  6.156173  6.124271
## [2,]  3.002799  2.995418  3.038701  3.079700  3.143715  3.209065
## [3,]  2.073305  2.043476  2.016319  1.997172  1.966900  1.957271
## [4,] 13.488945 13.490766 13.360998 13.384370 13.350671 13.324117
## [5,]  1.225516  1.210346  1.134420  1.156355  1.156696  1.156135
```

The predicted response at a specific value of lambda can be specified by the `s` argument:

```
predict(fit, s = 0.8)[1:5, ]
```

```
## [1] 5.624232 4.940944 3.847965 6.687777 3.058125
```

You can specify more than one value for ‘s‘:

```
predict(fit, s = c(0.8, 0.2))[1:5, ]
```

```
##            1         2
## [1,] 5.624232  6.523025
## [2,] 4.940944  2.975046
## [3,] 3.847965  2.326672
```

```
## [4,] 6.687777 13.956092
## [5,] 3.058125  1.568897
```

You can also extract a list of active variables (i.e. variables with a non-zero estimated coefficient) for each value of lambda:

```
fit[["active"]][50:55]

## [[1]]
##  [1] "X1"    "X2"    "X3"    "X4"    "X8"    "X10"   "X11"   "X20"
##  [9] "X1:E"  "X2:E"  "X3:E"  "X4:E"  "X8:E"  "X11:E" "E"
##
## [[2]]
##  [1] "X1"    "X2"    "X3"    "X4"    "X6"    "X8"    "X10"   "X11"
##  [9] "X20"   "X1:E"  "X2:E"  "X3:E"  "X4:E"  "X8:E"  "X11:E" "E"
##
## [[3]]
##  [1] "X1"    "X2"    "X3"    "X4"    "X6"    "X8"    "X10"   "X11"
##  [9] "X16"   "X20"   "X1:E"  "X2:E"  "X3:E"  "X4:E"  "X8:E"  "X11:E"
## [17] "E"
##
## [[4]]
##  [1] "X1"    "X2"    "X3"    "X4"    "X6"    "X8"    "X10"   "X11"
##  [9] "X15"   "X16"   "X19"   "X20"   "X1:E"  "X2:E"  "X3:E"  "X4:E"
## [17] "X8:E"  "X11:E" "E"
##
## [[5]]
##  [1] "X1"    "X2"    "X3"    "X4"    "X5"    "X6"    "X8"    "X10"
##  [9] "X11"   "X15"   "X16"   "X19"   "X20"   "X1:E"  "X2:E"  "X3:E"
## [17] "X4:E"  "X8:E"  "X11:E" "E"
##
## [[6]]
##  [1] "X1"    "X2"    "X3"    "X4"    "X5"    "X6"    "X8"    "X10"
##  [9] "X11"   "X15"   "X16"   "X19"   "X20"   "X1:E"  "X2:E"  "X3:E"
## [17] "X4:E"  "X8:E"  "X11:E" "E"
```

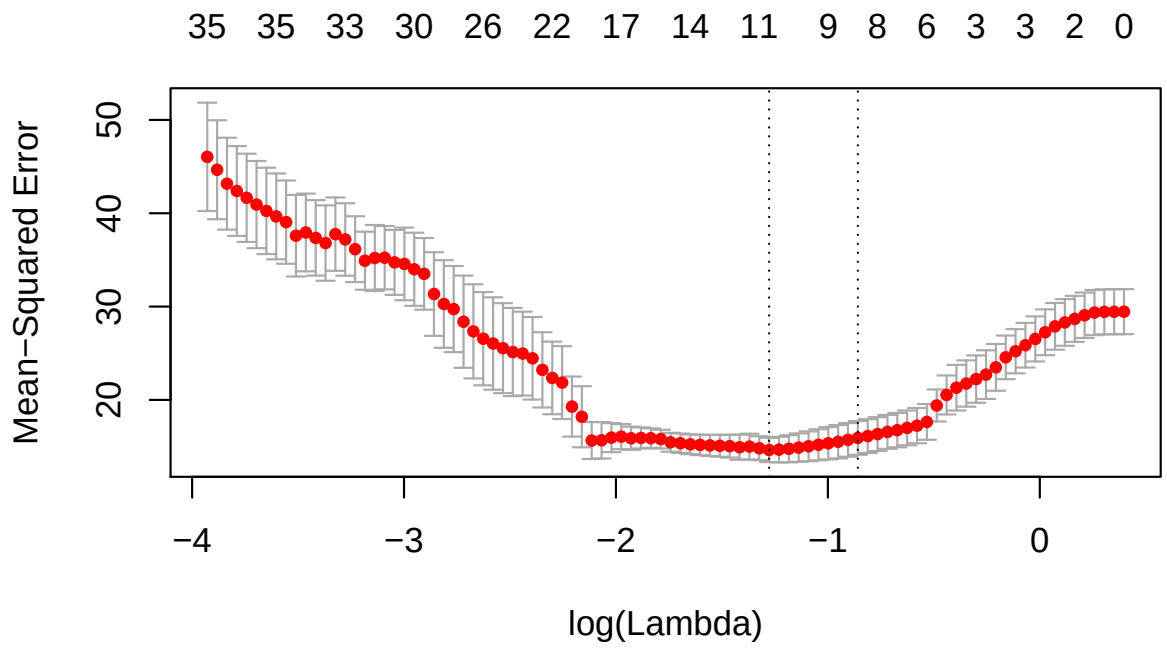
A.3.3 Cross-Validation

`cv.sail` is the main function to do cross-validation along with `plot`, `predict`, and `coef` methods for objects of class `cv.sail`. We run it in parallel:

```
set.seed(432) # to reproduce results (randomness due to CV folds)
library(doMC)
registerDoMC(cores = 8)
cvfit <- cv.sail(x = sailsim$x, y = sailsim$y, e = sailsim$e, basis = f.basis,
nfolds = 5, parallel = TRUE)
```

We plot the cross-validated error curve which has the mean-squared error on the y-axis and $\log(\lambda)$ on the x-axis. It includes the cross-validation curve (red dotted line), and upper and lower standard deviation curves along the λ sequence (error bars). Two selected λ's are indicated by the vertical dotted lines (see below). The numbers at the top of the plot represent the total number of non-zero variables at that value of λ (`df_main` + `df_environment` + `df_interaction`):

```
plot(cvfit)
```



`lambda.min` is the value of λ that gives minimum mean cross-validated error. The other λ saved is `lambda.1se`, which gives the most regularized model such that error is within one standard error of the minimum. We can view the selected λ's and the corresponding coefficients:

```
cvfit[["lambda.min"]]

## [1] 0.2788355

cvfit[["lambda.1se"]]

## [1] 0.4238052
```

The estimated nonzero coefficients at `lambda.1se` and `lambda.min`:

```
predict(cvfit, type = "nonzero", s="lambda.1se") # lambda.1se is the default

##                        1
## (Intercept)   5.42162330
## X1_1         -0.78259292
## X1_2          0.13434743
## X1_3          0.43872182
## X1_4          0.79452833
## X1_5          1.92193021
## X3_1          3.45123833
## X3_2          1.56772288
## X3_3         -1.27179135
## X3_4         -2.27792209
## X3_5         -1.23352137
## X4_1          4.47182924
## X4_2         -2.87488058
## X4_3         -6.65073351
## X4_4         -3.44085202
## X4_5         -0.47032703
## X8_1          0.17507051
## X8_2          0.11619435
## X8_3          0.01783624
## X8_4         -0.10505726
## X8_5         -0.24485431
## X11_1         0.13691003
## X11_2         0.02857999
## X11_3        -0.07324928
## X11_4        -0.20944262
```

```
## X11_5        -0.22648392
## E             1.99575642
## X3_1:E        1.80711590
## X3_2:E        0.82088128
## X3_3:E       -0.66592746
## X3_4:E       -1.19275137
## X3_5:E       -0.64588877
## X4_1:E        8.31249627
## X4_2:E       -5.34399523
## X4_3:E      -12.36277025
## X4_4:E       -6.39605585
## X4_5:E       -0.87427125
```

```
predict(cvfit, type = "nonzero", s = "lambda.min")
```

```
##                        1
## (Intercept)   5.43619834
## X1_1         -1.23832799
## X1_2          0.48736979
## X1_3          0.88915804
## X1_4          1.10813495
## X1_5          2.68123247
## X2_1         -0.21985179
## X2_2         -0.95071069
## X2_3         -0.76066212
## X2_4         -0.17413616
## X2_5         -0.06845895
## X3_1          4.42751615
## X3_2          2.28700197
## X3_3         -1.36817303
## X3_4         -2.56642377
## X3_5         -1.03093978
## X4_1          5.37962120
## X4_2         -3.35568118
## X4_3         -8.07355346
## X4_4         -3.50305791
## X4_5         -0.55031385
## X8_1          0.47925233
## X8_2          0.33992532
## X8_3          0.07652614
## X8_4         -0.28114406
## X8_5         -0.66616044
```

```
## X11_1         0.34768804
## X11_2         0.06274138
## X11_3        -0.20149241
## X11_4        -0.67859653
## X11_5        -0.69589312
## X20_1         0.10225010
## X20_2        -0.03831915
## X20_3        -0.17781683
## X20_4        -0.18564133
## X20_5         0.09741855
## E             2.06410796
## X1_1:E       -0.53459297
## X1_2:E        0.21040020
## X1_3:E        0.38385439
## X1_4:E        0.47838792
## X1_5:E        1.15750273
## X3_1:E        2.37995847
## X3_2:E        1.22935061
## X3_3:E       -0.73544508
## X3_4:E       -1.37955047
## X3_5:E       -0.55416937
## X4_1:E        8.87349228
## X4_2:E       -5.53507578
## X4_3:E      -13.31703694
## X4_4:E       -5.77816841
## X4_5:E       -0.90772296
```

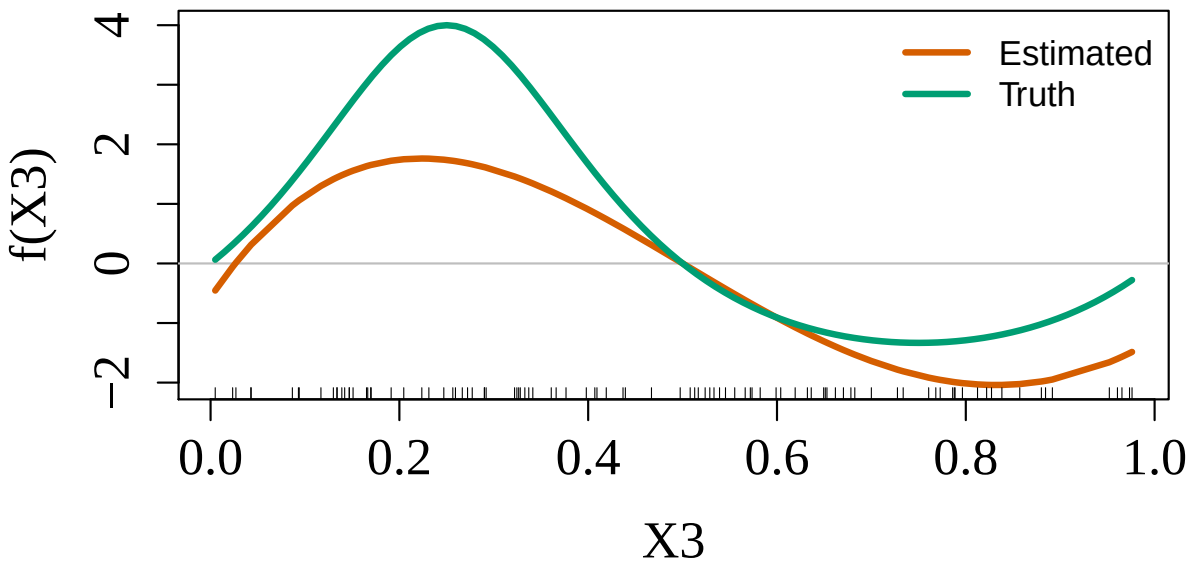
A.3.4 Visualizing the Effect of the Non-linear Terms

B-splines are difficult to interpret. We provide a plotting function to visualize the effect of the non-linear function on the response.

Main Effects

Since we are using simulated data, we also plot the true curve:

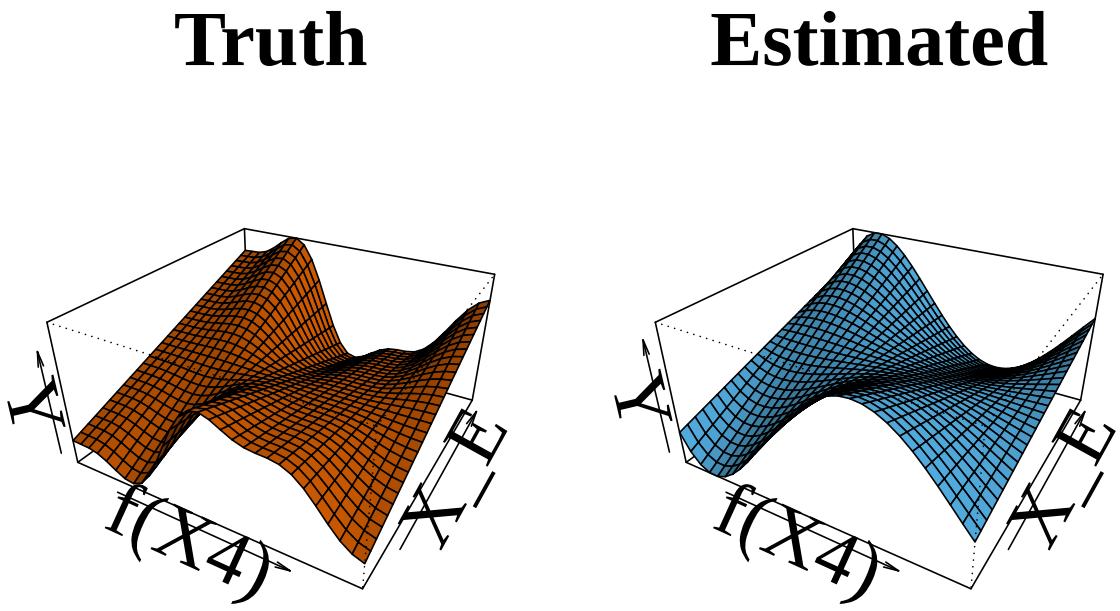
```
plotMain(cvfit$sail.fit, x = sailsim$x, xvar = "X3",
legend.position = "topright",
s = cvfit$lambda.min, f.truth = sailsim$f3)
```



Interaction Effects

Again, since we are using simulated data, we also plot the true interaction:

```
plotInter(cvfit$sail.fit, x = sailsim$x, xvar = "X4",
f.truth = sailsim$f4.inter,
s = cvfit$lambda.min,
title_z = "Estimated")
```



A.3.5 Linear Interactions

The `basis` argument in the `sail` function is very flexible in that it allows you to apply *any* basis expansion to the columns of $\mathbf{X}$. Of course, there might be situations where you do not expect any non-linear main effects or interactions to be present in your data. You can still use the `sail` method to search for linear main effects and interactions. This can be accomplished by specifying an identity map:

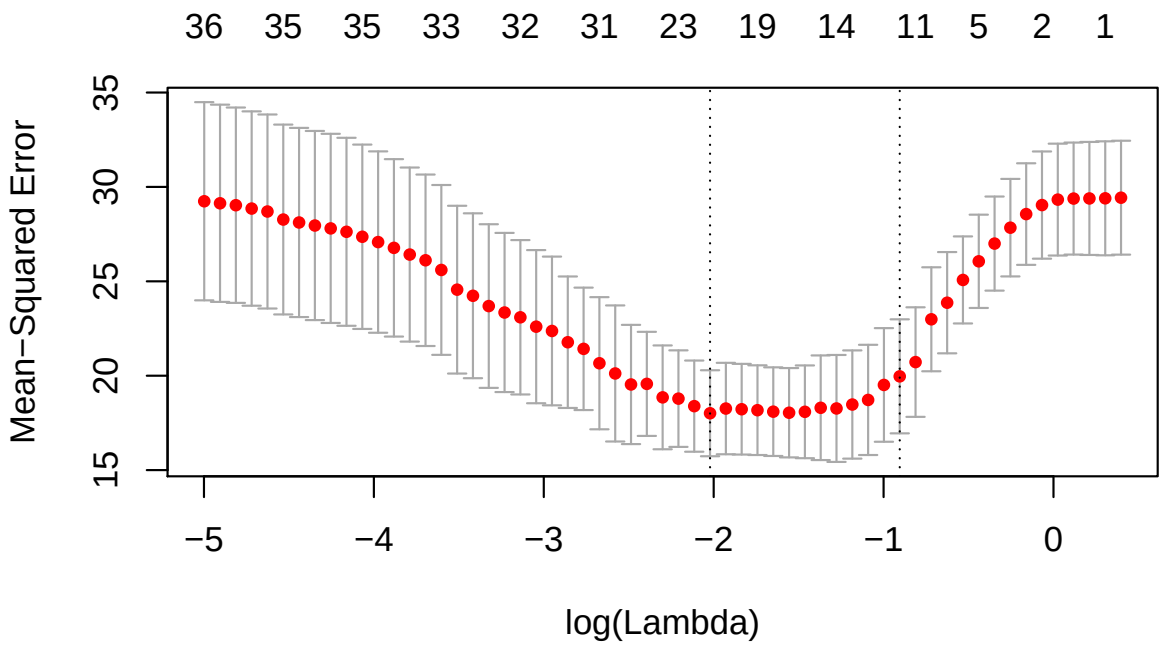
```
f.identity <- function(i) i
```

We then pass this function to the `basis` argument in `cv.sail`:

```
cvfit_linear <- cv.sail(x = sailsim$x, y = sailsim$y, e = sailsim$e,
basis = f.identity, nfolds = 5, parallel = TRUE)
```

Next we plot the cross-validated curve:

```
plot(cvfit_linear)
```



And extract the model at `lambda.min`:

```
predict(cvfit_linear, s = "lambda.min", type = "nonzero")

##                       1
## (Intercept)   5.4385240
## X1_1          2.4568930
## X2_1         -1.6517024
## X3_1         -5.1516226
## X4_1         -7.1356296
## X7_1         -1.1055458
## X8_1         -0.8602620
## X11_1        -2.1990021
## X14_1        -1.9208459
## X16_1         3.2004798
## X18_1         1.0234994
## X20_1        -0.2161983
## E             2.4466403
## X1_1:E       -4.3995787
## X2_1:E       -2.7115733
## X3_1:E       -4.7198552
## X4_1:E      -12.9777976
## X7_1:E        2.3705634
```

```
## X11_1:E      -2.1462985
## X14_1:E      -0.9788318
## X16_1:E       6.9268058
## X18_1:E       1.9005742
```

A.3.6 Applying a different penalty to each predictor

Recall that we consider the following penalized least squares criterion for this problem:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}(Y;\boldsymbol{\theta}) + \lambda(1-\alpha)\left(w_E|\beta_E| + \sum_{j=1}^{p} w_j\|\boldsymbol{\theta}_j\|_2\right) + \lambda\alpha\sum_{j=1}^{p} w_{jE}|\gamma_j| \tag{A.22}$$

The weights w_E, w_j, w_{jE} are by default set to 1 as specified by the `penalty.factor` argument. This argument allows users to apply separate penalty factors to each coefficient. In particular, any variable with `penalty.factor` equal to zero is not penalized at all. This feature can be applied mainly for two reasons:

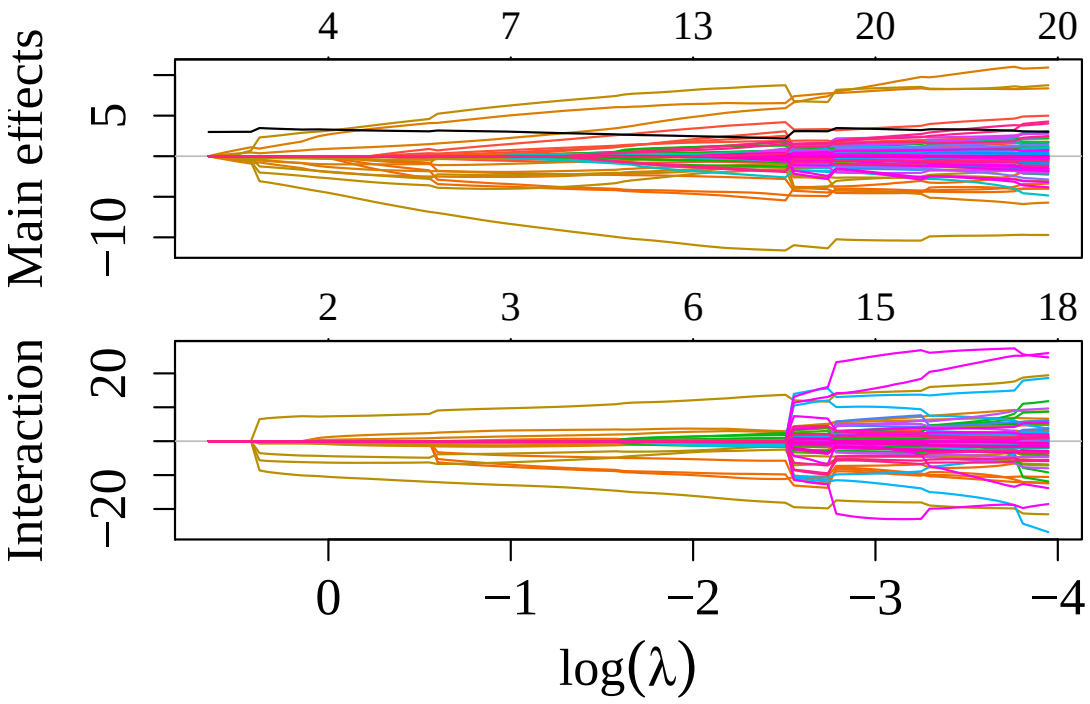
1. Prior knowledge about the importance of certain variables is known. Larger weights will penalize the variable more, while smaller weights will penalize the variable less 2. Allows users to apply the Adaptive `sail`, similar to the Adaptive Lasso

In the following example, we want the environment variable to always be included so we set the first element of `p.fac` to zero. We also want to apply less of a penalty to the main effects for X_2, X_3, X_4:

```
# the weights correspond to E, X1, X2, X3, ... X_p, X1:E, X2:E, ... X_p:E
p.fac <- c(0, 1, 0.4, 0.6, 0.7, rep(1, 2*ncol(sailsim$x) - 4))
```

```
fit_pf <- sail(x = sailsim$x, y = sailsim$y, e = sailsim$e, basis = f.basis,
penalty.factor = p.fac)
```

```
plot(fit_pf)
```



We see from the plot above that the black line (corresponding to the X_E variable with `penalty.factor` equal to zero) is always included in the model.

A.3.7 User-Defined Design Matrix

A limitation of the `sail` method is that the same basis expansion function $f(\cdot)$ is applied to all columns of the predictor matrix $\mathbf{X}$. Being able to automatically select linear vs. nonlinear components was not a focus of our paper, but is an active area of research for main effects only e.g. `gamsel` and `HierBasis`.

However, if the user has some prior knowledge on possible effect relationships, then they can supply their own design matrix. This can be useful for example, when one has a combination of categorical (e.g. gender, race) and continuous variables, but would only like to apply $f(\cdot)$ on the continuous variables. We provide an example below to illustrate this functionality.

We use the simulated dataset `sailsim` provided in our package. We first add a categorical variable `race` to the data:

```
set.seed(1234)
library(sail)
x_df <- as.data.frame(sailsim$x)
x_df$race <- factor(sample(1:2, nrow(x_df), replace = TRUE))
table(x_df$race)

##
##  1  2
## 55 45
```

We then use the `model.matrix` function to create the design matrix. Note that the intercept should not be included, as this is computed internally in the `sail` function. This is why we add 0 to the formula. Notice also the flexibility we can have by including different basis expansions to each predictor:

```
library(splines)
x <- stats::model.matrix(~ 0 +  bs(X1, degree = 5) + bs(X2, degree = 3) + ns(X3, df = 8) +
bs(X4, degree = 6) + X5 + poly(X6,2) + race, data = x_df)
head(x)

##   bs(X1, degree = 5)1 bs(X1, degree = 5)2 bs(X1, degree = 5)3
## 1        0.0001654794         0.003945507        0.0470361237
## 2        0.2470181057         0.345144379        0.2411253263
## 3        0.1299195522         0.007832449        0.0002360971
## 4        0.3808392973         0.121815907        0.0194821217
## 5        0.1737663057         0.014898419        0.0006386822
## 6        0.1184145931         0.281407715        0.3343772913
##   bs(X1, degree = 5)4 bs(X1, degree = 5)5 bs(X2, degree = 3)1
## 1        2.803692e-01        6.684809e-01           0.3272340
## 2        8.422768e-02        1.176866e-02           0.3065738
## 3        3.558391e-06        2.145244e-08           0.1896790
## 4        1.557896e-03        4.983113e-05           0.4100900
## 5        1.368987e-05        1.173746e-07           0.3946500
## 6        1.986587e-01        4.721047e-02           0.3175164
##   bs(X2, degree = 3)2 bs(X2, degree = 3)3 ns(X3, df = 8)1 ns(X3, df = 8)2
## 1          0.41274967         0.173537682      0.06566652               0
## 2          0.04879618         0.002588901      0.00000000               0
## 3          0.01508834         0.000400076      0.00000000               0
```

```
## 4          0.12345871         0.012389196      0.00000000               0
## 5          0.35302552         0.105263760      0.00000000               0
## 6          0.05370432         0.003027827      0.00000000               0
##   ns(X3, df = 8)3 ns(X3, df = 8)4 ns(X3, df = 8)5 ns(X3, df = 8)6
## 1     0.000000000    0.000000e+00       0.0000000   -1.589937e-01
## 2     0.000000000    5.775107e-04       0.3179489    5.395130e-01
## 3     0.000000000    4.989926e-03       0.4147696    4.830810e-01
## 4     0.133404268    6.839146e-01       0.1826811    3.022366e-08
## 5     0.000000000    8.944913e-05       0.2775548    5.564842e-01
## 6     0.001578195    3.415384e-01       0.6070588    4.566909e-02
##   ns(X3, df = 8)7 ns(X3, df = 8)8 bs(X4, degree = 6)1 bs(X4, degree = 6)2
## 1    4.436233e-01   -2.846296e-01        0.1820918880        0.3088147022
## 2    1.732713e-01   -3.131078e-02        0.0120101010        0.0000608354
## 3    1.434410e-01   -4.628144e-02        0.0002900763        0.0044075535
## 4    7.673343e-09   -4.923233e-09        0.2978877432        0.0579746877
## 5    1.863219e-01   -2.045032e-02        0.0114895681        0.0645689076
## 6    1.159471e-02   -7.439189e-03        0.0102152807        0.0595722132
##   bs(X4, degree = 6)3 bs(X4, degree = 6)4 bs(X4, degree = 6)5
## 1        2.793213e-01        1.421126e-01        3.856204e-02
## 2        1.643482e-07        2.497444e-10        2.024070e-13
## 3        3.571755e-02        1.628127e-01        3.958163e-01
## 4        6.017595e-03        3.513419e-04        1.094046e-05
## 5        1.935272e-01        3.262743e-01        2.933747e-01
## 6        1.852831e-01        3.241534e-01        3.024572e-01
##   bs(X4, degree = 6)6         X5 poly(X6, 2)1 poly(X6, 2)2 race1 race2
## 1        4.359896e-03 0.51332996  -0.13705545   0.09851639     1     0
## 2        6.835086e-17 0.02643863   0.18835303   0.22584415     0     1
## 3        4.009478e-01 0.76746637  -0.15841216   0.16140597     0     1
## 4        1.419483e-07 0.69077618  -0.03664279  -0.07954100     0     1
## 5        1.099135e-01 0.27718210   0.13128945   0.05620199     0     1
## 6        1.175889e-01 0.48384748   0.08486354  -0.03559388     0     1
```

One benefit of using `stats::model.matrix` is that it returns the group membership as an attribute:

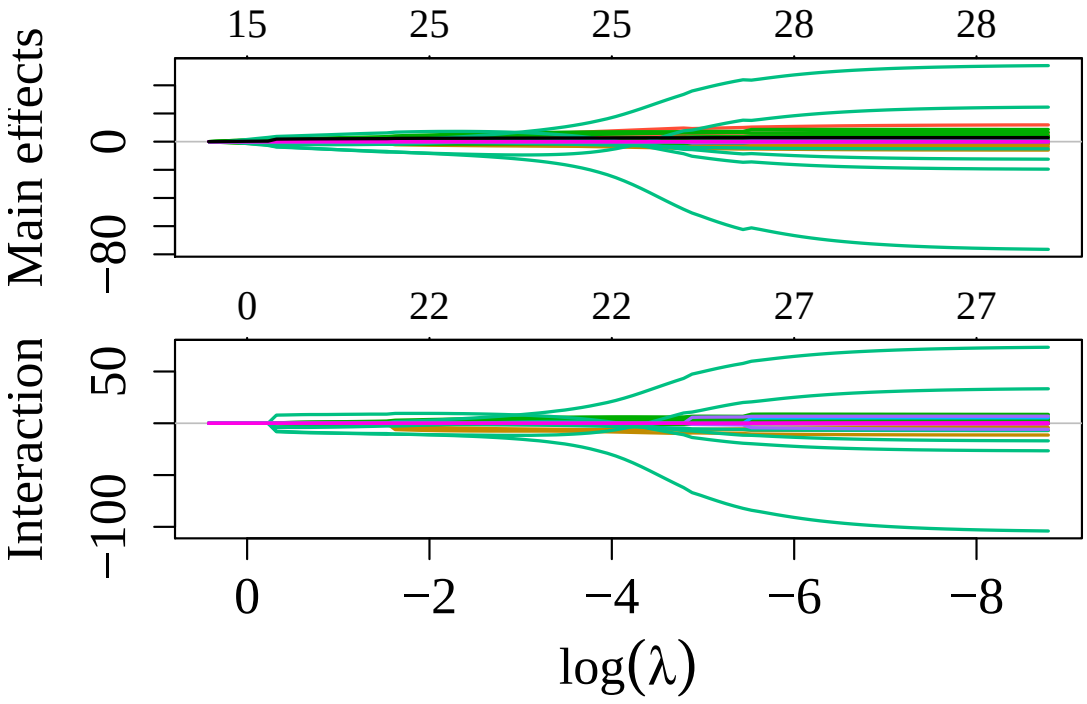
```
attr(x, "assign")

##  [1] 1 1 1 1 1 2 2 2 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 6 6 7 7
```

The group membership must be supplied to the `sail` function. This information is needed for the group lasso penalty, which will select the whole group as zero or non-zero.

Fit the `sail` Model

We need to set the argument `expand = FALSE` and provide the group membership. The first element of the group membership corresponds to the first column of `x`, the second element to the second column of `x`, and so on.

We can plot the solution path for both main effects and interactions using the `plot` method for objects of class `sail`:

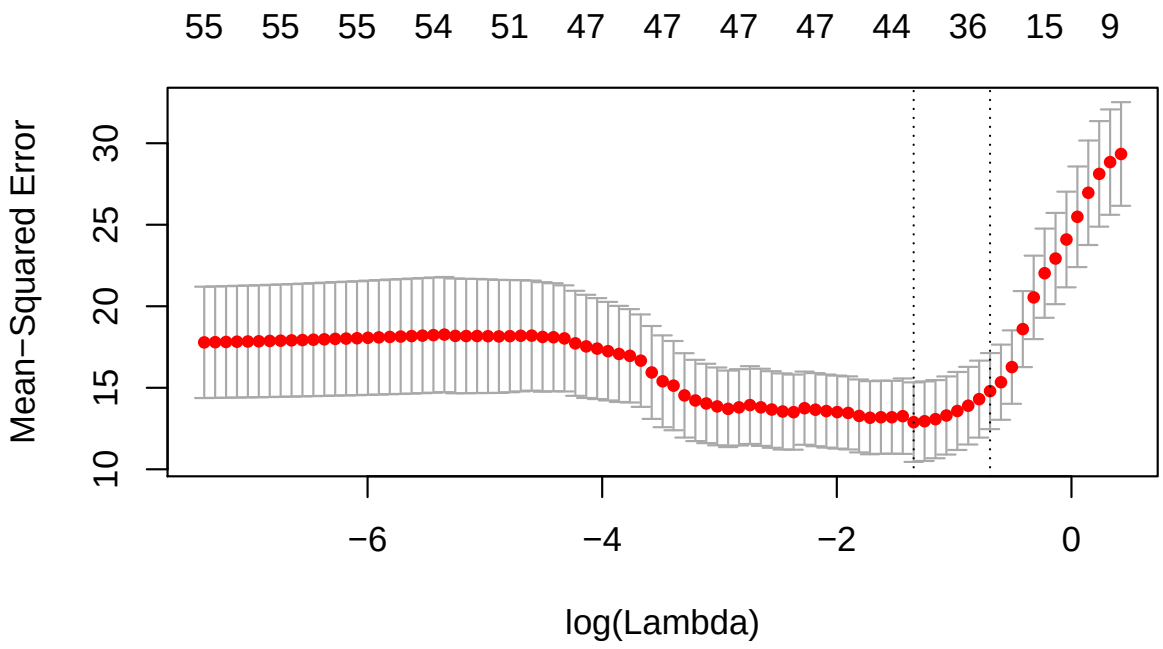


In this instance, since we provided a user-defined design matrix and 'expand = FALSE', the numbers at the top of the plot represent the total number of non-zero coefficients.

Find the Optimal Value for λ

We can use cross-validation to find the optimal value of lambda:

We can plot the cross-validated mean squared error as a function of lambda:



The estimated non-zero coefficients at `lambda.1se`:

```
##                                1
## (Intercept)            5.427451767
## bs(X1, degree = 5)1   -0.525424522
## bs(X1, degree = 5)2    0.052358374
## bs(X1, degree = 5)3    0.271119758
## bs(X1, degree = 5)4    0.554195548
## bs(X1, degree = 5)5    1.330393115
## ns(X3, df = 8)1        2.349455333
## ns(X3, df = 8)2        2.089923982
## ns(X3, df = 8)3        0.666828606
## ns(X3, df = 8)4       -1.200690572
## ns(X3, df = 8)5       -1.662360501
## ns(X3, df = 8)6       -1.365480040
## ns(X3, df = 8)7        0.516186563
## ns(X3, df = 8)8       -1.215186213
## bs(X4, degree = 6)1    4.466614577
## bs(X4, degree = 6)2   -0.252832683
## bs(X4, degree = 6)3   -4.706674900
## bs(X4, degree = 6)4   -4.868782936
## bs(X4, degree = 6)5   -2.105379737
```

```
## bs(X4, degree = 6)6   -0.213392506
## race1                  0.006726548
## race2                 -0.006726548
## E                      1.963105161
## ns(X3, df = 8)1:E      1.170879214
## ns(X3, df = 8)2:E      1.041538657
## ns(X3, df = 8)3:E      0.332322025
## ns(X3, df = 8)4:E     -0.598378532
## ns(X3, df = 8)5:E     -0.828457273
## ns(X3, df = 8)6:E     -0.680503338
## ns(X3, df = 8)7:E      0.257247758
## ns(X3, df = 8)8:E     -0.605602609
## bs(X4, degree = 6)1:E  8.430067510
## bs(X4, degree = 6)2:E -0.477183905
## bs(X4, degree = 6)3:E -8.883145495
## bs(X4, degree = 6)4:E -9.189100187
## bs(X4, degree = 6)5:E -3.973589620
## bs(X4, degree = 6)6:E -0.402746465
```

Appendix B

Supplemental Methods and Simulation Results for Chapter 4

B.1 Block Coordinate Descent Algorithm

We use a general purpose block coordinate descent algorithm (CGD) (Tseng & Yun, 2009) to solve (4.15). Following Tseng and Yun Tseng & Yun (2009), the complete CGD algorithm is given by Algorithm 9.

The Armijo rule is defined as follows (Tseng & Yun, 2009):

Choose $\alpha_{init}^{(k)} > 0$ and let $\alpha^{(k)}$ be the largest element of $\left\{\alpha_{init}^{k}\delta^{r}\right\}_{r=0,1,2,\ldots}$ satisfying

$$Q_{\lambda}(\Theta_{j}^{(k)} + \alpha^{(k)}d^{(k)}) \leq Q_{\lambda}(\Theta_{j}^{(k)}) + \alpha^{(k)}\varrho\Delta^{(k)} \tag{B.5}$$

where $0 < \delta < 1$, $0 < \varrho < 1$, $0 \leq \gamma < 1$ and

$$\Delta^{(k)} \coloneqq \nabla f(\Theta_{j}^{(k)})d^{(k)} + \gamma(d^{(k)})^{2}H_{jj}^{(k)} + \lambda P(\Theta_{j}^{(k)} + d^{(k)}) - \lambda P(\Theta^{(k)}) \tag{B.6}$$

Algorithm 9 Coordinate Gradient Descent Algorithm to solve (4.15)

Set the iteration counter $k \leftarrow 0$ and choose initial values for the parameter vector $\boldsymbol{\Theta}^{(0)}$

repeat

Approximate the Hessian $\nabla^2 f(\boldsymbol{\Theta}^{(k)})$ by a symmetric matix $H^{(k)}$:

$$H^{(k)} = \text{diag}\left[\min\left\{\max\left\{\left[\nabla^2 f(\boldsymbol{\Theta}^{(k)})\right]_{jj}, c_{min}\right\} c_{max}\right\}\right]_{j=1,\ldots,p} \tag{B.1}$$

for $j = 1, \ldots, p$ **do**

Solve the descent direction $d^{(k)} := d_{H^{(k)}}(\Theta_j^{(k)})$

if $\Theta_j^{(k)} \in \{\beta_1, \ldots, \beta_p\}$ **then**

$$d_{H^{(k)}}(\Theta_j^{(k)}) \leftarrow \arg\min_d \left\{\nabla f(\Theta_j^{(k)})d + \frac{1}{2}d^2 H_{jj}^{(k)} + \lambda P(\Theta_j^{(k)} + d)\right\} \tag{B.2}$$

Choose a stepsize

$$\alpha_j^{(k)} \leftarrow \text{line search given by the Armijo rule}$$

Update

$$\widehat{\Theta}_j^{(k+1)} \leftarrow \widehat{\Theta}_j^{(k)} + \alpha_j^{(k)} d^{(k)}$$

Update

$$\widehat{\eta}^{(k+1)} \leftarrow \arg\min_\eta \frac{1}{2}\sum_{i=1}^{N_T} \log(1 + \eta(\Lambda_i - 1)) + \frac{1}{2\sigma^{2\,(k)}} \sum_{i=1}^{N_T} \frac{\left(\widetilde{Y}_i - \sum_{j=0}^p \widetilde{X}_{ij+1}\beta_j^{(k+1)}\right)^2}{1 + \eta(\Lambda_i - 1)} \tag{B.3}$$

Update

$$\widehat{\sigma^2}^{\,(k+1)} \leftarrow \frac{1}{N_T}\sum_{i=1}^{N_T} \frac{\left(\widetilde{Y}_i - \sum_{j=0}^p \widetilde{X}_{ij+1}\beta_j^{(k+1)}\right)^2}{1 + \eta^{(k+1)}(\Lambda_i - 1)} \tag{B.4}$$

$k \leftarrow k + 1$

until convergence criterion is satisfied

Common choices for the constants are $\delta = 0.1$, $\varrho = 0.001$, $\gamma = 0$, $\alpha_{init}^{(k)} = 1$ for all k (Schelldorfer et al., 2011).

Below we detail the specifics of Algorithm 9 for the ℓ_1 penalty.

B.1.1 ℓ_1 penalty

The objective function is given by

$$Q_\lambda(\boldsymbol{\Theta}) = f(\boldsymbol{\Theta}) + \lambda|\boldsymbol{\beta}| \tag{B.7}$$

Descent Direction

For simplicity, we remove the iteration counter (k) from the derivation below.

For $\Theta_j^{(k)} \in \{\beta_1, \ldots, \beta_p\}$, let

$$d_H(\Theta_j) = \arg\min_d G(d) \tag{B.8}$$

where

$$G(d) = \nabla f(\Theta_j)d + \frac{1}{2}d^2 H_{jj} + \lambda|\Theta_j + d|$$

Since $G(d)$ is not differentiable at $-\Theta_j$, we calculate the subdifferential $\partial G(d)$ and search for d with $0 \in \partial G(d)$:

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} + \lambda u \tag{B.9}$$

where

$$u = \begin{cases} 1 & \text{if} \quad d > -\Theta_j \\ -1 & \text{if} \quad d < -\Theta_j \\ [-1, 1] & \text{if} \quad d = \Theta_j \end{cases} \tag{B.10}$$

We consider each of the three cases in (B.9) below

1. $d > -\Theta_j$

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} + \lambda = 0$$
$$d = \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}$$

Since $\lambda > 0$ and $H_{jj} > 0$, we have

$$\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}} > \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}} = d \overset{\text{def}}{>} -\Theta_j$$

The solution can be written compactly as

$$d = \text{mid}\left\{\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}, -\Theta_j, \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}\right\}$$

where mid $\{a, b, c\}$ denotes the median (mid-point) of a, b, c (Tseng & Yun, 2009).

2. $d < -\Theta_j$

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} - \lambda = 0$$
$$d = \frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}$$

Since $\lambda > 0$ and $H_{jj} > 0$, we have

$$\frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}} < \frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}} = d \overset{\text{def}}{<} -\Theta_j$$

Again, the solution can be written compactly as

$$d = \text{mid}\left\{\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}, -\Theta_j, \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}\right\}$$

3. $d_j = -\Theta_j$

There exists $u \in [-1, 1]$ such that

$$\partial G(d) = \nabla f(\Theta_j) + dH_{jj} + \lambda u = 0$$
$$d = \frac{-(\nabla f(\Theta_j) + \lambda u)}{H_{jj}}$$

For $-1 \leq u \leq 1$, $\lambda > 0$ and $H_{jj} > 0$ we have

$$\frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}} \leq d \stackrel{\text{def}}{=} -\Theta_j \leq \frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}$$

The solution can again be written compactly as

$$d = \text{mid}\left\{\frac{-(\nabla f(\Theta_j) - \lambda)}{H_{jj}}, -\Theta_j, \frac{-(\nabla f(\Theta_j) + \lambda)}{H_{jj}}\right\}$$

We see all three cases lead to the same solution for (B.8). Therefore the descent direction for $\Theta_j^{(k)} \in \{\beta_1, \ldots, \beta_p\}$ for the ℓ_1 penalty is given by

$$d = \text{mid}\left\{\frac{-(\nabla f(\beta_j) - \lambda)}{H_{jj}}, -\beta_j, \frac{-(\nabla f(\beta_j) + \lambda)}{H_{jj}}\right\} \tag{B.11}$$

Solution for the β parameter

If the Hessian $\nabla^2 f(\boldsymbol{\Theta}^{(k)}) > 0$ then $H^{(k)}$ defined in (B.1) is equal to $\nabla^2 f(\boldsymbol{\Theta}^{(k)})$. Using $\alpha_{init} = 1$, the largest element of $\left\{\alpha_{init}^{(k)}\delta^r\right\}_{r=0,1,2,\ldots}$ satisfying the Armijo Rule inequality is reached for $\alpha^{(k)} = \alpha_{init}^{(k)}\delta^0 = 1$. The Armijo rule update for the $\boldsymbol{\beta}$ parameter is then given by

$$\beta_j^{(k+1)} \leftarrow \beta_j^{(k)} + d^{(k)}, \qquad j = 1, \ldots, p \tag{B.12}$$

Substituting the descent direction given by (B.11) into (B.12) we get

$$\beta_j^{(k+1)} = \text{mid}\left\{\beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) - \lambda)}{H_{jj}}, 0, \beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) + \lambda)}{H_{jj}}\right\} \tag{B.13}$$

We can further simplify this expression. Let

$$w_i \coloneqq \frac{1}{\sigma^2\left(1 + \eta(\Lambda_i - 1)\right)} \tag{B.14}$$

.

Re-write the part depending on $\boldsymbol{\beta}$ of the negative log-likelihood in (4.13) as

$$g(\boldsymbol{\beta}^{(k)}) = \frac{1}{2}\sum_{i=1}^{N_T} w_i \left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell}\beta_\ell^{(k)} - \widetilde{X}_{ij}\beta_j^{(k)}\right)^2 \tag{B.15}$$

The gradient and Hessian are given by

$$\nabla f(\beta_j^{(k)}) \coloneqq \frac{\partial}{\partial \beta_j^{(k)}} g(\boldsymbol{\beta}^{(k)}) = -\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}\left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell}\beta_\ell^{(k)} - \widetilde{X}_{ij}\beta_j^{(k)}\right) \tag{B.16}$$

$$H_{jj} \coloneqq \frac{\partial^2}{\partial {\beta_j^{(k)}}^2} g(\boldsymbol{\beta}^{(k)}) = \sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2 \tag{B.17}$$

Substituting (B.16) and (B.17) into $\beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)}) - \lambda)}{H_{jj}}$

$$\begin{aligned}
&\beta_j^{(k)} + \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}\left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell}\beta_\ell^{(k)} - \widetilde{X}_{ij}\beta_j^{(k)}\right) + \lambda}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} \\
&= \beta_j^{(k)} + \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}\left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell}\beta_\ell^{(k)}\right) + \lambda}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} - \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2 \beta_j^{(k)}}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} \\
&= \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}\left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell}\beta_\ell^{(k)}\right) + \lambda}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2}
\end{aligned} \tag{B.18}$$

Similarly, substituting (B.16) and (B.17) in $\beta_j^{(k)} + \frac{-(\nabla f(\beta_j^{(k)})+\lambda)}{H_{jj}}$ we get

$$\frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell} \beta_\ell^{(k)} \right) - \lambda}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} \tag{B.19}$$

Finally, substituting (B.18) and (B.19) into (B.13) we get

$$\begin{aligned} \beta_j^{(k+1)} &= \text{mid} \left\{ \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell} \beta_\ell^{(k)} \right) - \lambda}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2}, 0, \frac{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell} \beta_\ell^{(k)} \right) + \lambda}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} \right\} \\ &= \frac{\mathcal{S}_\lambda \left(\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij} \left(\widetilde{Y}_i - \sum_{\ell \neq j} \widetilde{X}_{i\ell} \beta_\ell^{(k)} \right) \right)}{\sum_{i=1}^{N_T} w_i \widetilde{X}_{ij}^2} \end{aligned} \tag{B.20}$$

Where $\mathcal{S}_\lambda(x)$ is the soft-thresholding operator

$$\mathcal{S}_\lambda(x) = \text{sign}(x)(|x| - \lambda)_+$$

$\text{sign}(x)$ is the signum function

$$\text{sign}(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0 \end{cases}$$

and $(x)_+ = \max(x, 0)$.

B.2 Additional Simulation Results

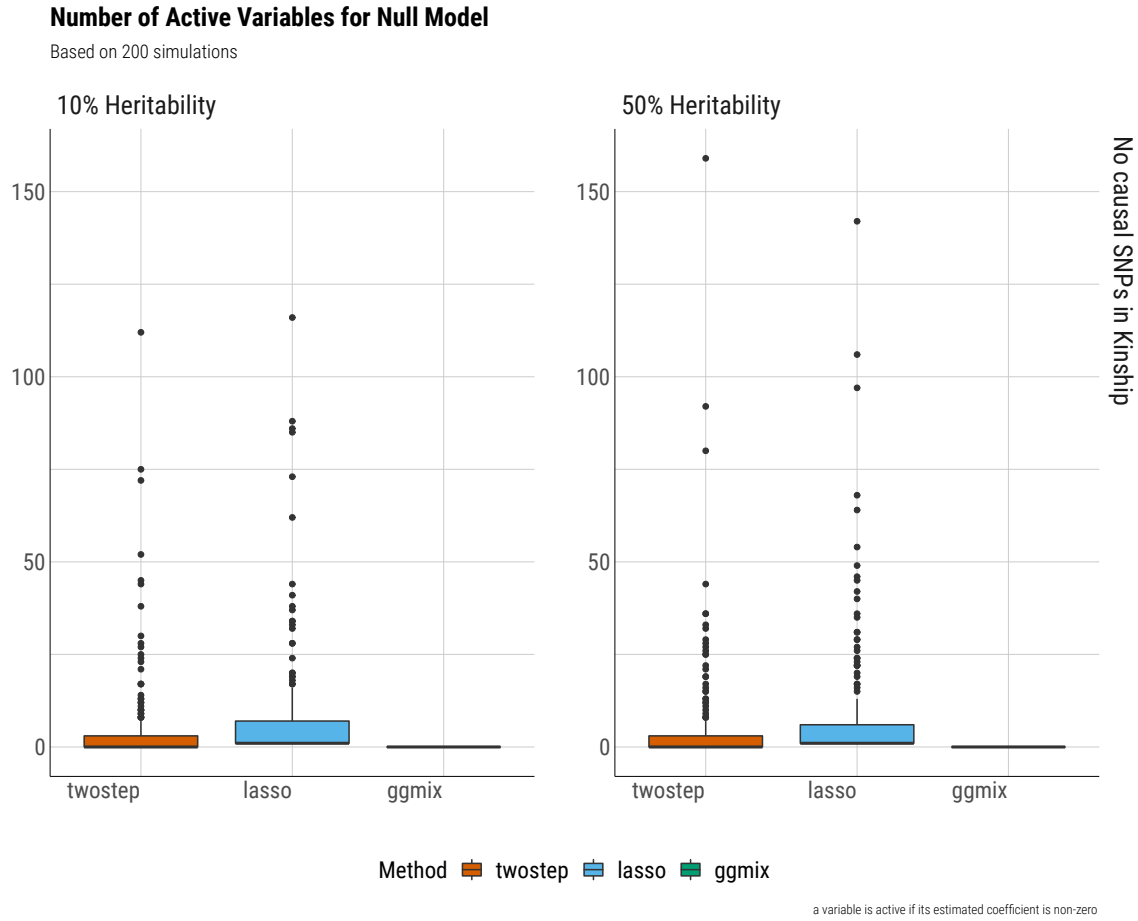


Figure B.1: Boxplots of the number of active variables from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ for the null model ($c = 0$).

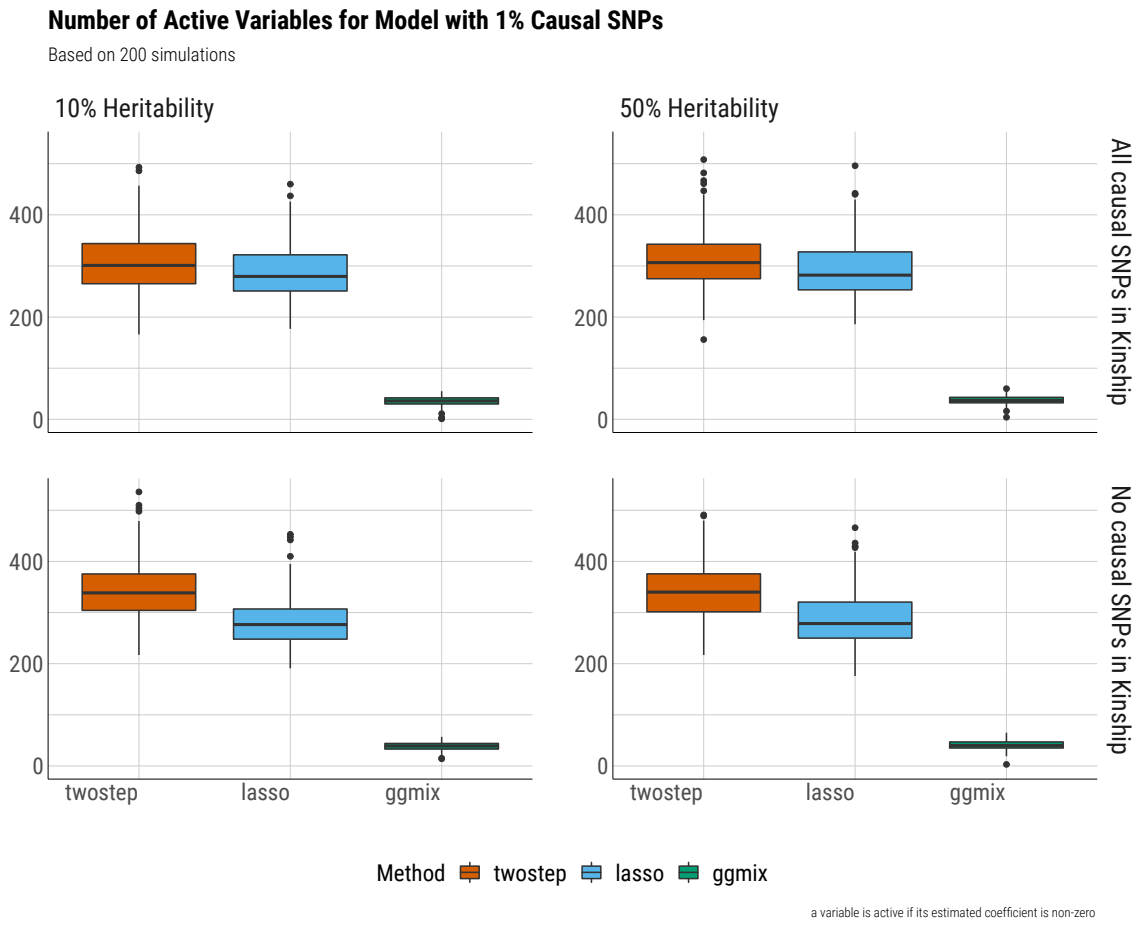


Figure B.2: Boxplots of the number of active variables from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ and number of causal SNPs that were included in the calculation of the kinship matrix for the model with 1% causal SNPs ($c = 0.01$).

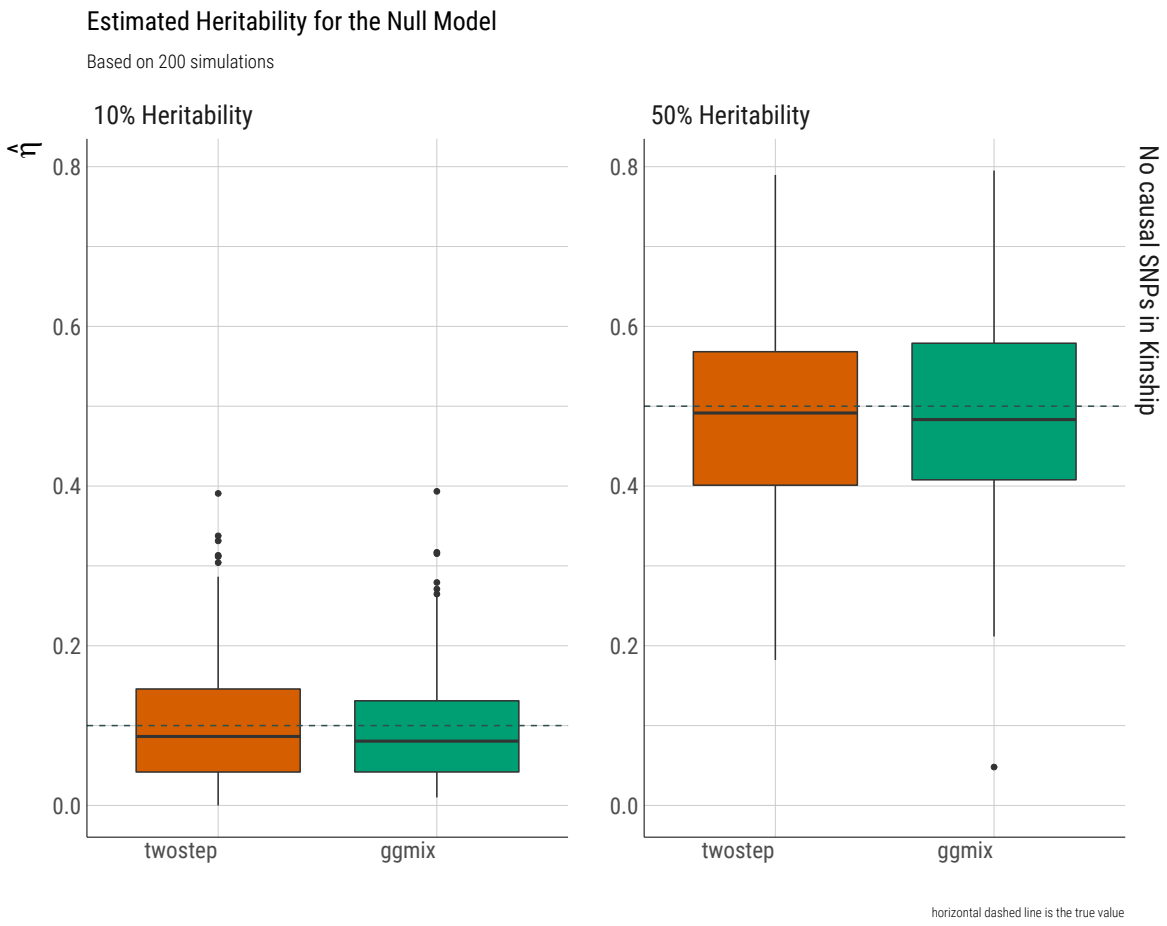


Figure B.3: Boxplots of the heritability estimate $\hat{\eta}$ from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ for the null model ($c = 0$).

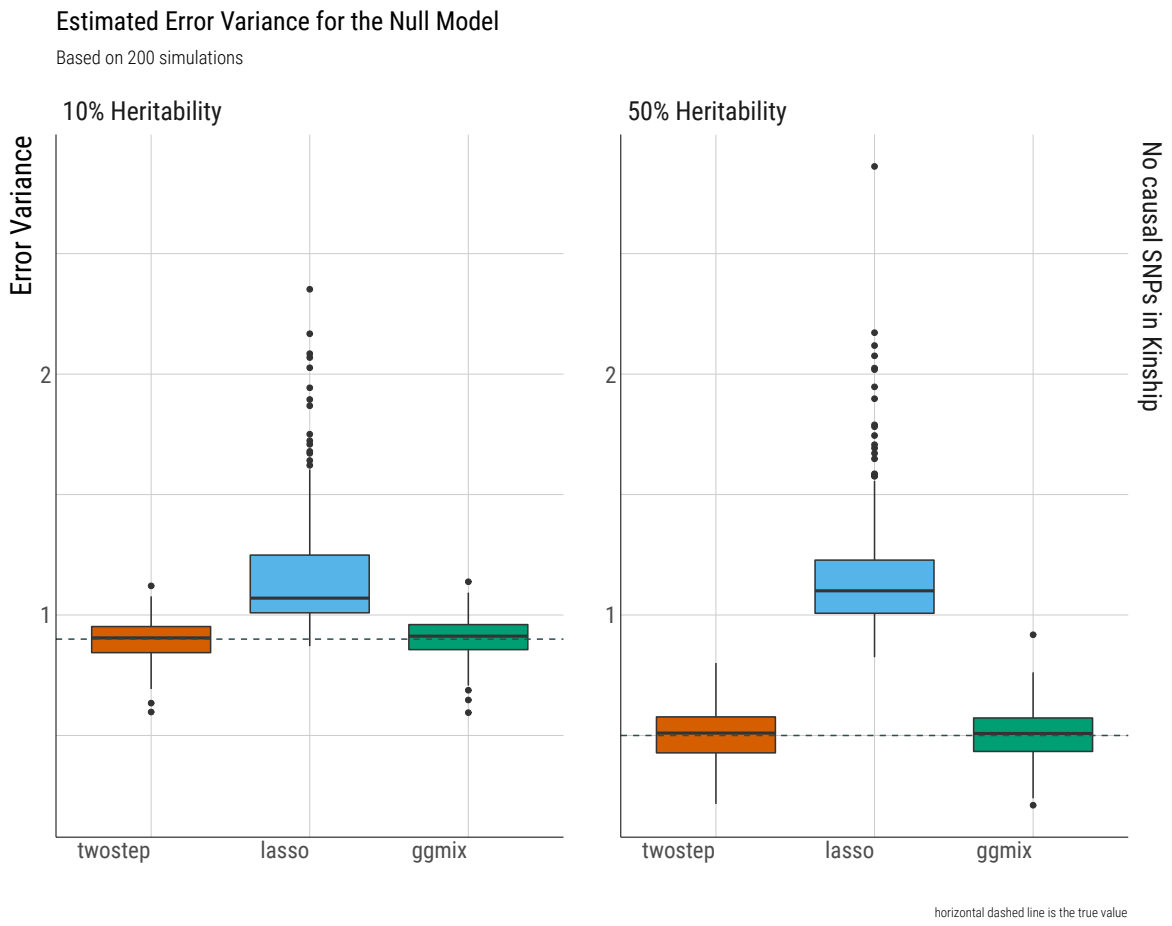


Figure B.4: Boxplots of the estimated error variance from 200 simulations by the true heritability $\eta = \{10\%, 50\%\}$ for the null model ($c = 0$).

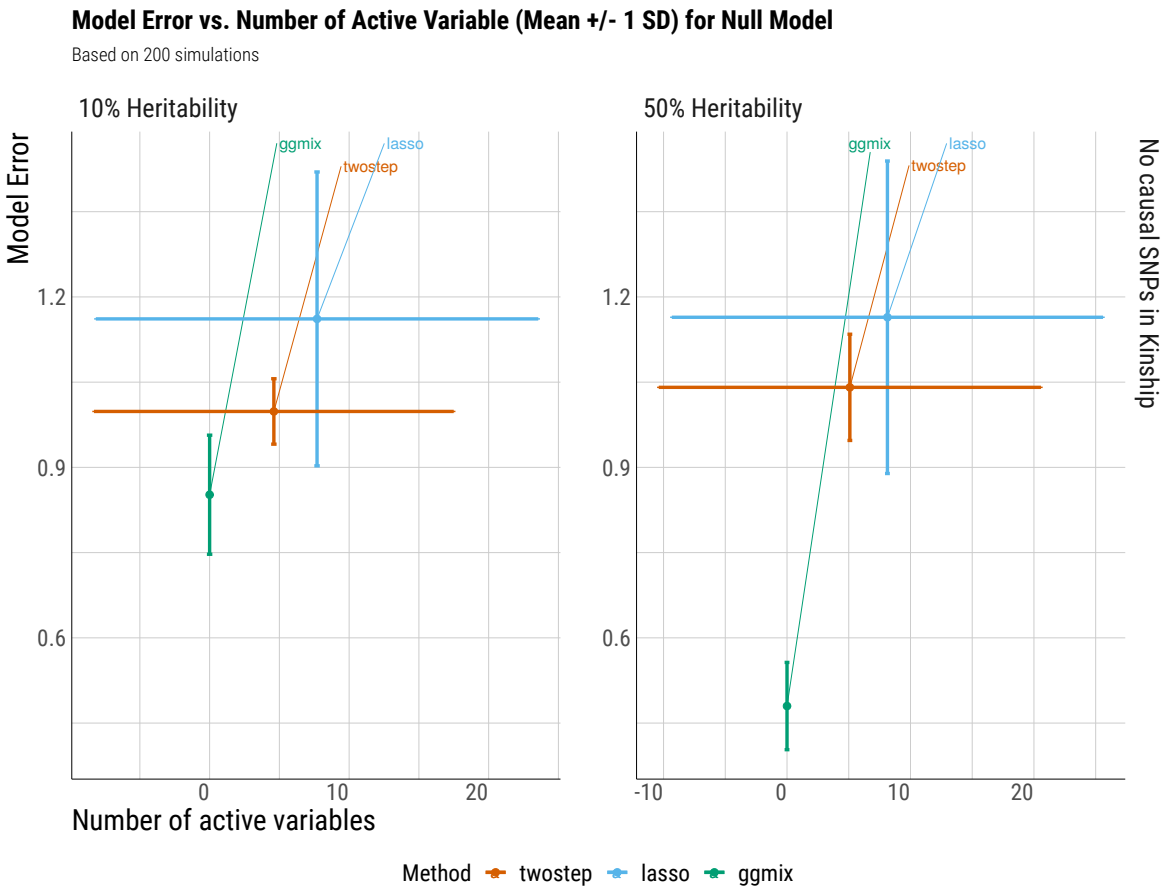


Figure B.5: Means ± 1 standard deviation of the model error vs. the number of active variables by the true heritability $\eta = \{10\%, 50\%\}$ for the null model ($c = 0$).

B.3 ggmix Package Showcase

In this section we briefly introduce the freely available and open source ggmix package in R. More comprehensive documentation is available at https://sahirbhatnagar.com/ggmix. Note that this entire section is reproducible; the code and text are combined in an .Rnw[1] file and compiled using knitr (Xie, 2015).

B.3.1 Installation

The package can be installed from GitHub via

```
install.packages("pacman")
pacman::p_load_gh('sahirbhatnagar/ggmix')
```

To showcase the main functions in ggmix, we will use the simulated data which ships with the package and can be loaded via:

```
library(ggmix)
data("admixed")
names(admixed)

##  [1] "y"               "x"               "causal"
##  [4] "beta"            "kin"             "Xkinship"
##  [7] "not_causal"      "causal_positive" "causal_negative"
## [10] "x_lasso"
```

For details on how this data was simulated, see help(admixed).

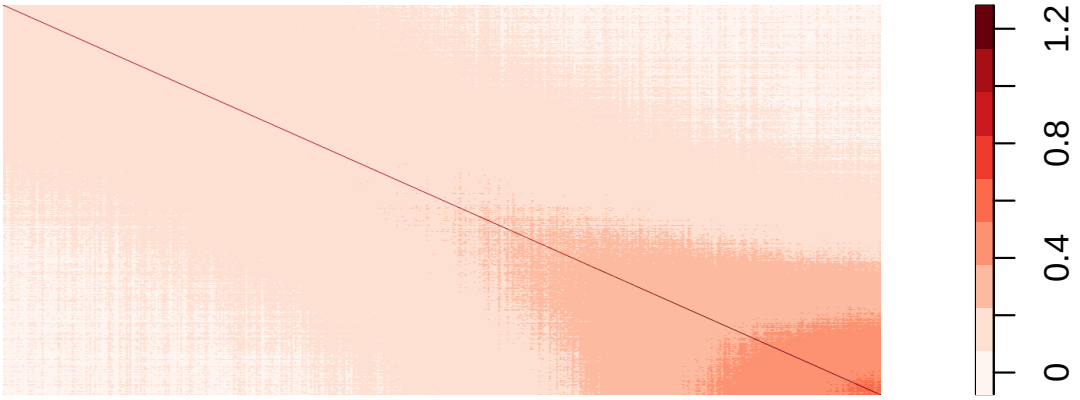
There are three basic inputs that ggmix needs:

1. Y: a continuous response variable
2. X: a matrix of covariates of dimension $N \times p$ where N is the sample size and p is the number of covariates
3. $\mathbf{\Phi}$: a kinship matrix

[1] scripts available at https://github.com/sahirbhatnagar/ggmix/tree/master/manuscript

We can visualize the kinship matrix in the `admixed` data using the `popkin` package:

```
# need to install the package if you don't have it
# pacman::p_load_gh('StoreyLab/popkin')
popkin::plotPopkin(admixed$kin)
```



B.3.2 Fit the linear mixed model with Lasso Penalty

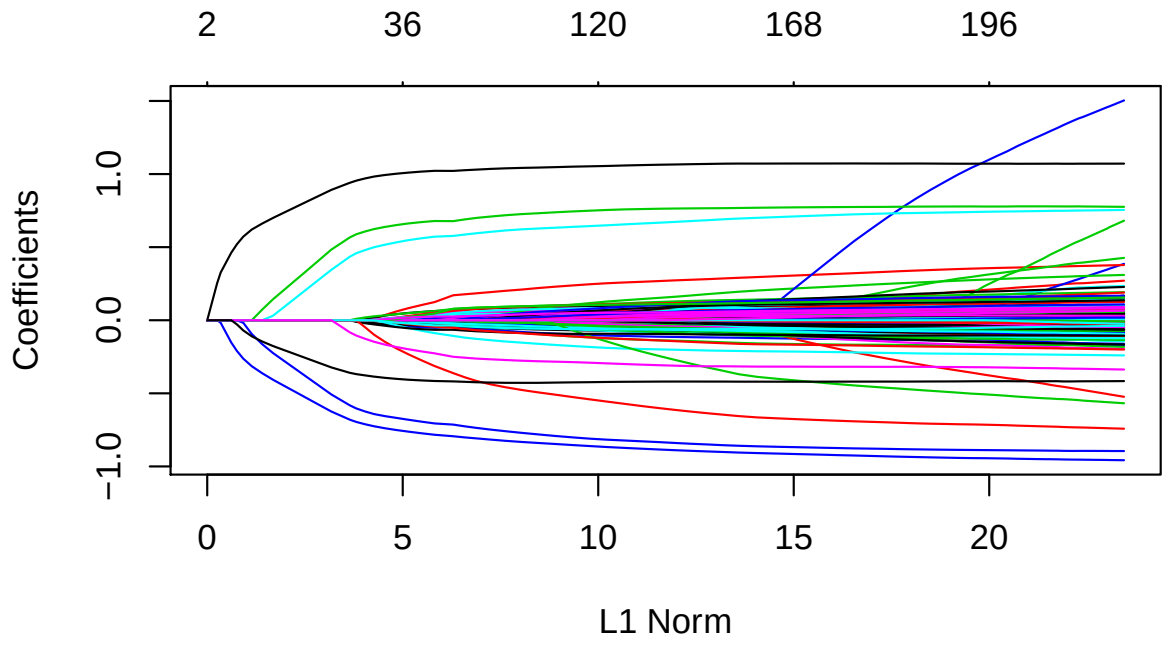
We will use the most basic call to the main function of this package, which is called `ggmix`. This function will by default fit a L_1 penalized linear mixed model (LMM) for 100 distinct values of the tuning parameter λ. It will choose its own sequence:

```
fit <- ggmix(x = admixed$x, y = admixed$y, kinship = admixed$kin)
names(fit)

##  [1] "result"       "ggmix_object" "n_design"     "p_design"
##  [5] "lambda"       "coef"         "b0"           "beta"
##  [9] "df"           "eta"          "sigma2"       "nlambda"
## [13] "cov_names"    "call"

class(fit)
```

```
## [1] "lassofullrank" "ggmix_fit"
```

We can see the solution path for each variable by calling the `plot` method for objects of class `ggmix_fit`:

```
plot(fit)
```



We can also get the coefficients for given value(s) of lambda using the `coef` method for objects of class `ggmix_fit`:

```
# only the first 5 coefficients printed here for brevity
coef(fit, s = c(0.1,0.02))[1:5, ]

## 5 x 2 Matrix of class "dgeMatrix"
##                      1           2
## (Intercept) -0.3824525 -0.030227753
## X62          0.0000000  0.000000000
## X185         0.0000000  0.001444670
## X371         0.0000000  0.009513604
## X420         0.0000000  0.000000000
```

Here, `s` specifies the value(s) of λ at which the extraction is made. The function uses linear

interpolation to make predictions for values of `s` that do not coincide with the lambda sequence used in the fitting algorithm.

We can also get predictions ($X\widehat{\boldsymbol{\beta}}$) using the `predict` method for objects of class `ggmix_fit`:

```
# need to provide x to the predict function
# predict for the first 5 subjects
predict(fit, s = c(0.1,0.02), newx = admixed$x[1:5,])

##               1          2
## id1 -1.19165061 -1.3123396
## id2 -0.02913052  0.3885921
## id3 -2.00084875 -2.6460045
## id4 -0.37255277 -0.9542455
## id5 -1.03967831 -2.1377274
```

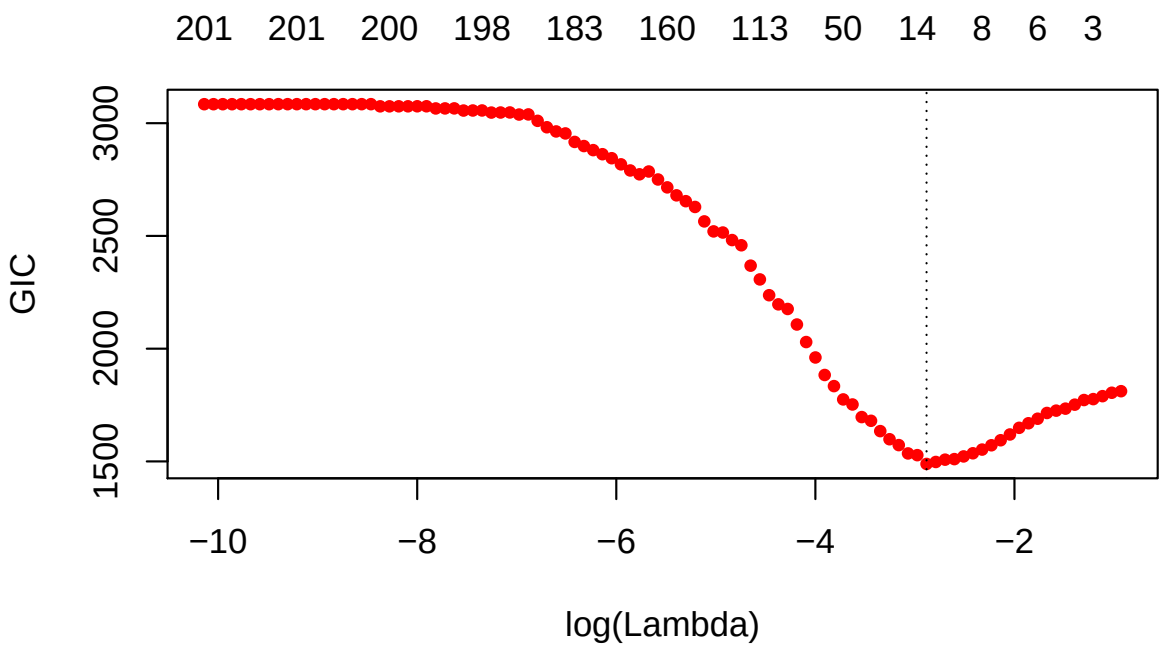
B.3.3 Find the Optimal Value of the Tuning Parameter

We use the Generalized Information Criterion (GIC) to select the optimal value for λ. The default is $a_n = log(log(n)) * log(p)$ which corresponds to a high-dimensional BIC (HDBIC):

```
# pass the fitted object from ggmix to the gic function:
hdbic <- gic(fit)
class(hdbic)

## [1] "ggmix_gic"     "lassofullrank" "ggmix_fit"

# we can also fit the BIC by specifying the an argument
bicfit <- gic(fit, an = log(length(admixed$y)))
```

We can plot the HDBIC values against $\log(\lambda)$ using the `plot` method for objects of class `ggmix_gic`:

```
plot(hdbic)
```

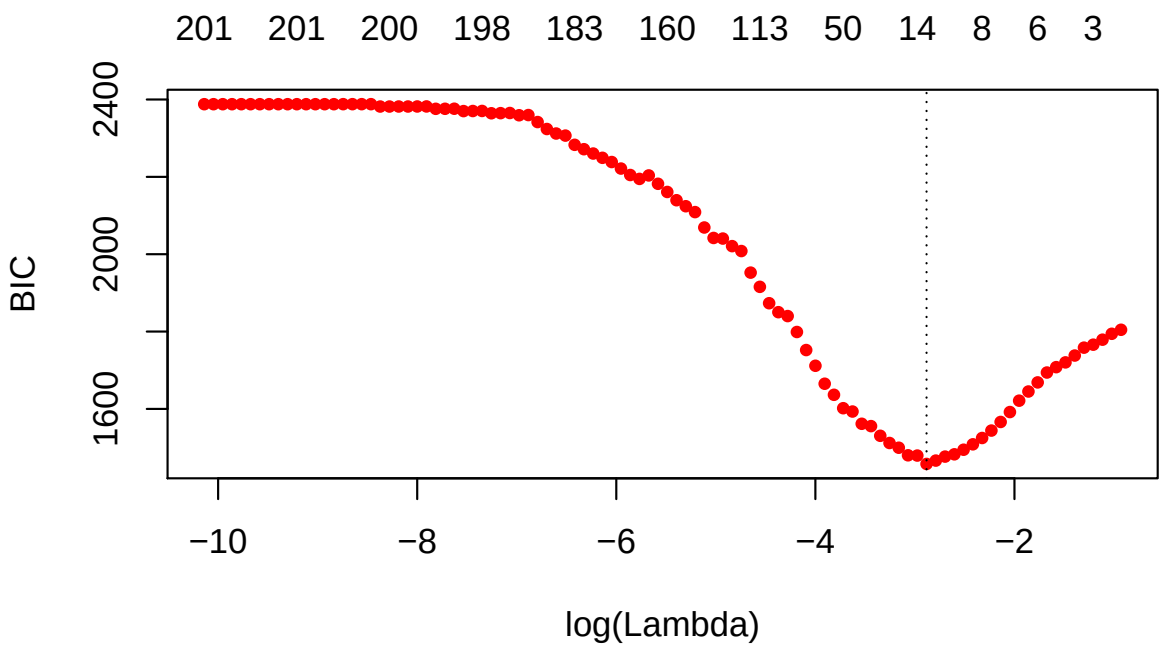


The optimal value for λ according to the HDBIC, i.e., the λ that leads to the minium HDBIC is:

```
hdbic[["lambda.min"]]

## [1] 0.05596623
```

We can also plot the BIC results:

```
plot(bicfit, ylab = "BIC")
```




```
bicfit[["lambda.min"]]

## [1] 0.05596623
```

B.3.4 Get Coefficients Corresponding to Optimal Model

We can use the object outputted by the `gic` function to extract the coefficients corresponding to the selected model using the `coef` method for objects of class `ggmix_gic`:

```
coef(hdbic)[1:5, , drop = FALSE]

## 5 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) -0.2668419
## X62          .
## X185         .
## X371         .
## X420         .
```

We can also extract just the nonzero coefficients which also provide the estimated variance components η and σ^2:

```
coef(hdbic, type = "nonzero")

##                       1
## (Intercept) -0.26684191
## X336        -0.67986393
## X7638        0.43403365
## X1536        0.93994982
## X1943        0.56600730
## X2849       -0.58157979
## X56         -0.08244685
## X4106       -0.35939830
## eta          0.26746240
## sigma2       0.98694300
```

We can also make predictions from the `hdbic` object, which by default will use the model corresponding to the optimal tuning parameter:

```
predict(hdbic, newx = admixed$x[1:5,])

##              1
## id1 -1.3061041
## id2  0.2991654
## id3 -2.3453664
## id4 -0.4486012
## id5 -1.3895793
```

B.3.5 Extracting Random Effects

The user can compute the random effects using the provided `ranef` method for objects of class `ggmix_gic`. This command will compute the estimated random effects for each subject using the parameters of the selected model:

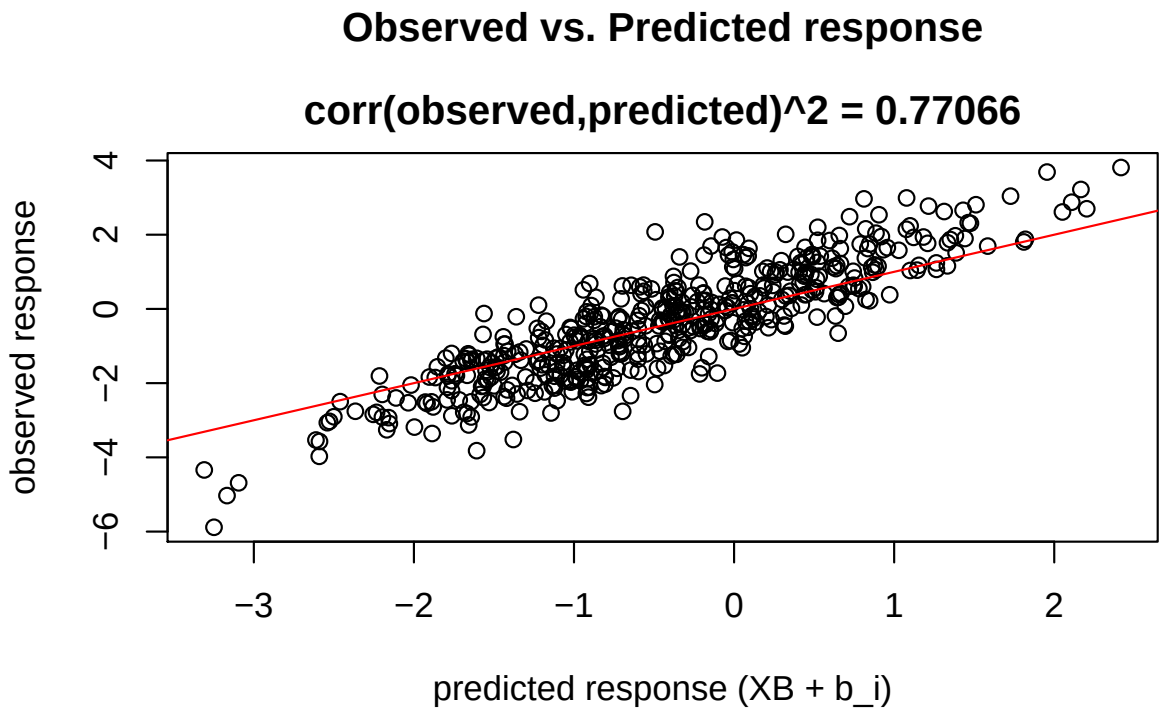
```
ranef(hdbic)[1:5]

## [1] -0.02548691 -0.10011680  0.13020240 -0.30650997  0.16045768
```

B.3.6 Diagnostic Plots

We can also plot some standard diagnotic plots such as the observed vs. predicted response, QQ-plots of the residuals and random effects and the Tukey-Anscombe plot. These can be plotted using the `plot` method on a `ggmix_gic` object as shown below.

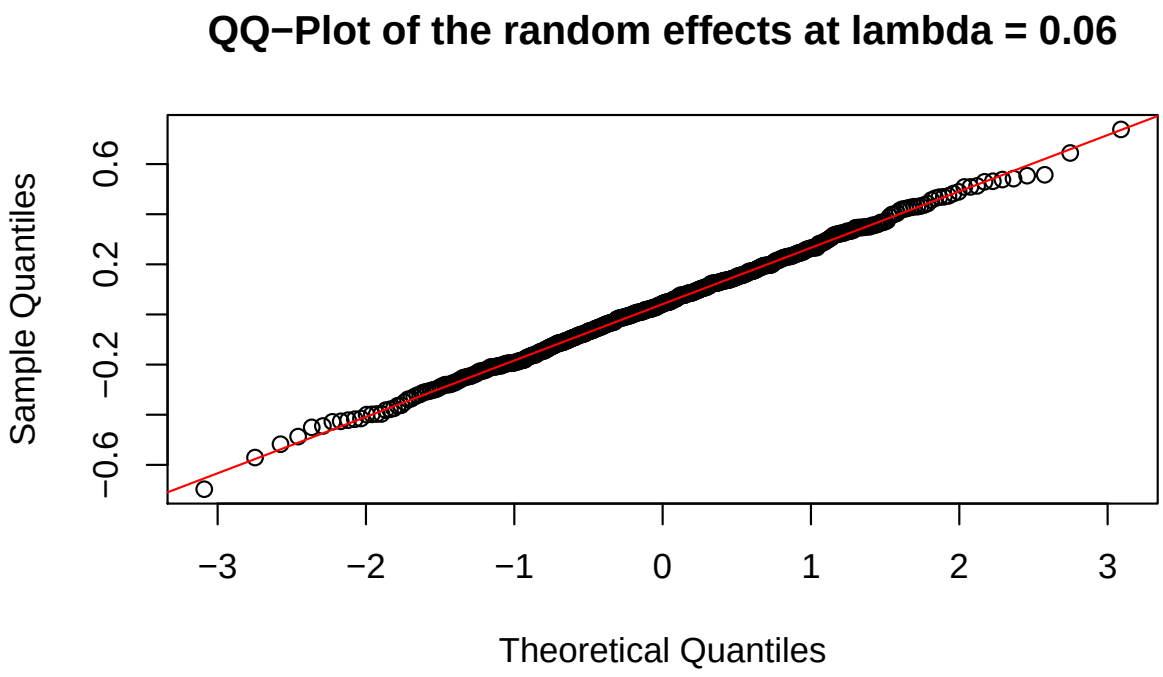
Observed vs. Predicted Response

```
plot(hdbic, type = "predicted", newx = admixed$x, newy = admixed$y)
```

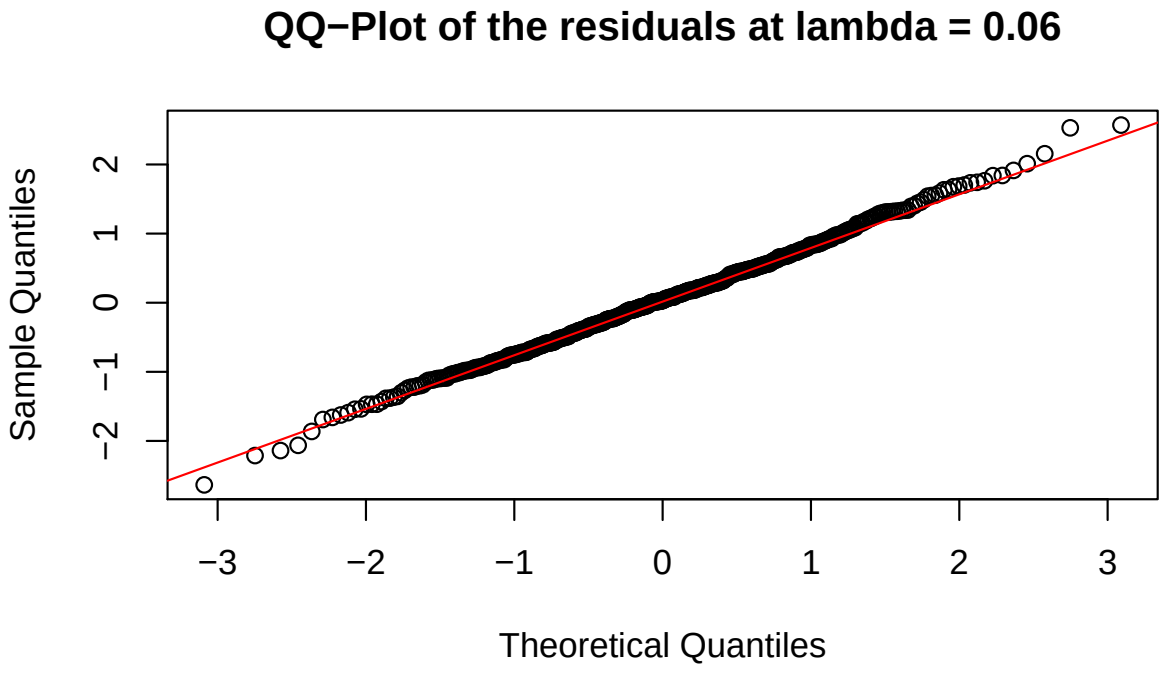


QQ-plots for Residuals and Random Effects

```
plot(hdbic, type = "QQranef", newx = admixed$x, newy = admixed$y)
```

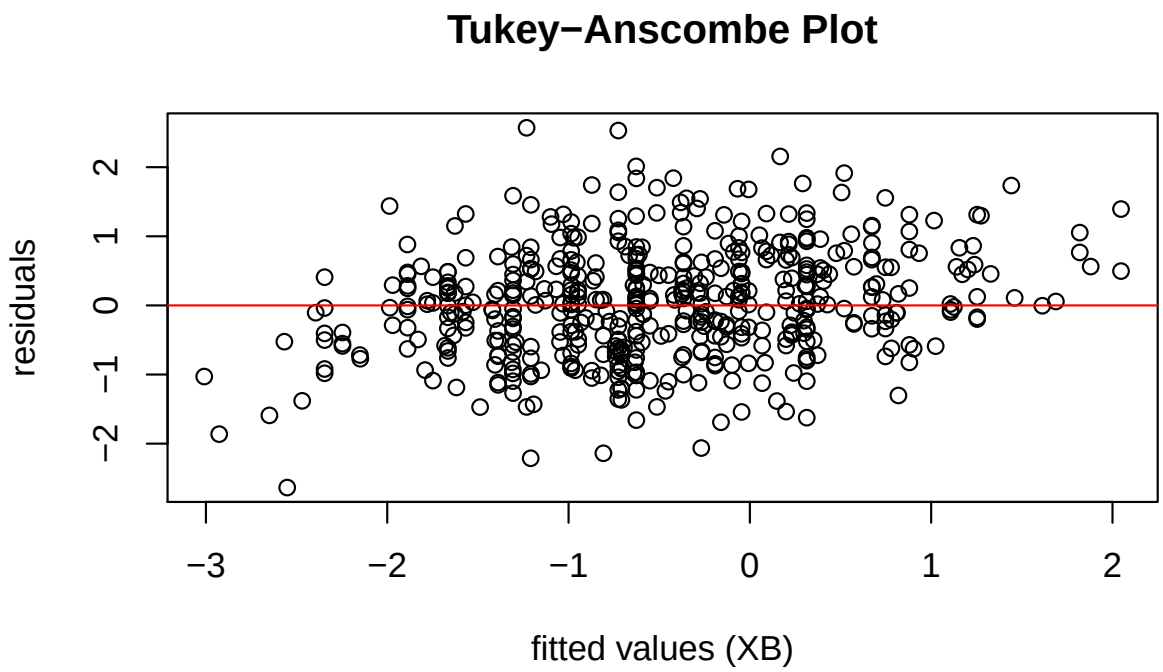



```
plot(hdbic, type = "QQresid", newx = admixed$x, newy = admixed$y)
```



Tukey-Anscombe Plot

```
plot(hdbic, type = "Tukey", newx = admixed$x, newy = admixed$y)
```



Appendix C

Supplemental Methods and Simulation Results for Chapter 5

C.1 Description of Topological Overlap Matrix

Starting with a similarity measure $s_{ij} = |cor(i,j)|$ between node i and node j, one could apply a hard threshold to determine if this pair is considered connected or not resulting in an un-weighted network (a matrix of 0's and 1's). Instead, Zhang and Horvath (B. Zhang & Horvath, 2005) propose a soft thresholding framework that assigns a connection weight to each gene pair using a power adjacency function $a_{ij} = |s_{ij}|^{\beta}$. The parameter β determines the sensitivity and specificity of the pairwise connection strengths e.g. a larger β will result in fewer connected nodes which can reduce noise in the network but can also eliminate signal if too large. A measure of similarity is then derived using the symmetric and non-negative topological overlap matrix (Ravasz et al., 2002) (TOM) $\Omega = [\omega_{ij}]$:

$$\omega_{ij} = \frac{l_{ij} + a_{ij}}{min\{k_i, k_j\} + 1 - a_{ij}} \tag{C.1}$$

where $l_{ij} = \sum_u a_{iu}a_{uj}$, $k_i = \sum_u a_{iu}$ is the node connectivity, and the index u runs across all nodes of the network. Basically, ω_{ij} is a measure of similarity in terms of the commonality of the nodes they connect to. If i and j are unconnected and do not share any neighbors then $\omega_{ij} = 0$. An $\omega_{ij} = 1$ means that i and j are connected, and the neighbors of the node with fewer connections are also neighbors of the other node.

C.2 Binary Outcome Simulation Results

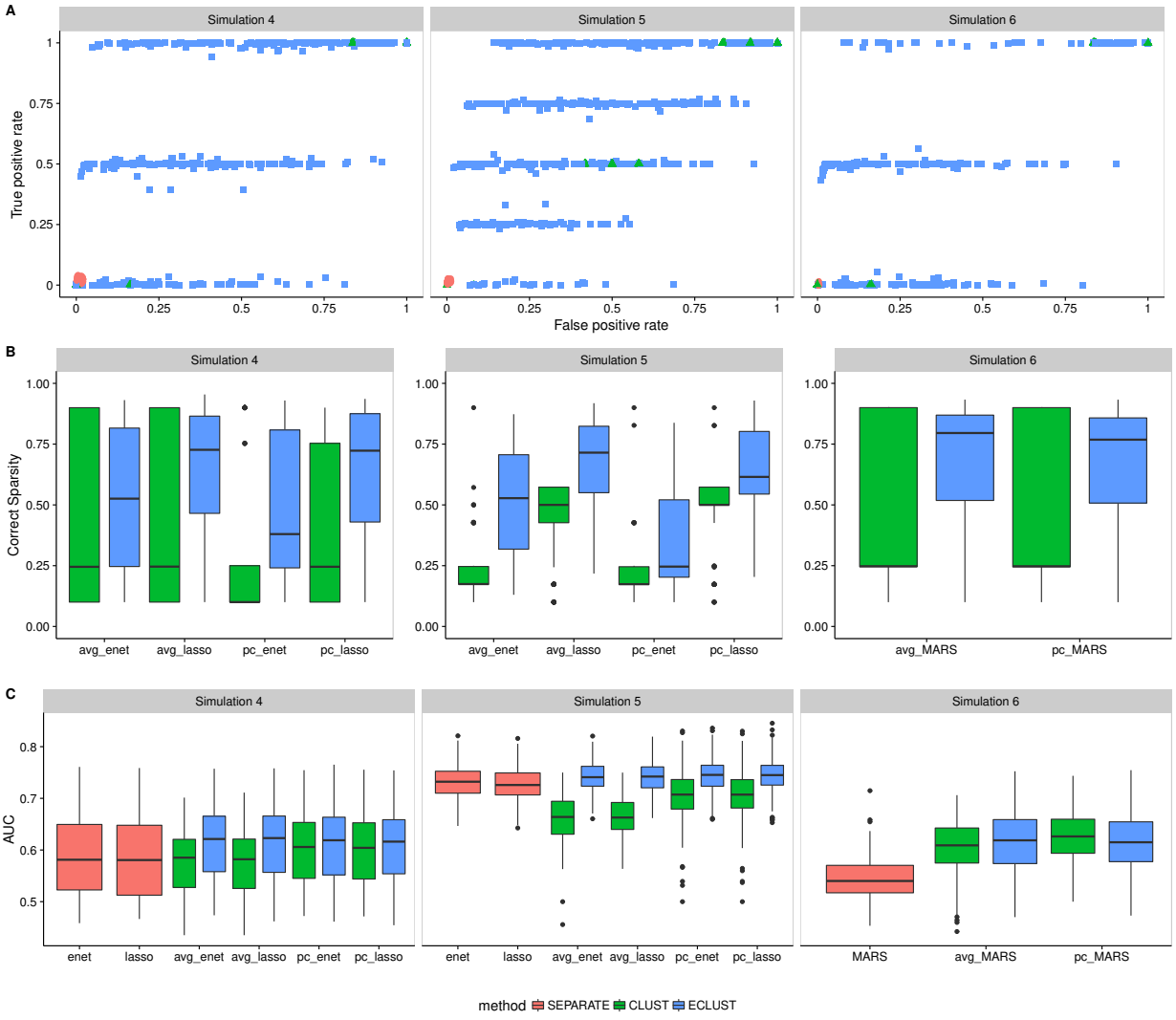


Figure C.1: Model fit results from simulations 4, 5 and 6 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

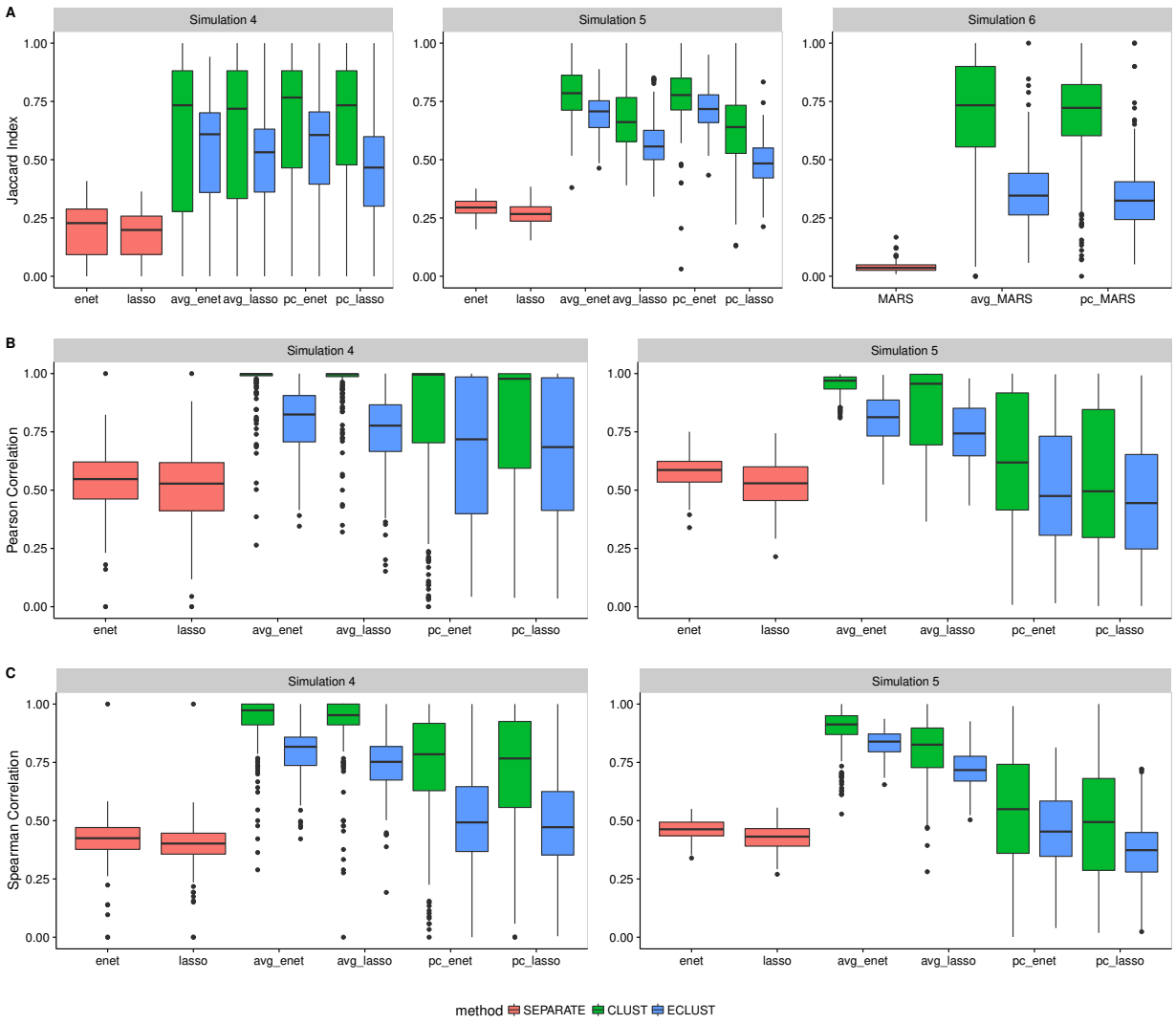


Figure C.2: Stability results from simulations 4, 5 and 6 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

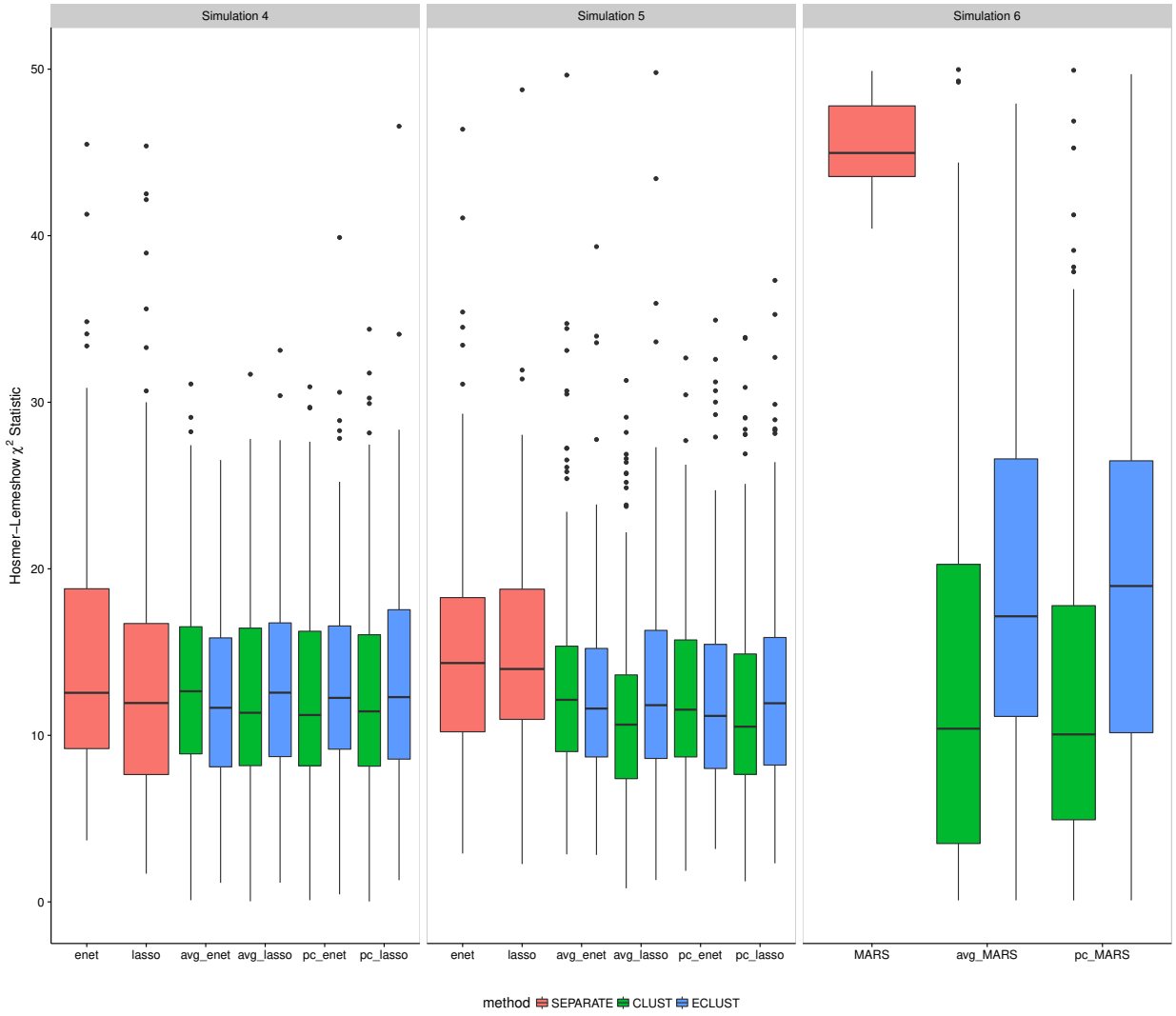


Figure C.3: Hosmer-Lemeshow statistics from simulations 4, 5 and 6 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

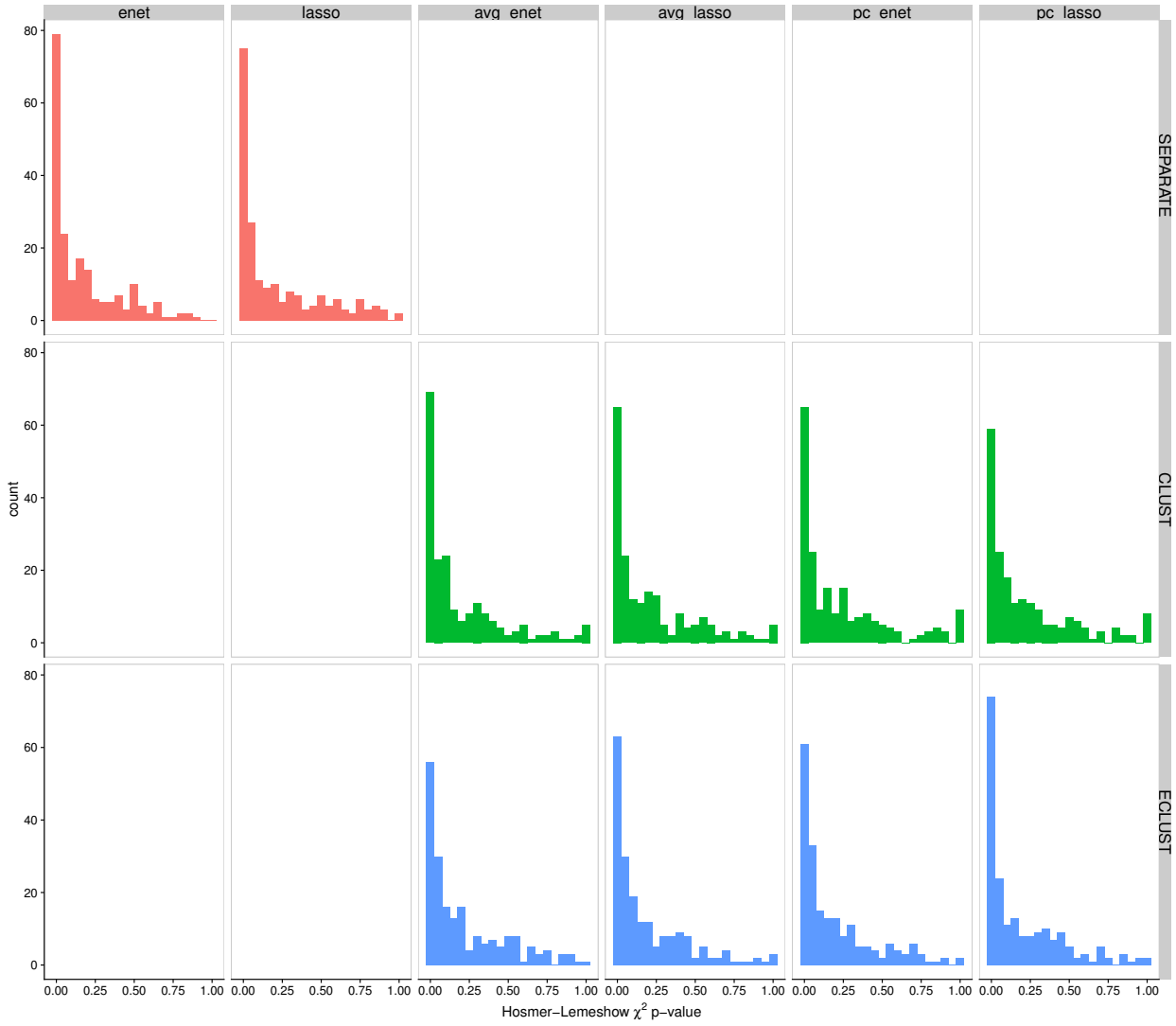


Figure C.4: Hosmer-Lemeshow p-values from simulation 4 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

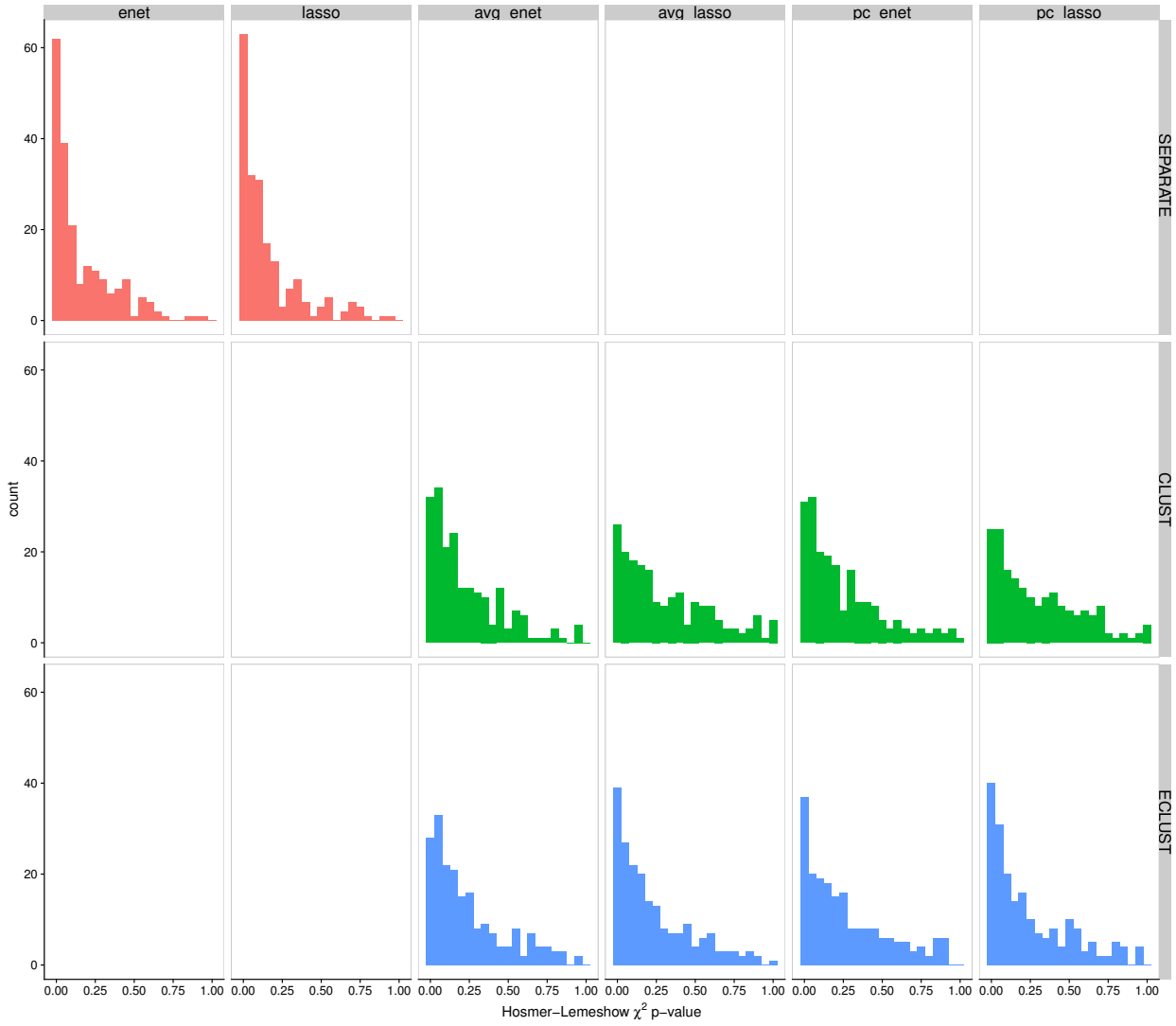


Figure C.5: Hosmer-Lemeshow p-values from simulation 5 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

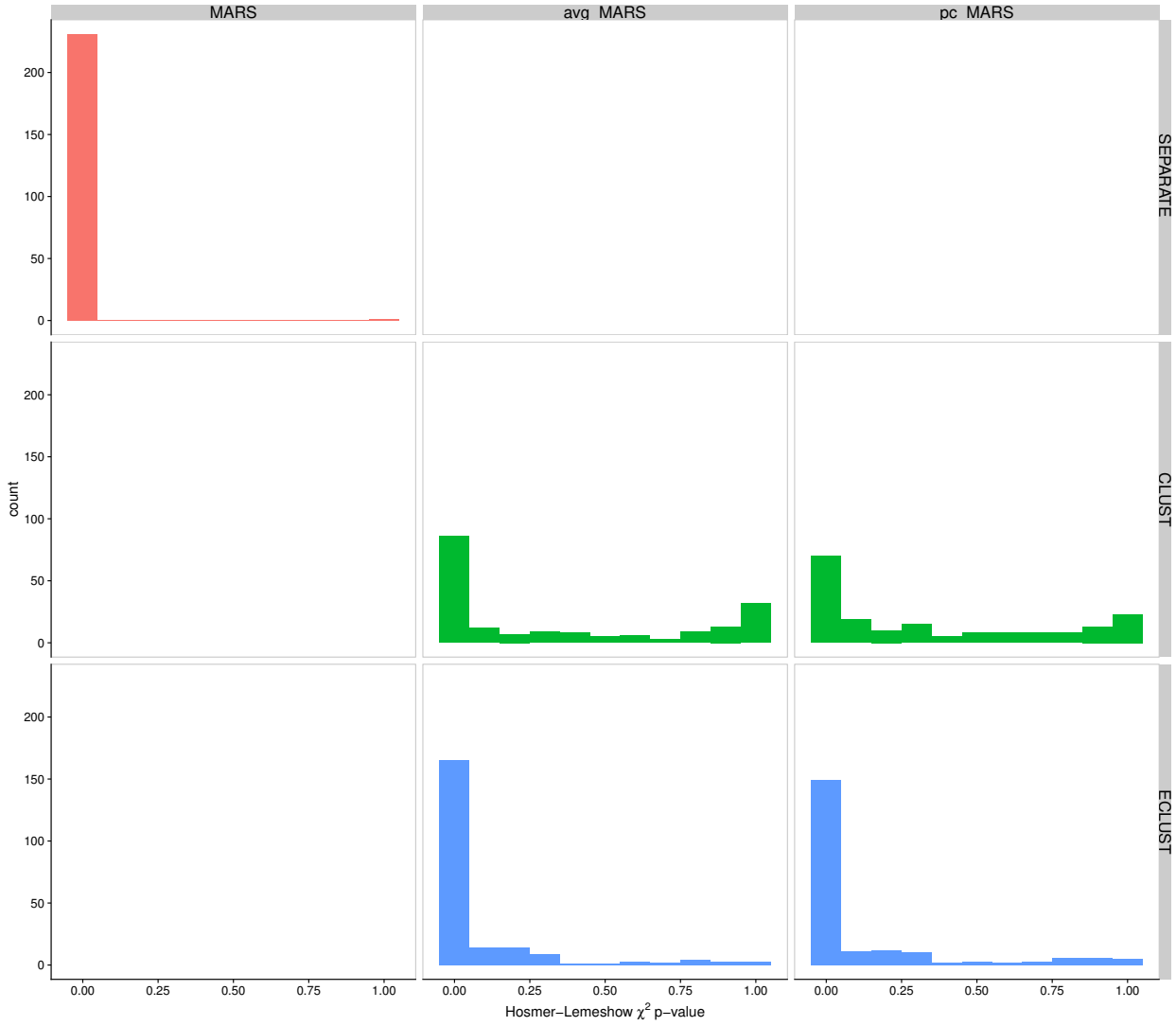


Figure C.6: Hosmer-Lemeshow p-values from simulation 6 for $SNR = 1$, $\rho = 0.9$, and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. SEPARATE results are in pink, CLUST in green and ECLUST in blue.

C.3 Analysis of Clusters

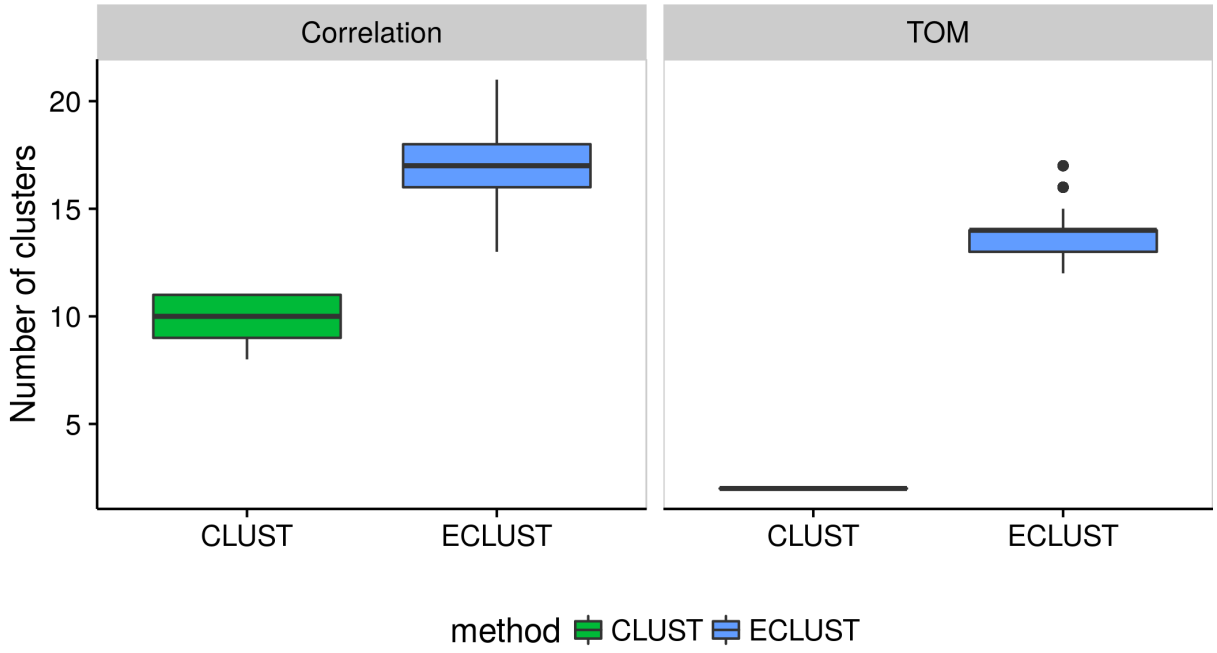


Figure C.7: Number of estimated clusters from applying the `dynamicTreeCut` algorithm to hierarchical clustering of the dissimilarity matrix with average linkage. Left panel: CLUST uses $1 - Cor(X_{all})$ and ECLUST uses the euclidean distance of $Cor(X_{\text{diff}})$ as measures of dissimilarity. Right panel: CLUST uses $1 - TOM(X_{all})$ and ECLUST uses the euclidean distance of $TOM(X_{\text{diff}})$ as measures of dissimilarity. Empirical distributions based on 200 simulation runs.

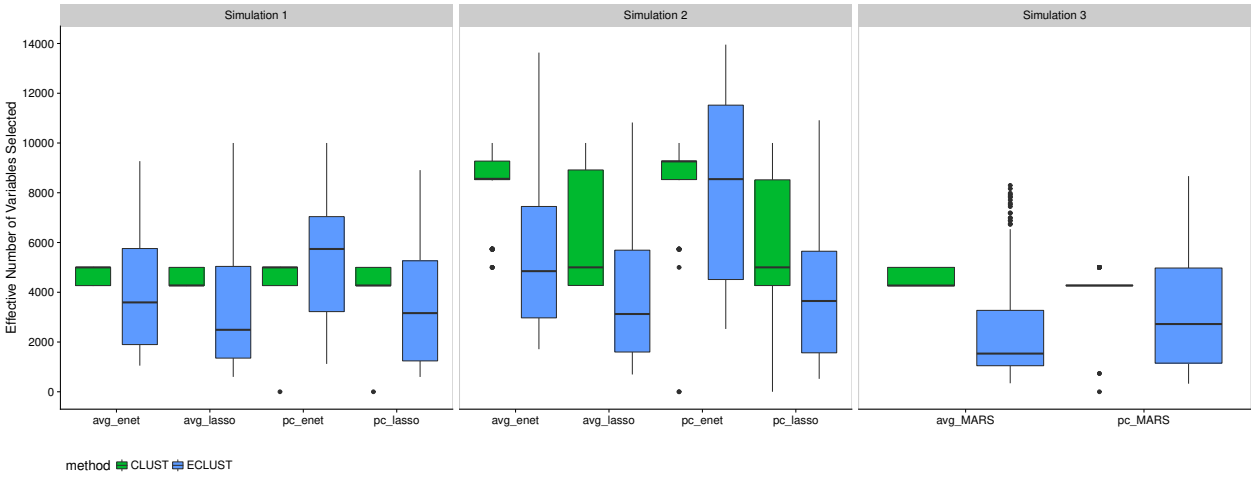


Figure C.8: Effective number of selected variables for simulations 1-3 for $SNR = 1, \rho = 0.9$. A variable was considered "selected" if its corresponding cluster representative was selected.

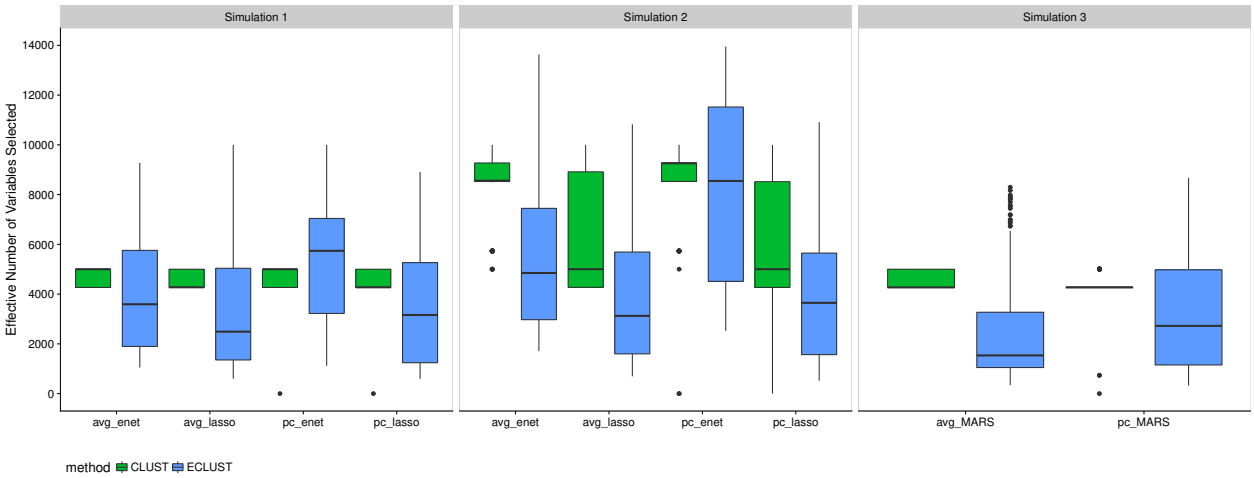


Figure C.9: Effective number of selected variables for simulations 4-6 for $SNR = 1, \rho = 0.9$ and $\alpha_j \sim \text{Unif}\left[\log(1.9), \log(2.1)\right]$. A variable was considered "selected" if its corresponding cluster representative was selected.

C.4 Simulation Results Using TOM as a Measure of Similarity

Simulation 1

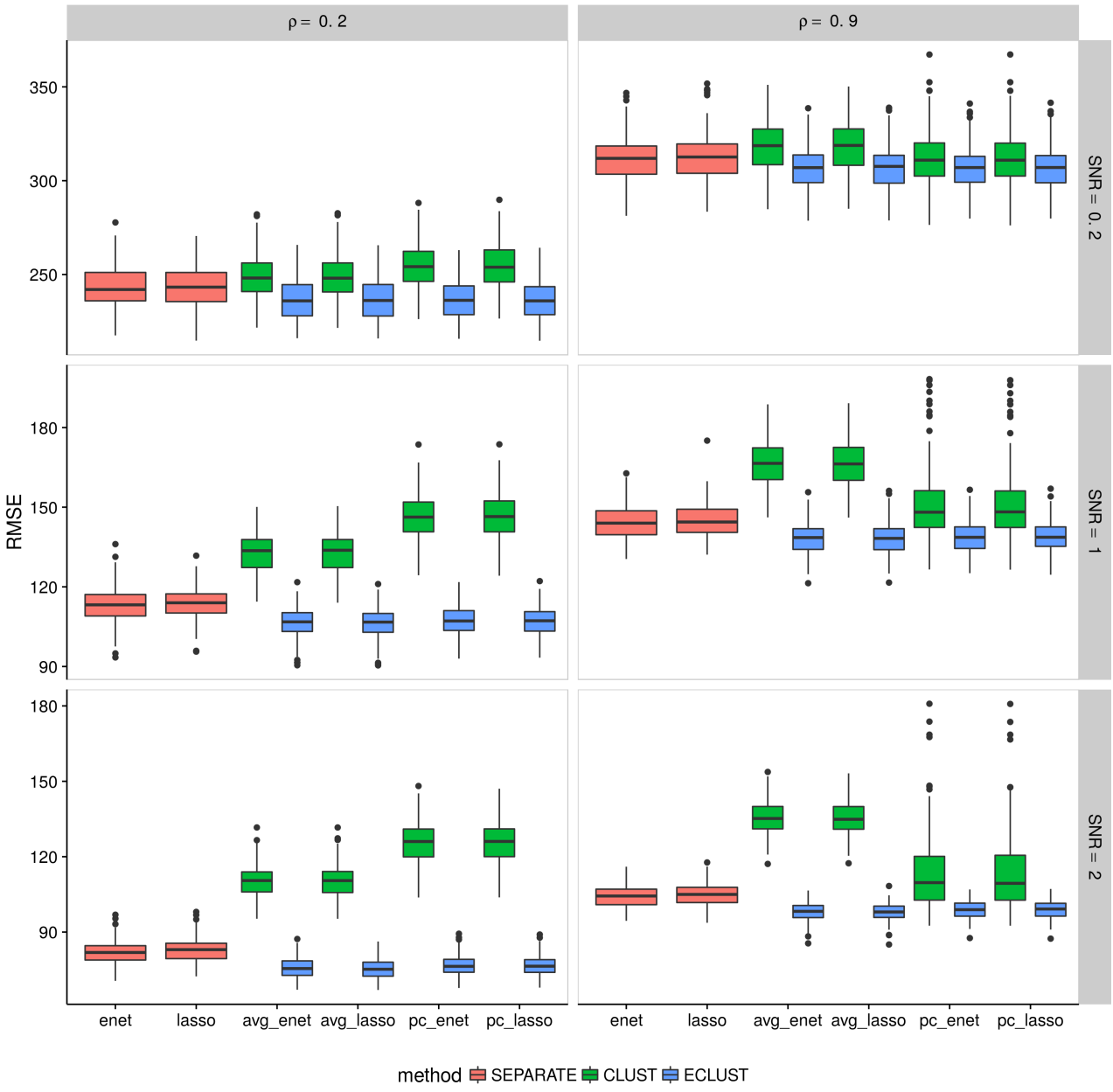


Figure C.10: Simulation 1 – Root mean squared error on an independent test set using the TOM as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

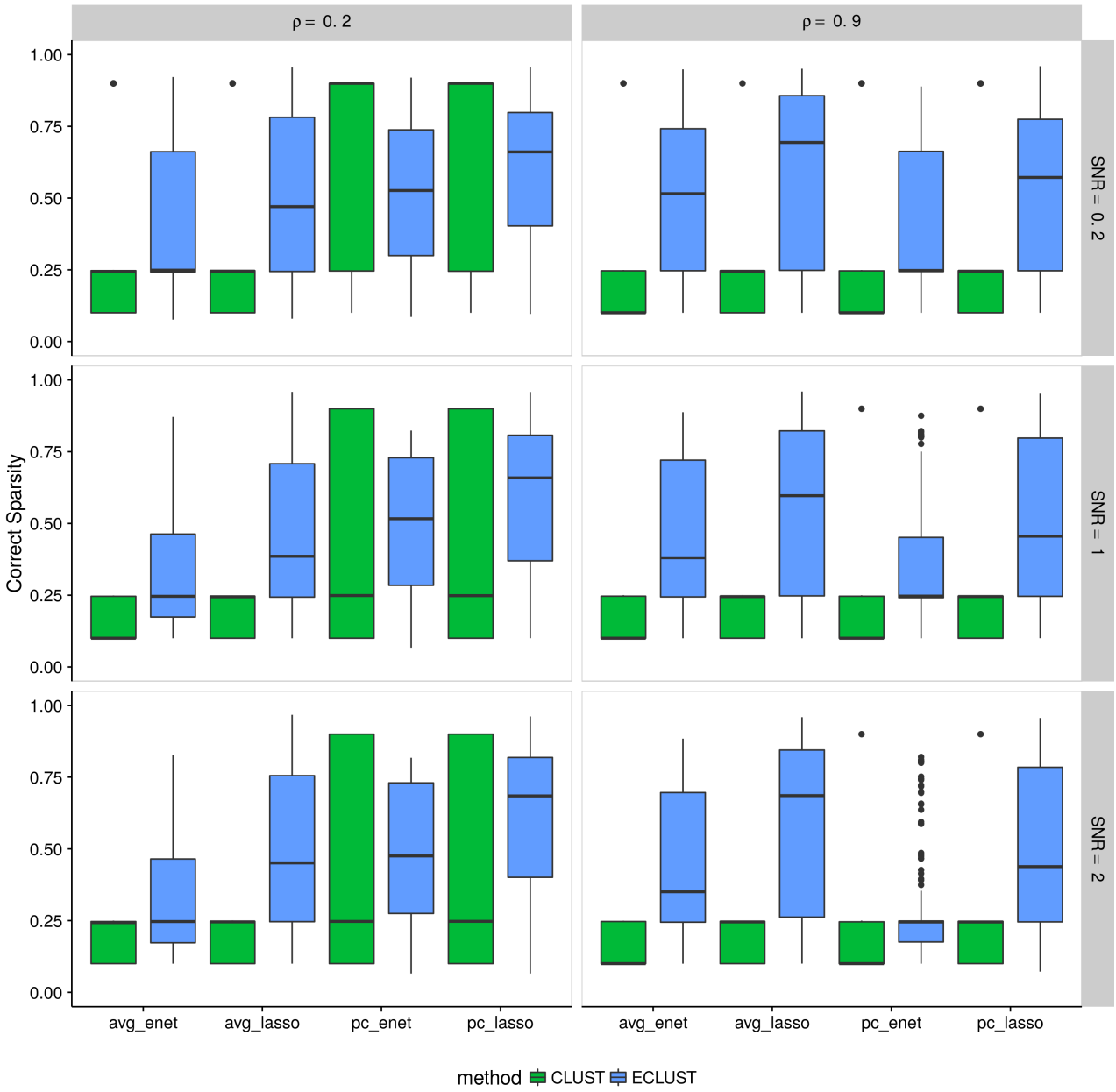


Figure C.11: Simulation 1 – Correct Sparsity based on the training set using the TOM as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

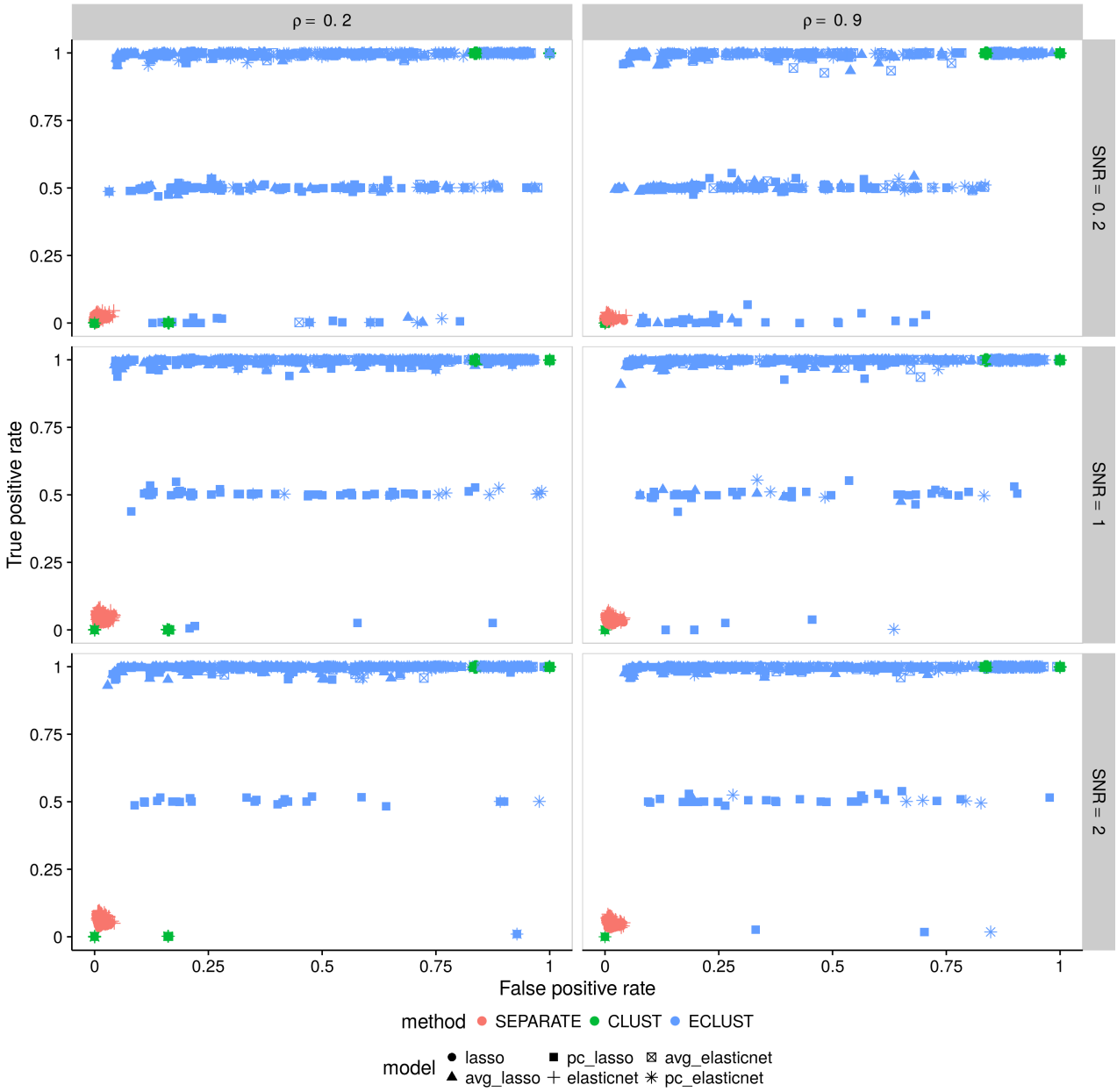


Figure C.12: Simulation 1 – True positive rate vs. false positive rate based on the training set using the TOM as a measure of similarity. Each point represents 1 simulation run (there are a total of 200 simulation runs). Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

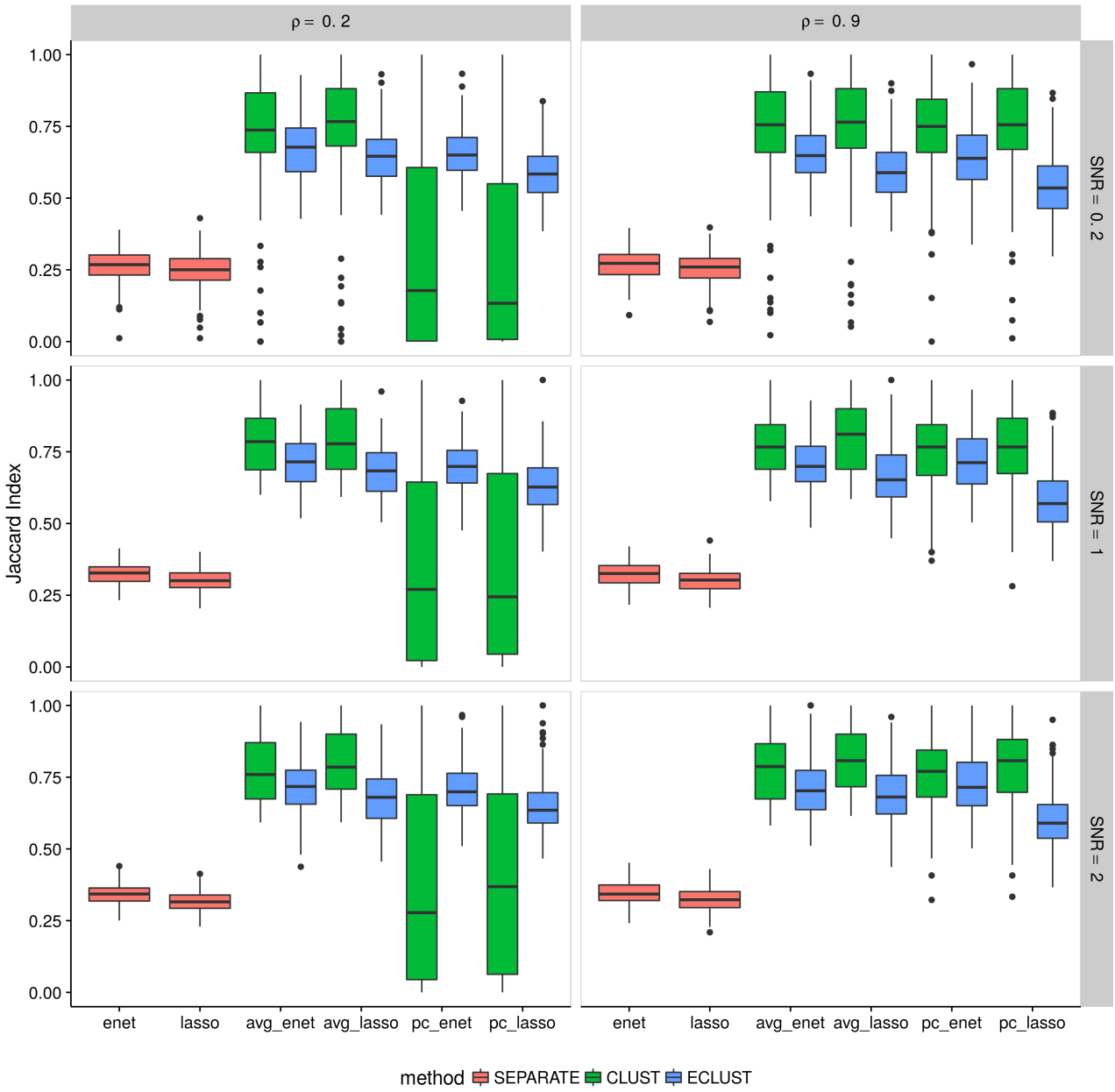


Figure C.13: Simulation 1 – Average Jaccard Index from 10 CV folds of the training set using the TOM as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of selected predictors. We then calculate the Jaccard Index between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

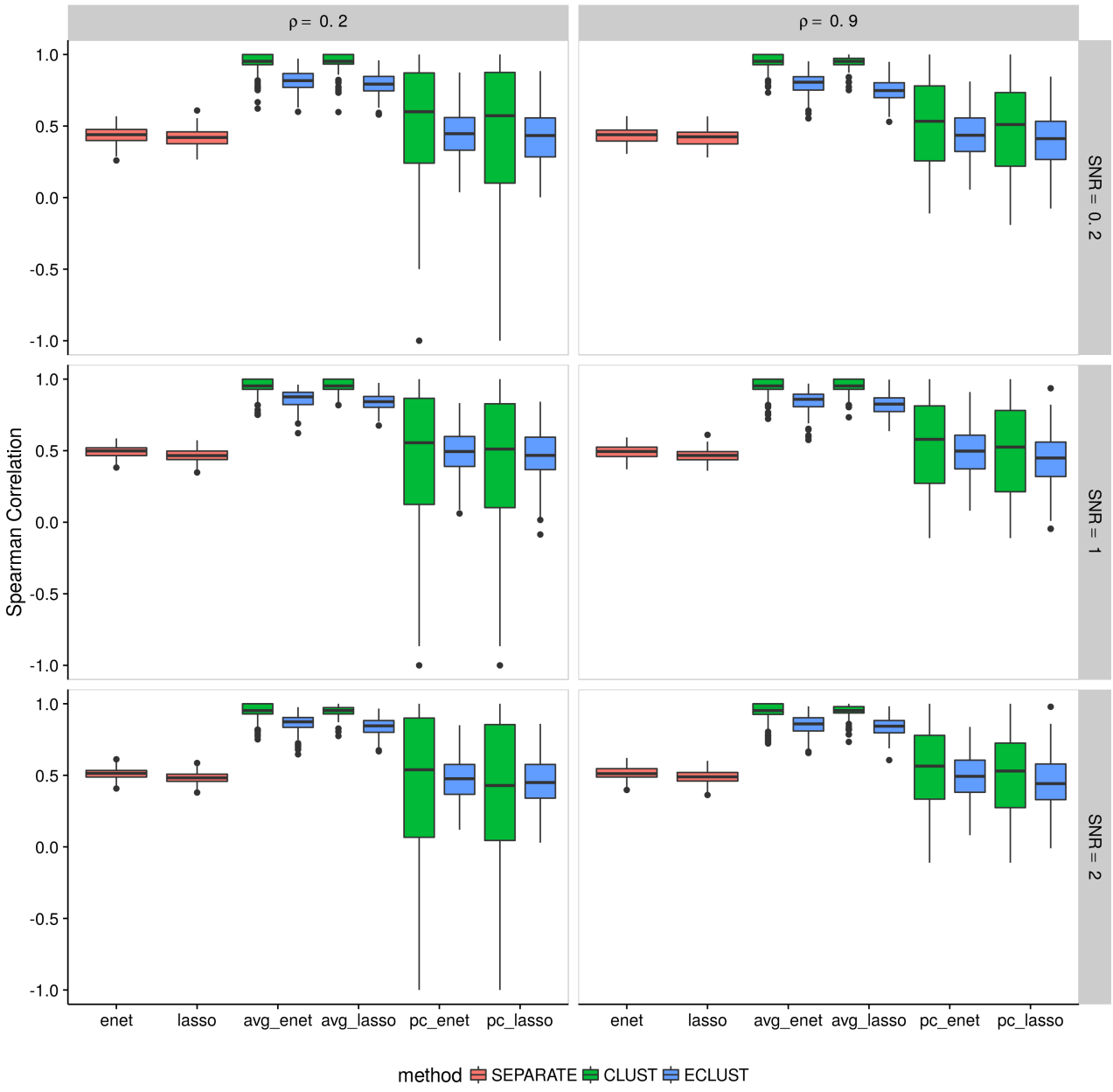


Figure C.14: Simulation 1 – Average Spearman correlation from 10 CV folds of the training set using the TOM as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Spearman correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

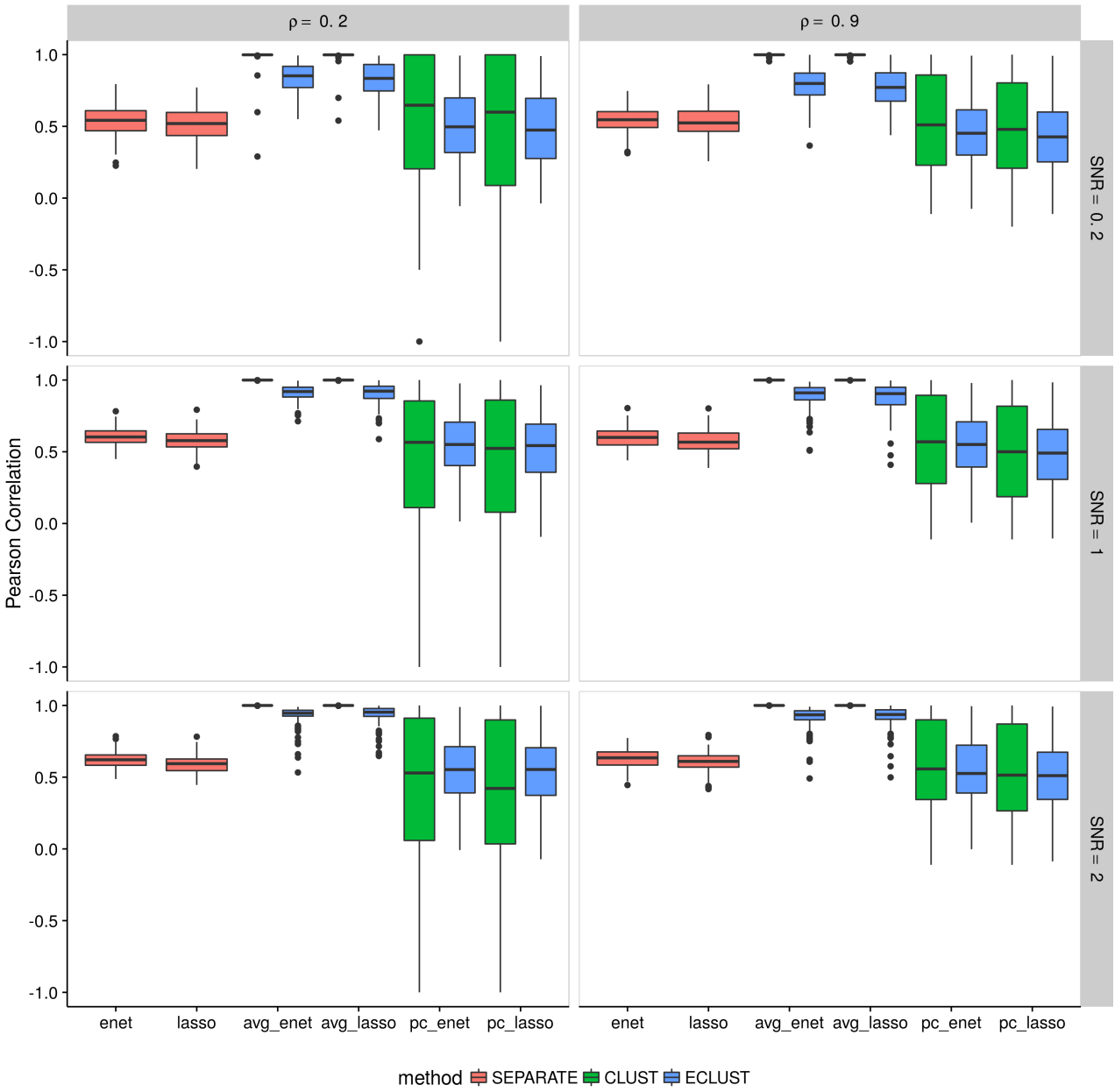


Figure C.15: Simulation 1 – Average Pearson correlation from 10 CV folds of the training set using the TOM as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Pearson correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

Simulation 2

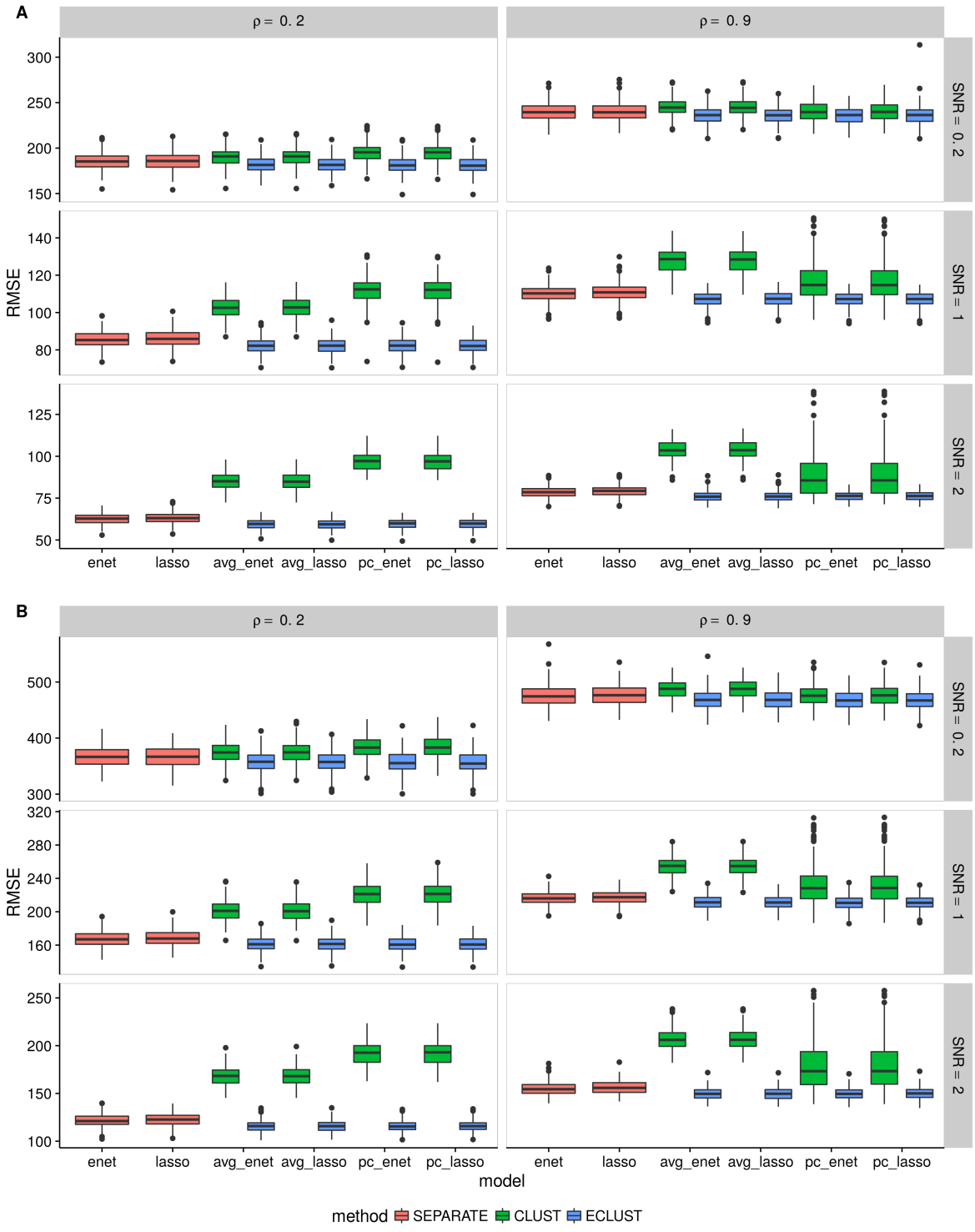


Figure C.16: Simulation 2 – Root mean squared error on an independent test set using the TOM as a measure of similarity from 200 simulation runs. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

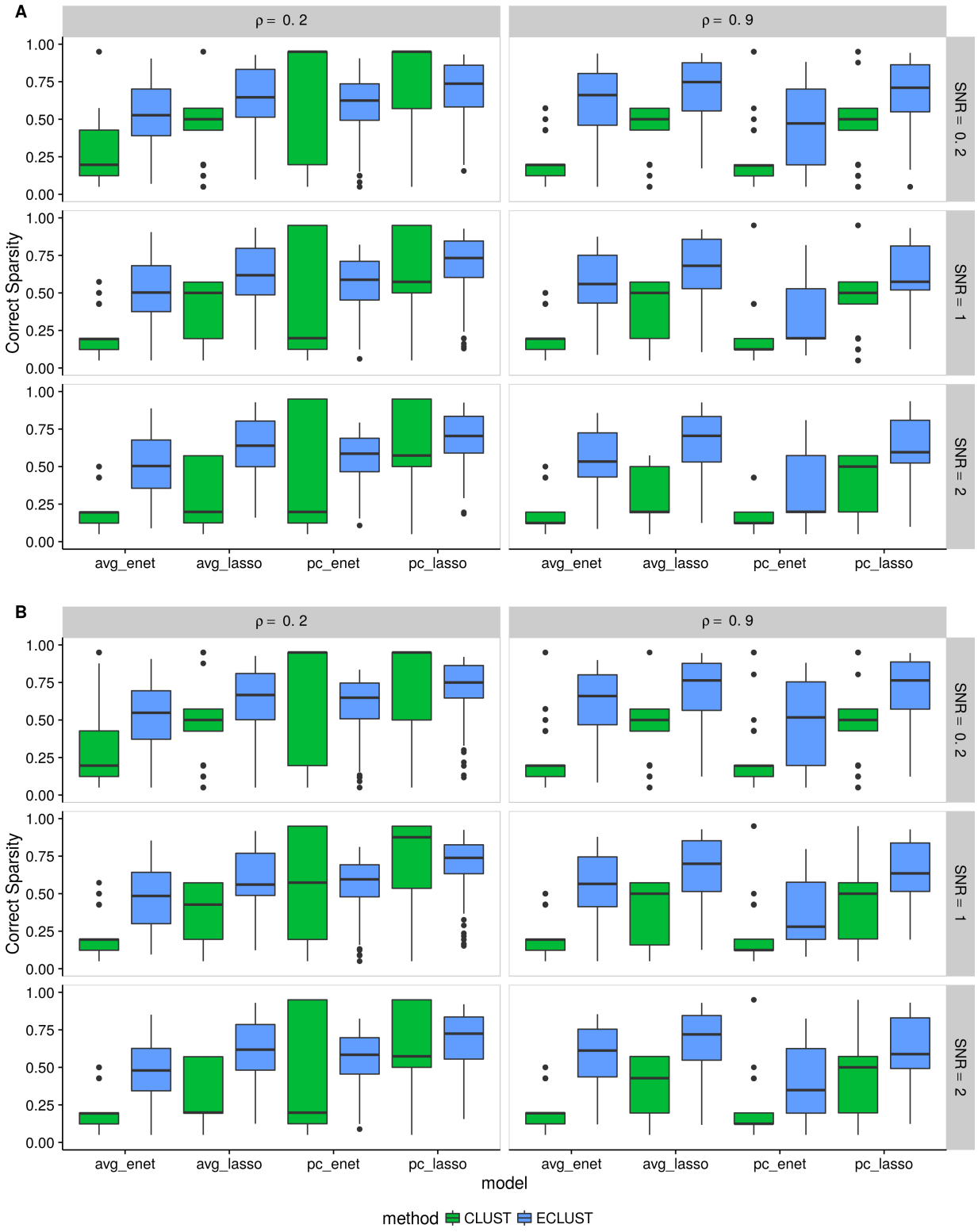


Figure C.17: Simulation 2 – Correct Sparsity based on the training set using the TOM as a measure of similarity from 200 simulation runs. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

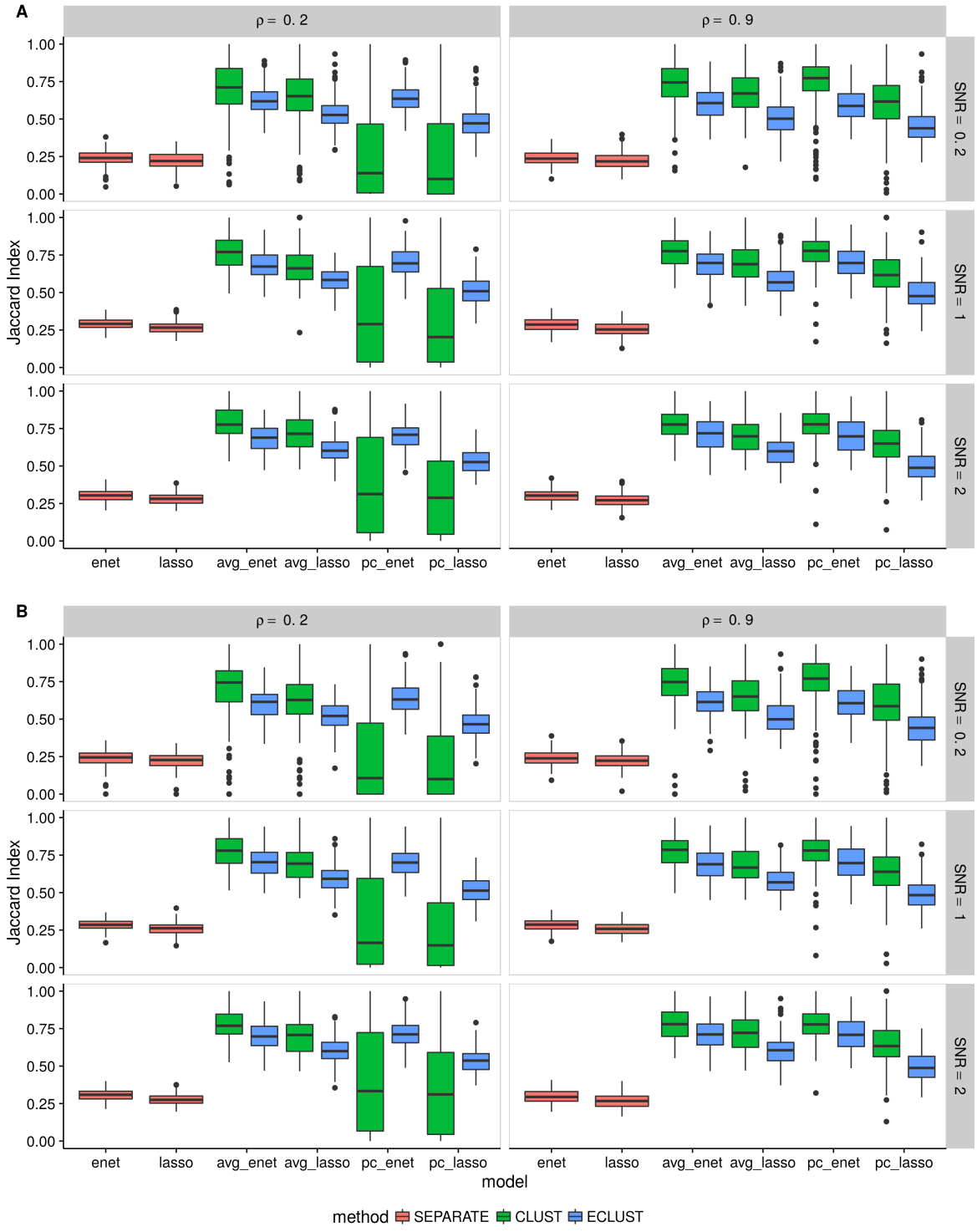


Figure C.19: Simulation 2 – Average Jaccard Index from 10 CV folds of the training set using the TOM as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. We fit the model to each of the 10 CV folds resulting in 10 sets of selected predictors. We then calculate the Jaccard Index between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

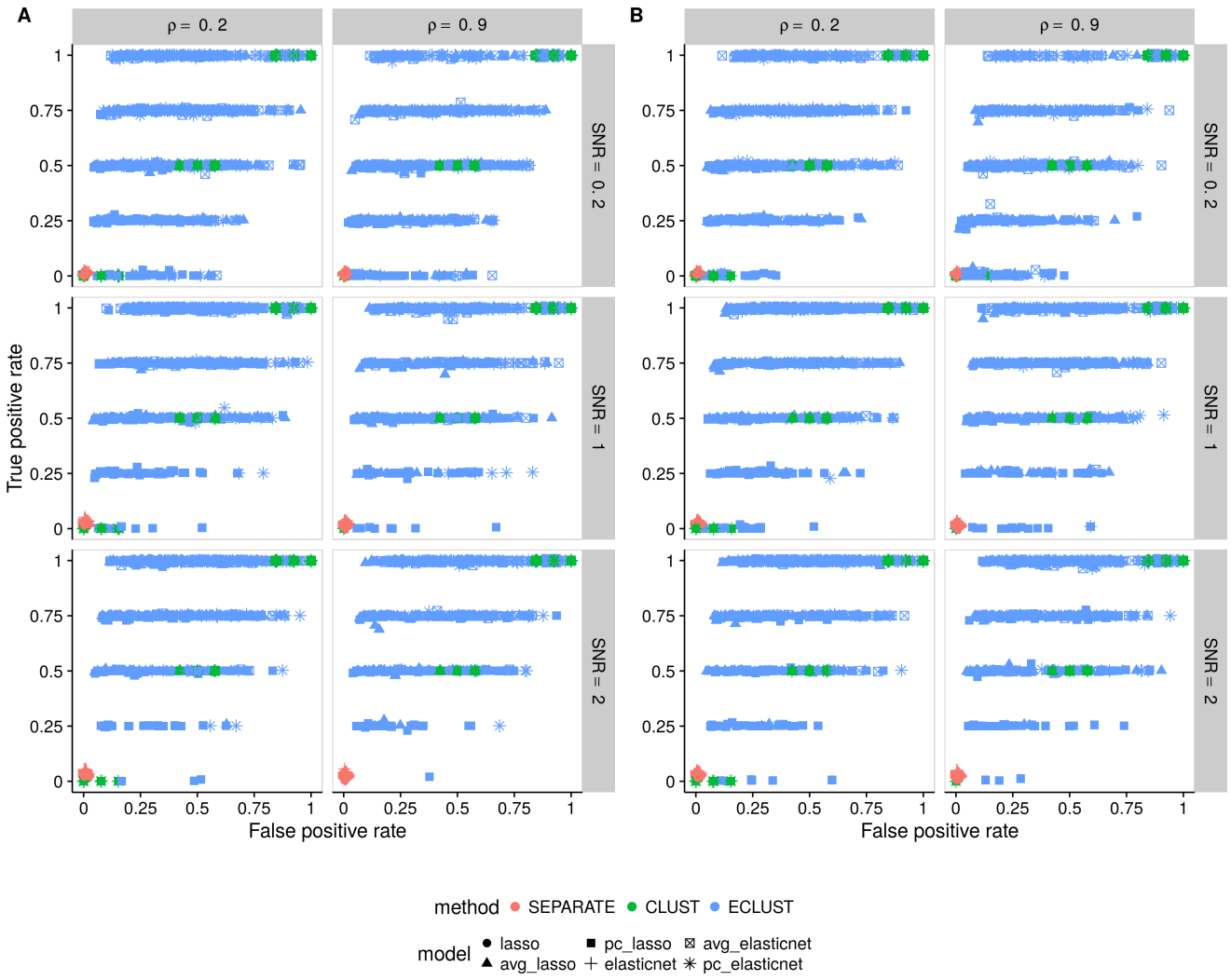


Figure C.18: Simulation 2 – True positive rate vs. false positive rate based on the training set using the TOM as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. Each point represents 1 simulation run (there are a total of 200 simulation runs). Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

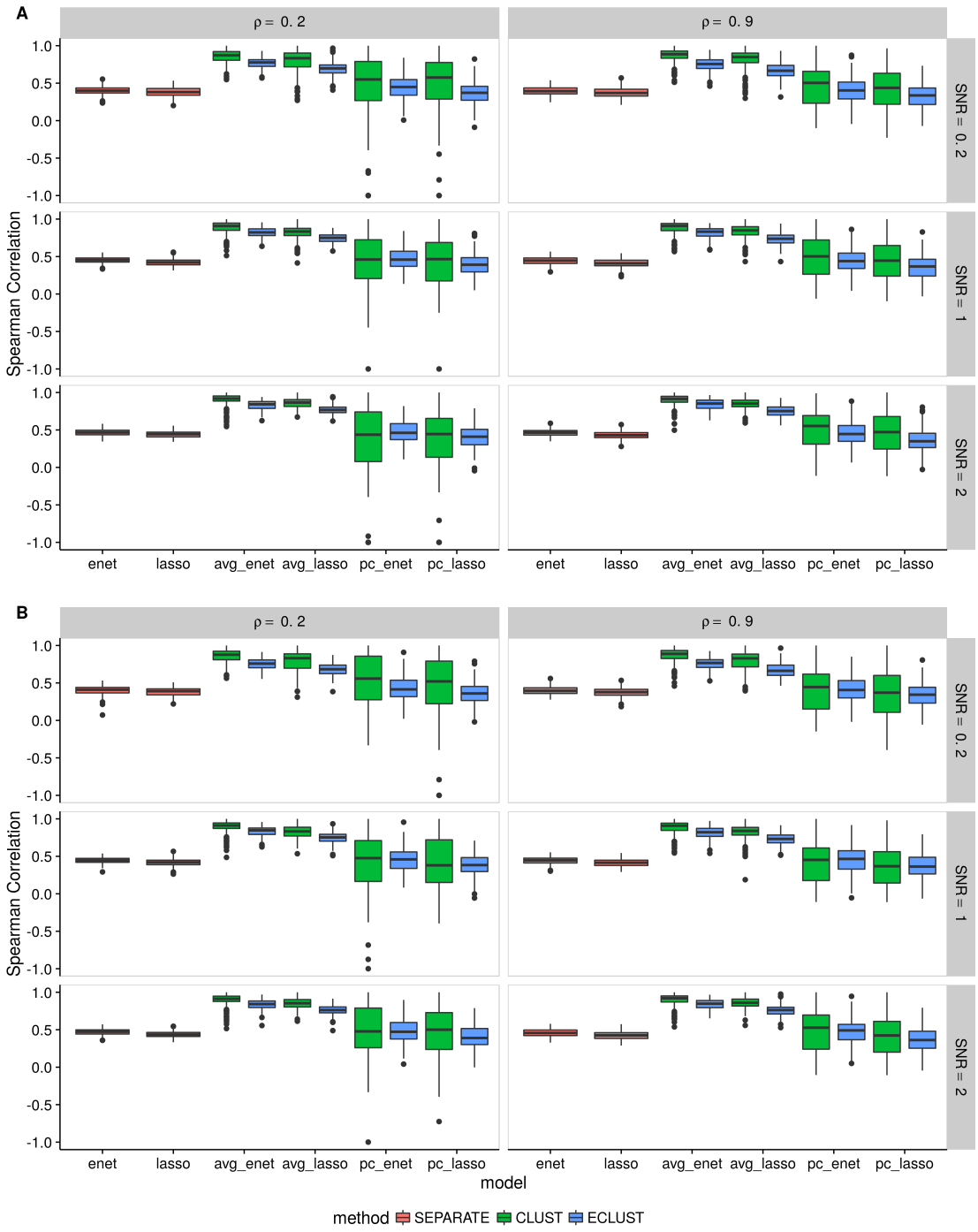


Figure C.20: Simulation 2 – Average Spearman correlation from 10 CV folds of the training set using the TOM as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Spearman correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

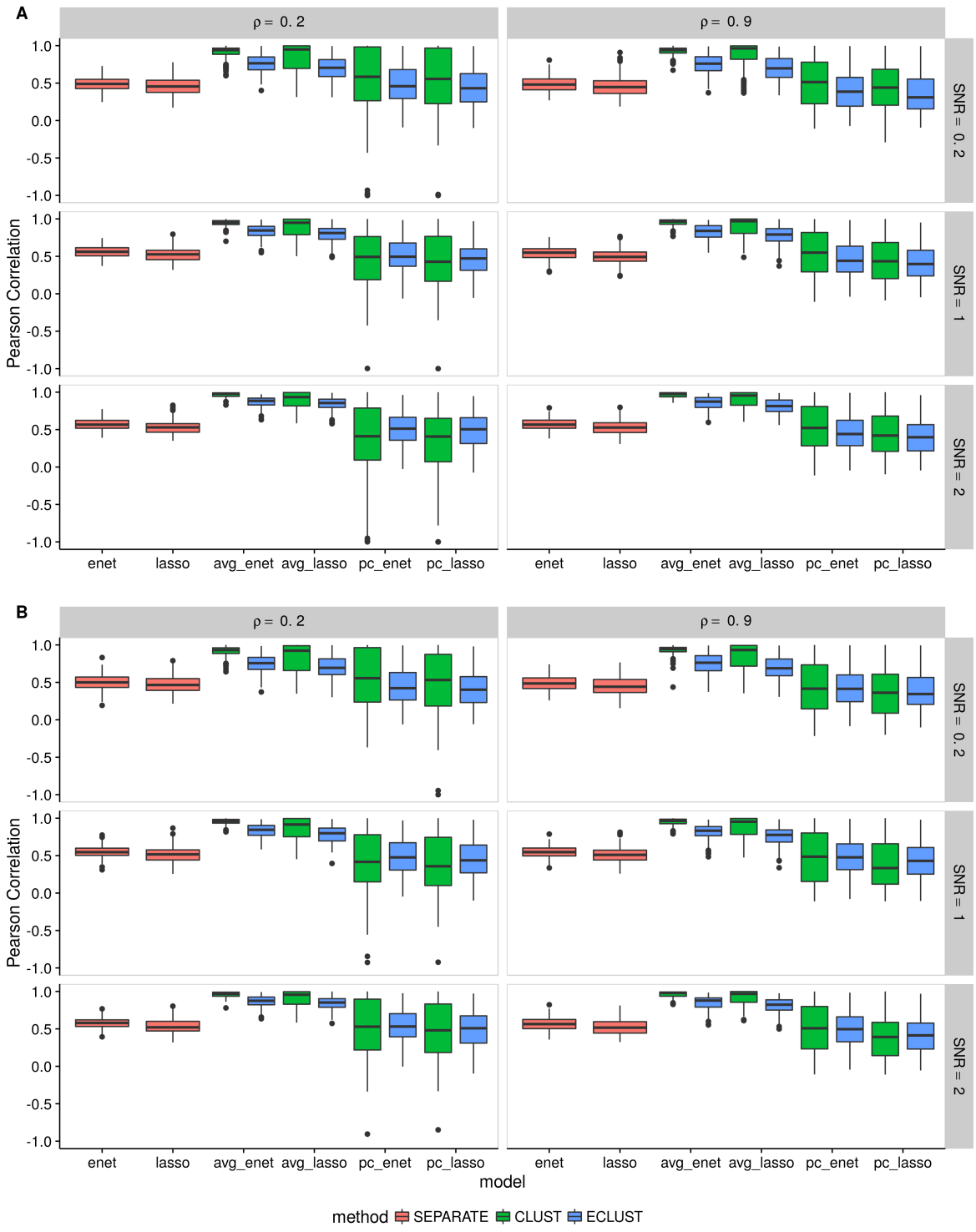


Figure C.21: Simulation 2 – Average Pearson correlation from 10 CV folds of the training set using the TOM as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Pearson correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

Simulation 3

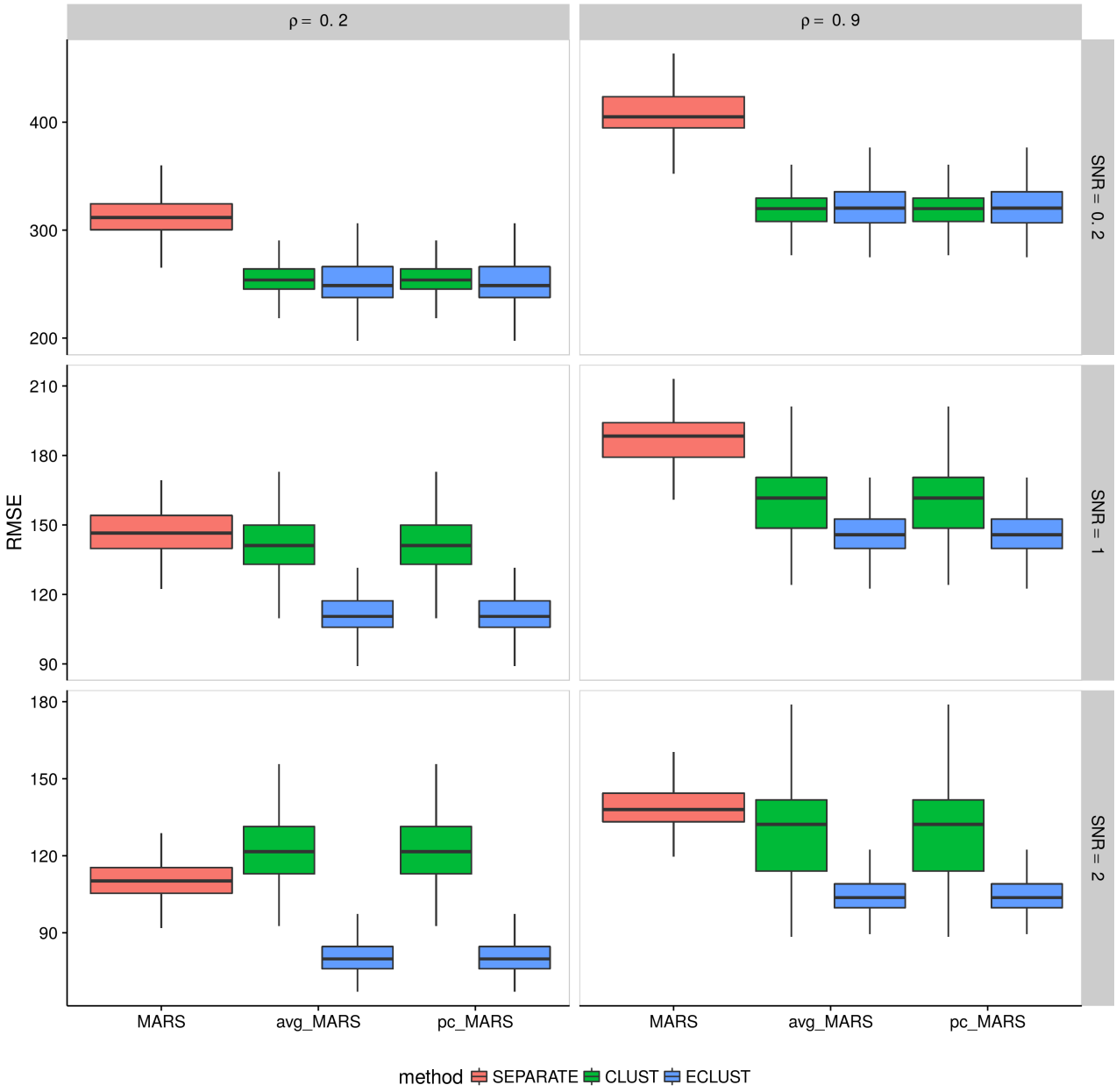


Figure C.22: Simulation 3 – Root mean squared error on an independent test set using the TOM as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

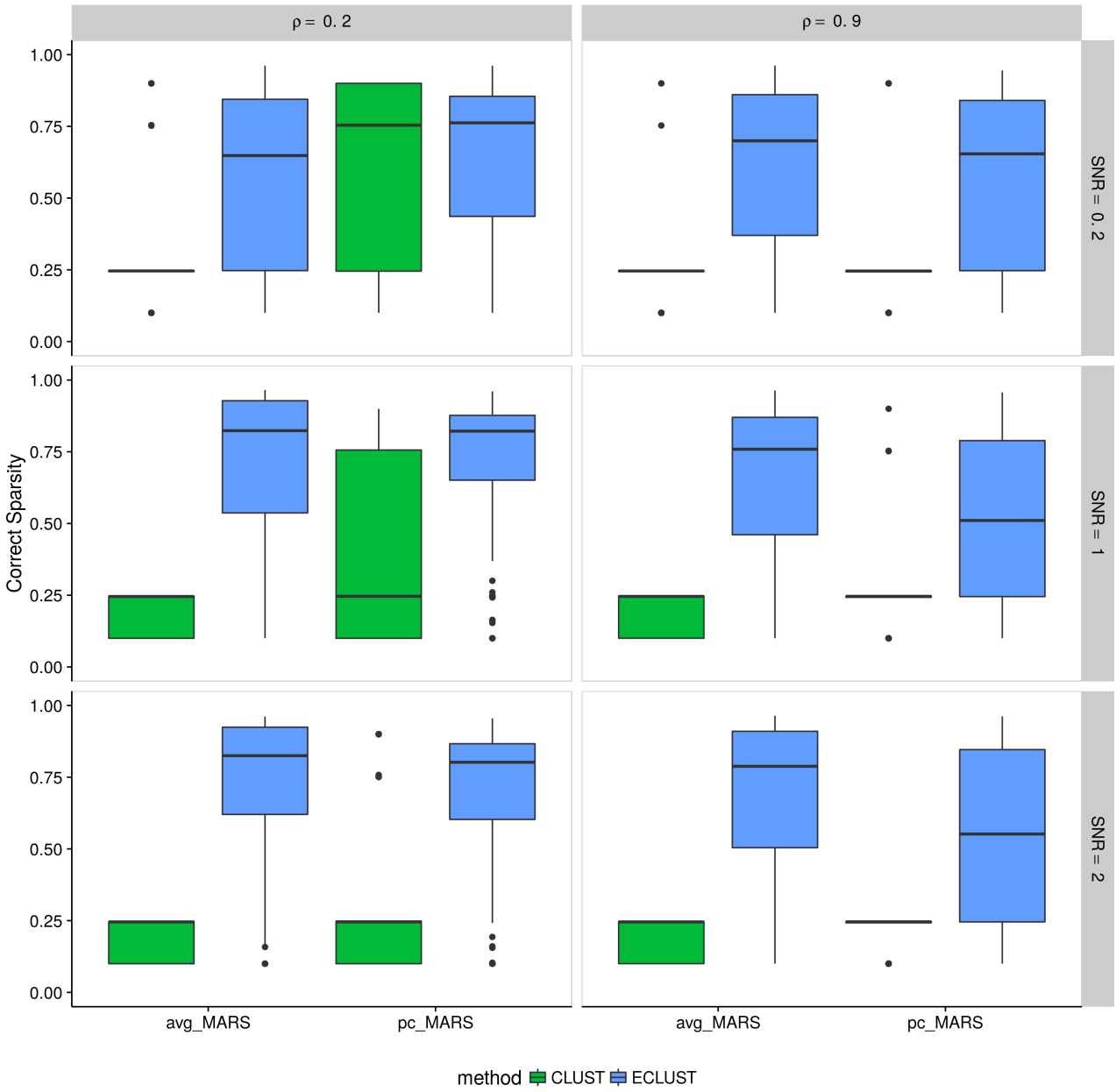


Figure C.23: Simulation 3 – Correct Sparsity based on the training set using the TOM as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

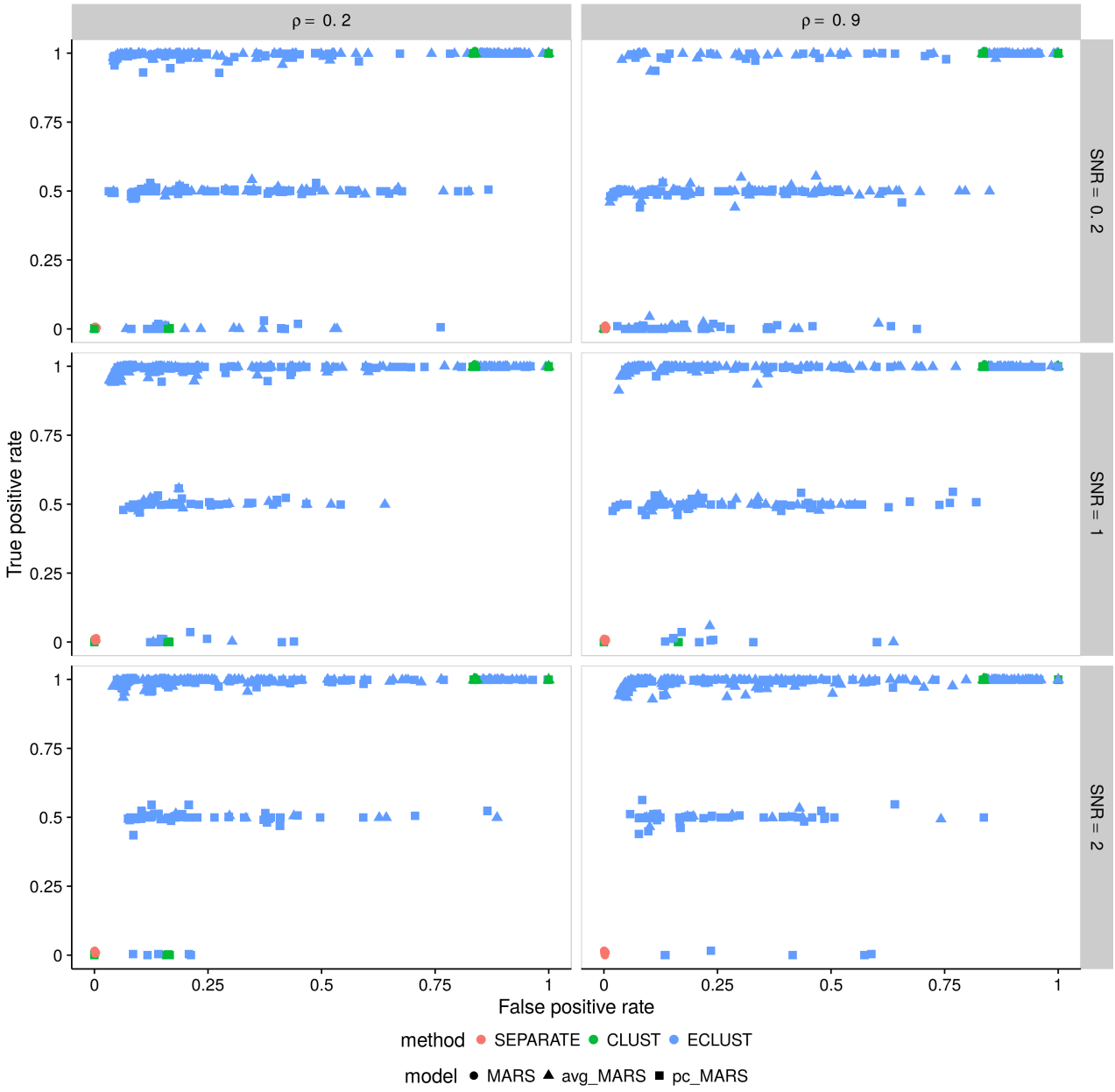


Figure C.24: Simulation 3 – True positive rate vs. false positive rate based on the training set using the TOM as a measure of similarity. Each point represents 1 simulation run (there are a total of 200 simulation runs). Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

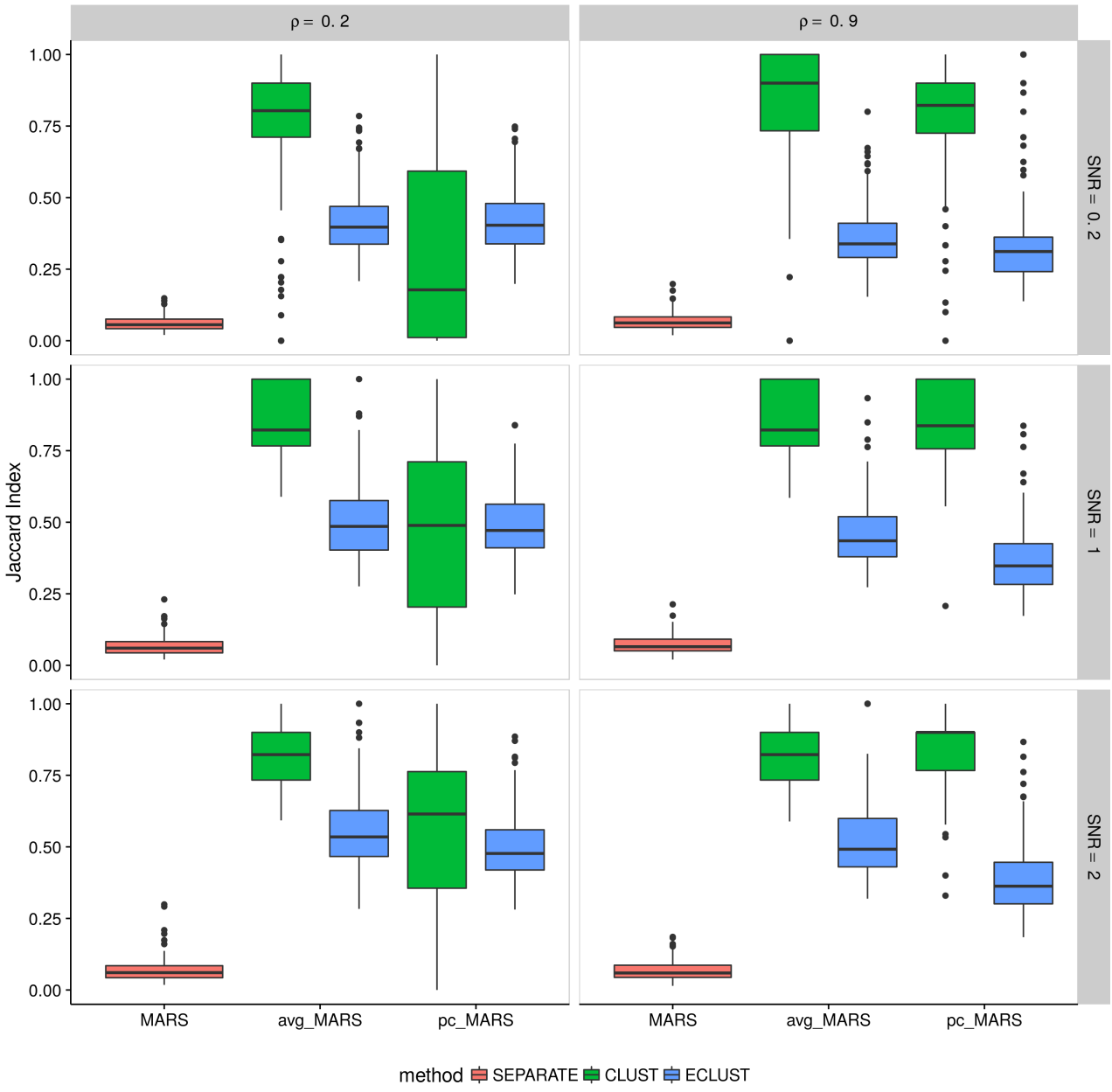


Figure C.25: Simulation 3 – Average Jaccard Index from 10 CV folds of the training set using the TOM as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of selected predictors. We then calculate the Jaccard Index between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

C.5 Simulation Results Using Pearson Correlations as a Measure of Similarity

Simulation 1

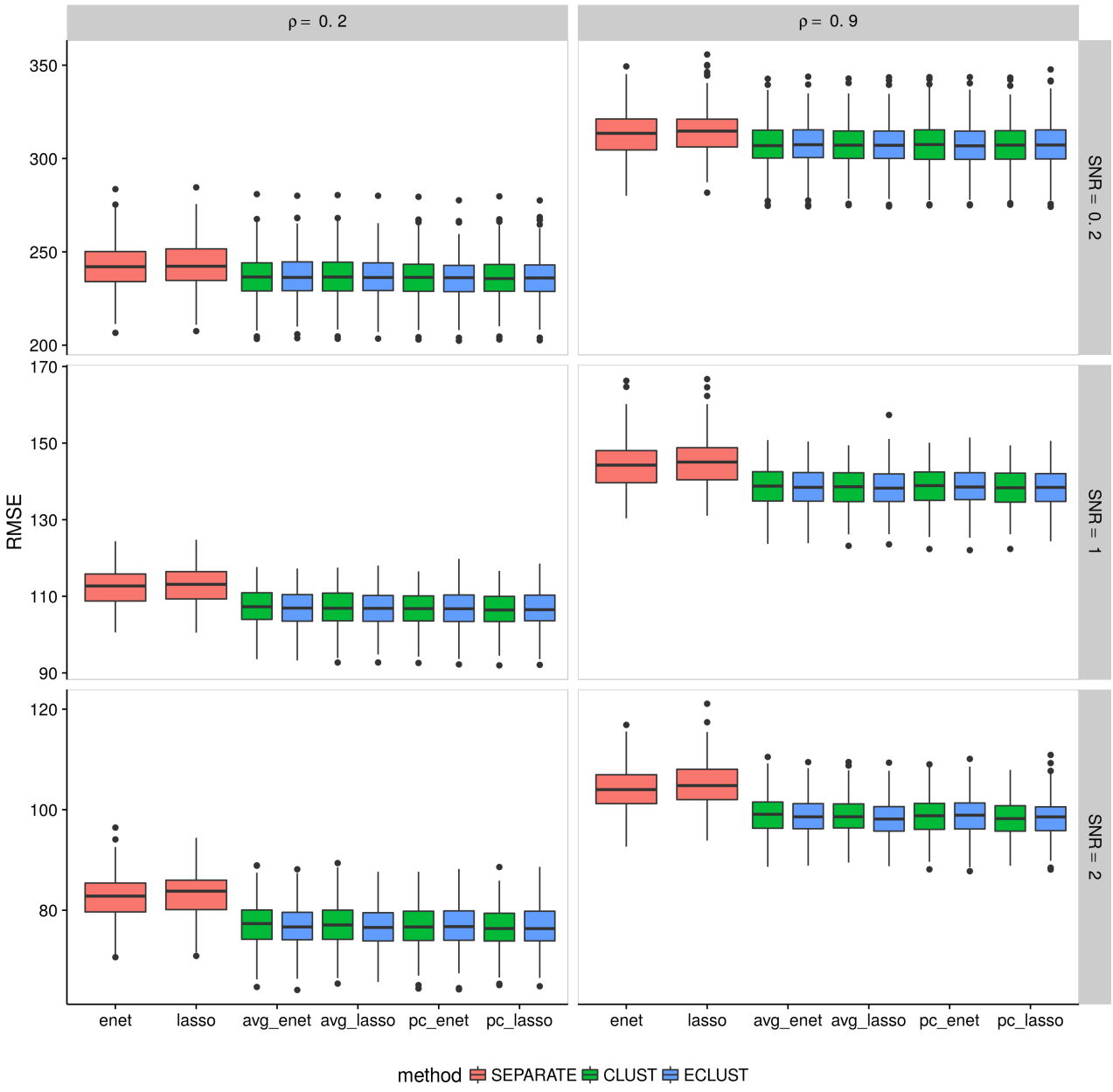


Figure C.26: Simulation 1 – Root mean squared error on an independent test set using the Correlation as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

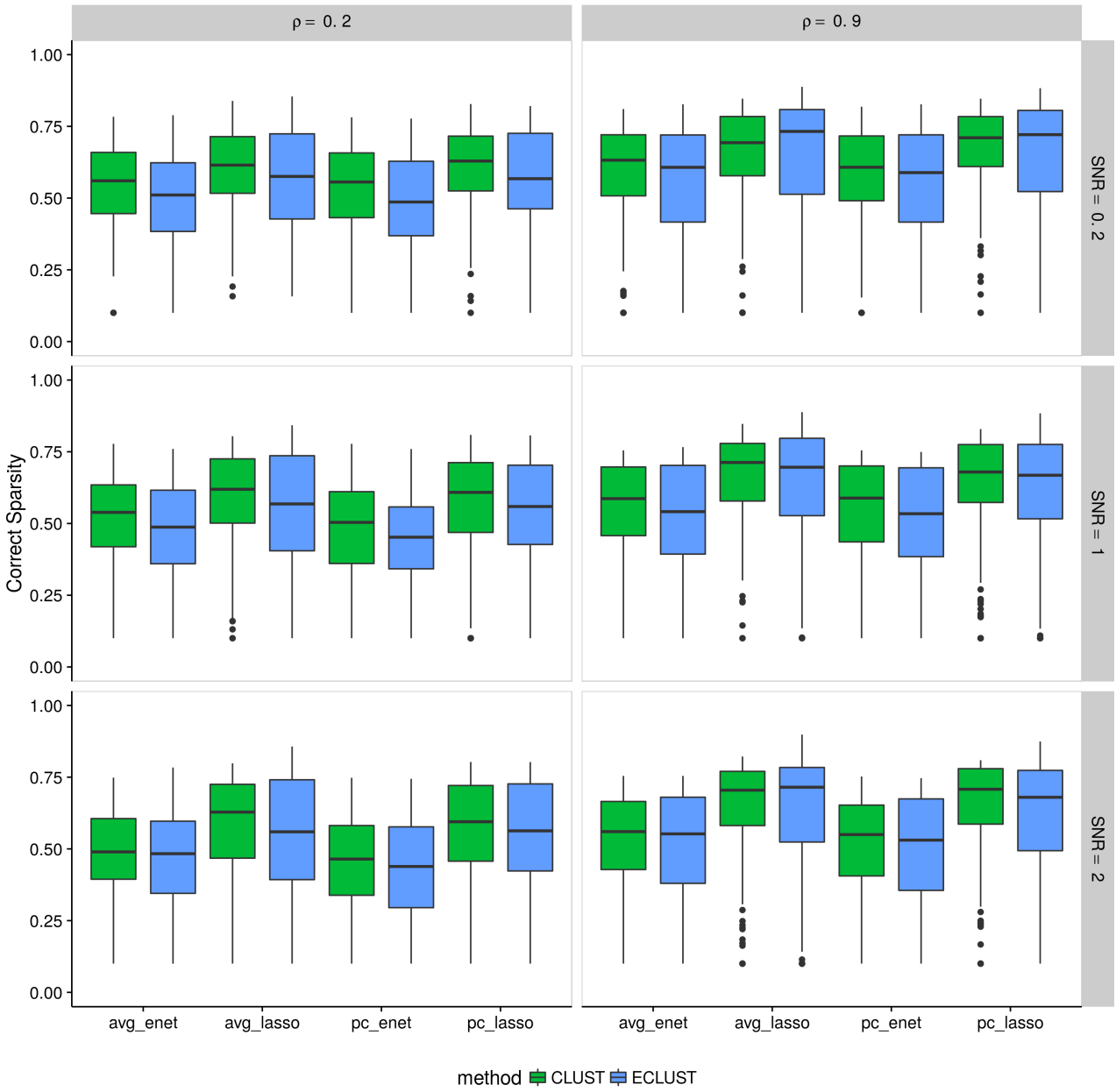


Figure C.27: Simulation 1 – Correct Sparsity based on the training set using the Pearson correlation as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

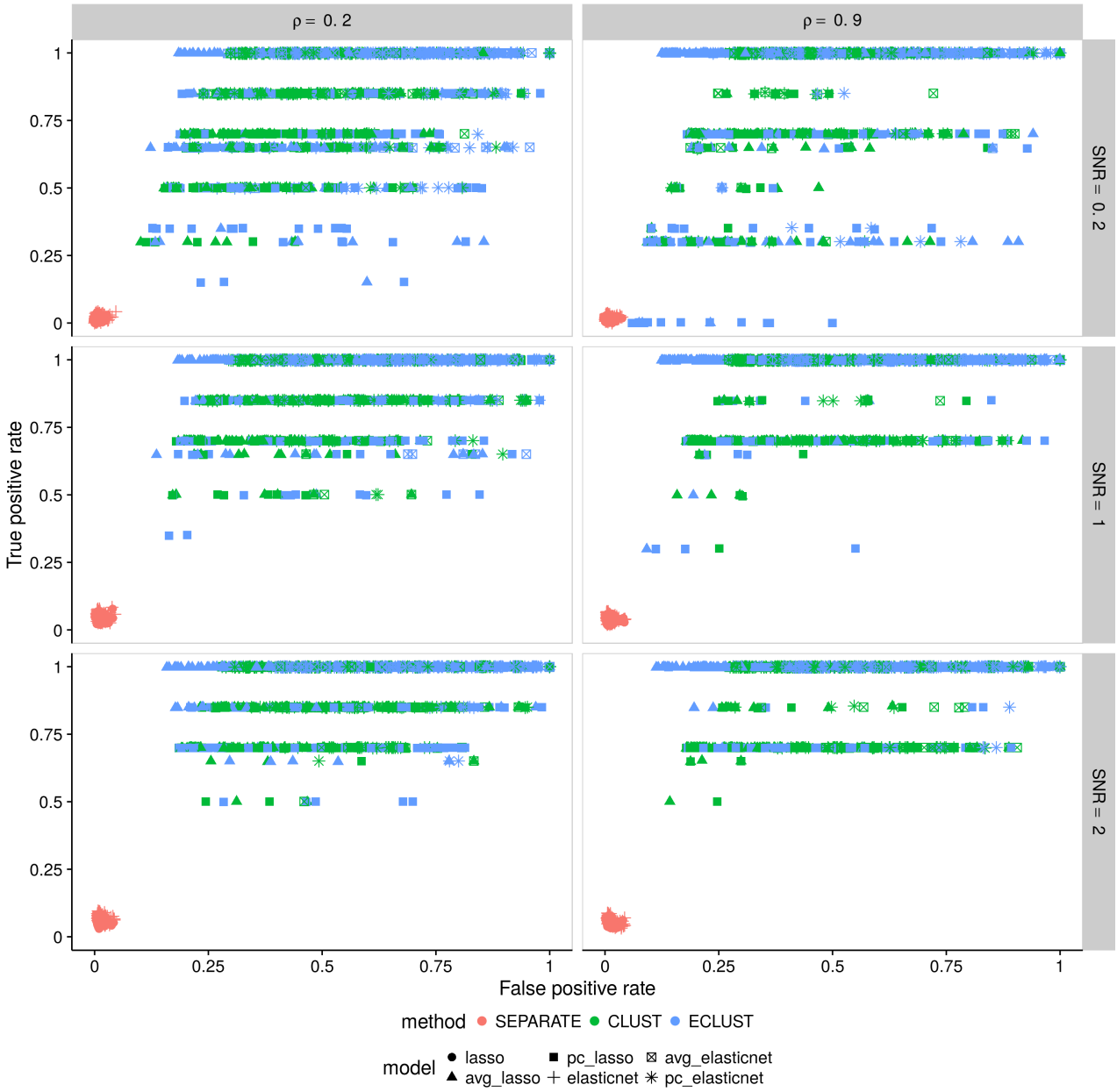


Figure C.28: Simulation 1 – True positive rate vs. false positive rate based on the training set using the Pearson correlation as a measure of similarity. Each point represents 1 simulation run (there are a total of 200 simulation runs). Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

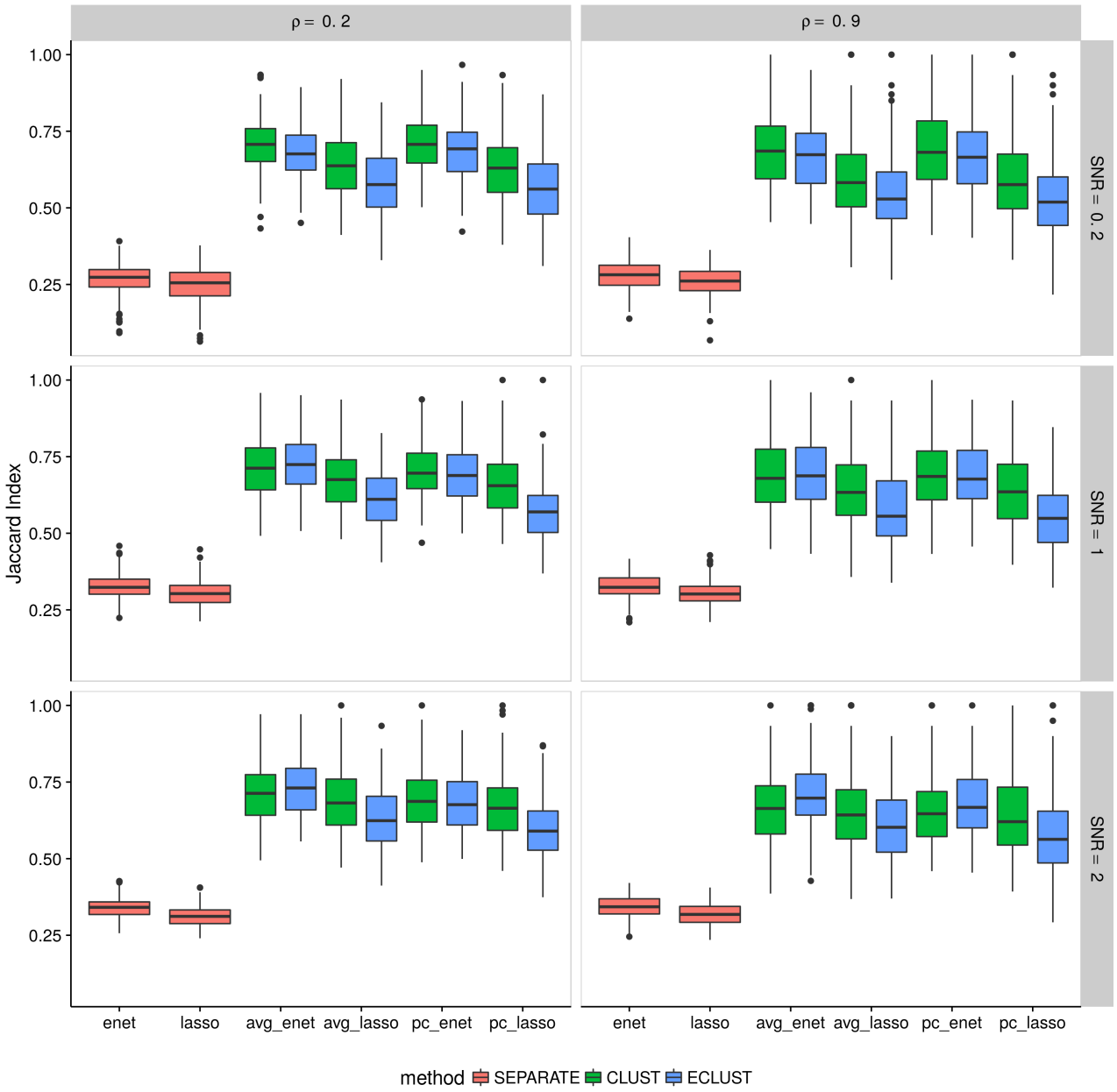


Figure C.29: Simulation 1 – Average Jaccard Index from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of selected predictors. We then calculate the Jaccard Index between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

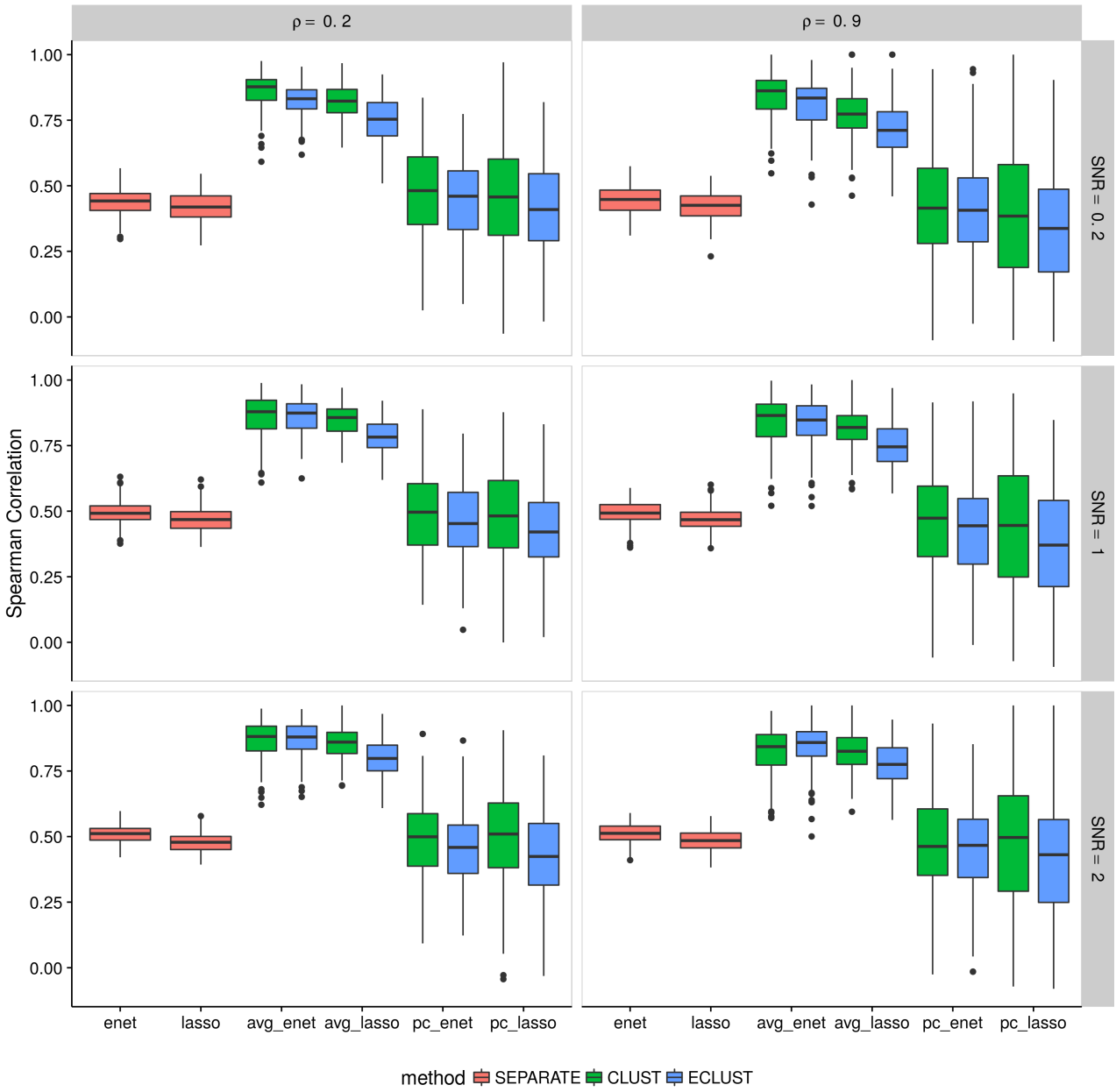


Figure C.30: Simulation 1 – Average Spearman correlation from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Spearman correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

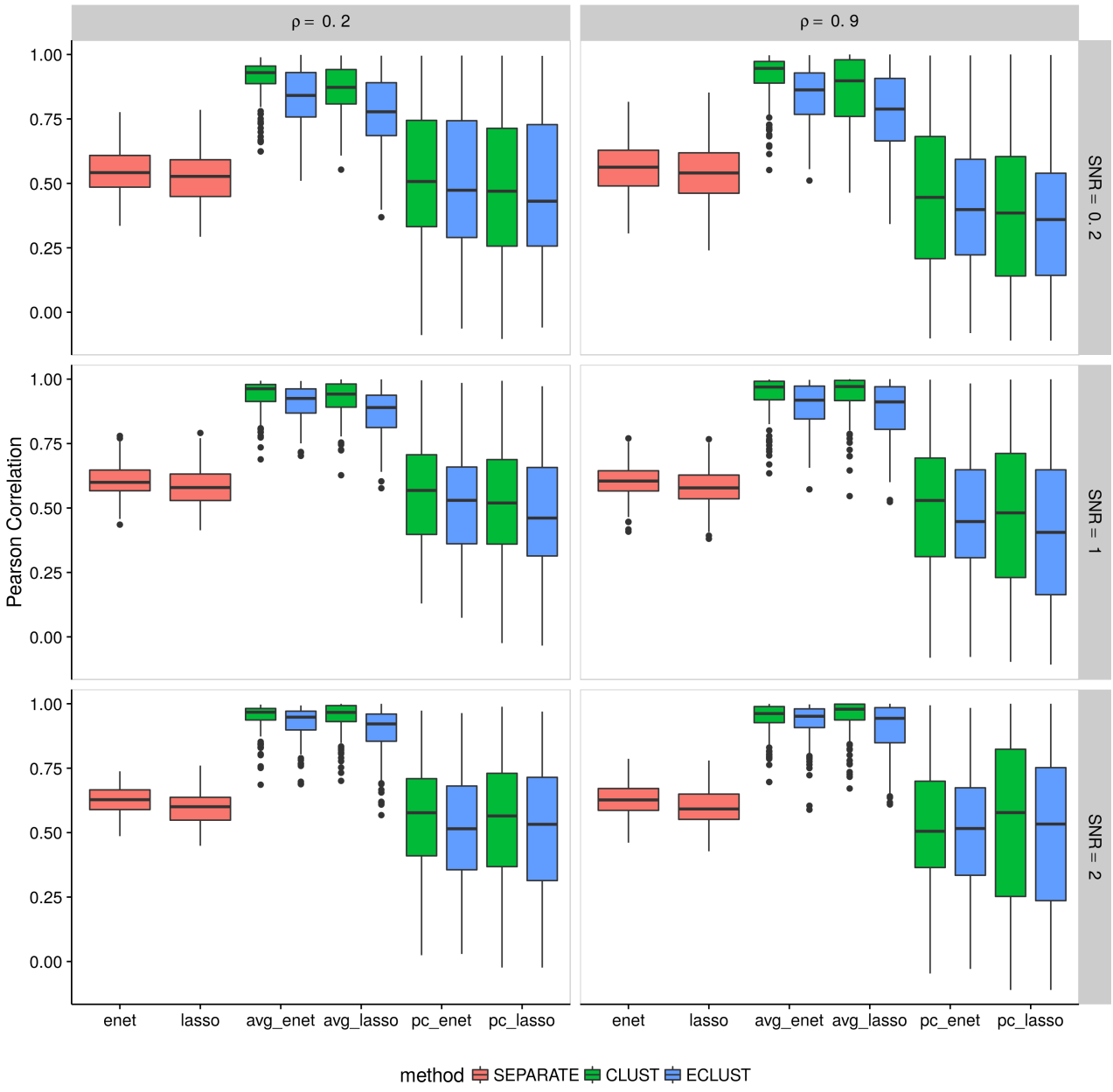


Figure C.31: Simulation 1 – Average Pearson correlation from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Pearson correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

Simulation 2

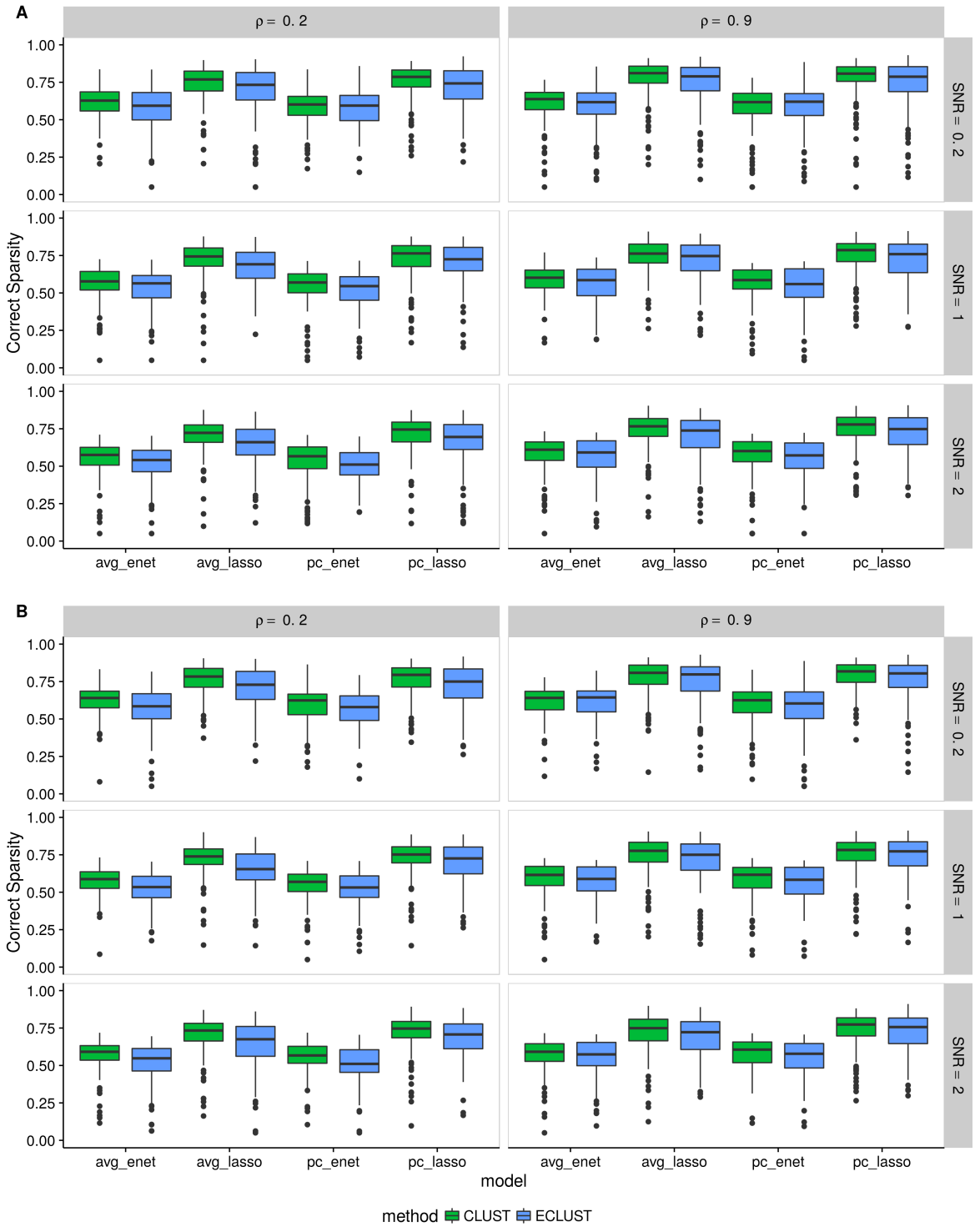


Figure C.33: Simulation 2 – Correct Sparsity based on the training set using the Pearson correlation as a measure of similarity from 200 simulation runs. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

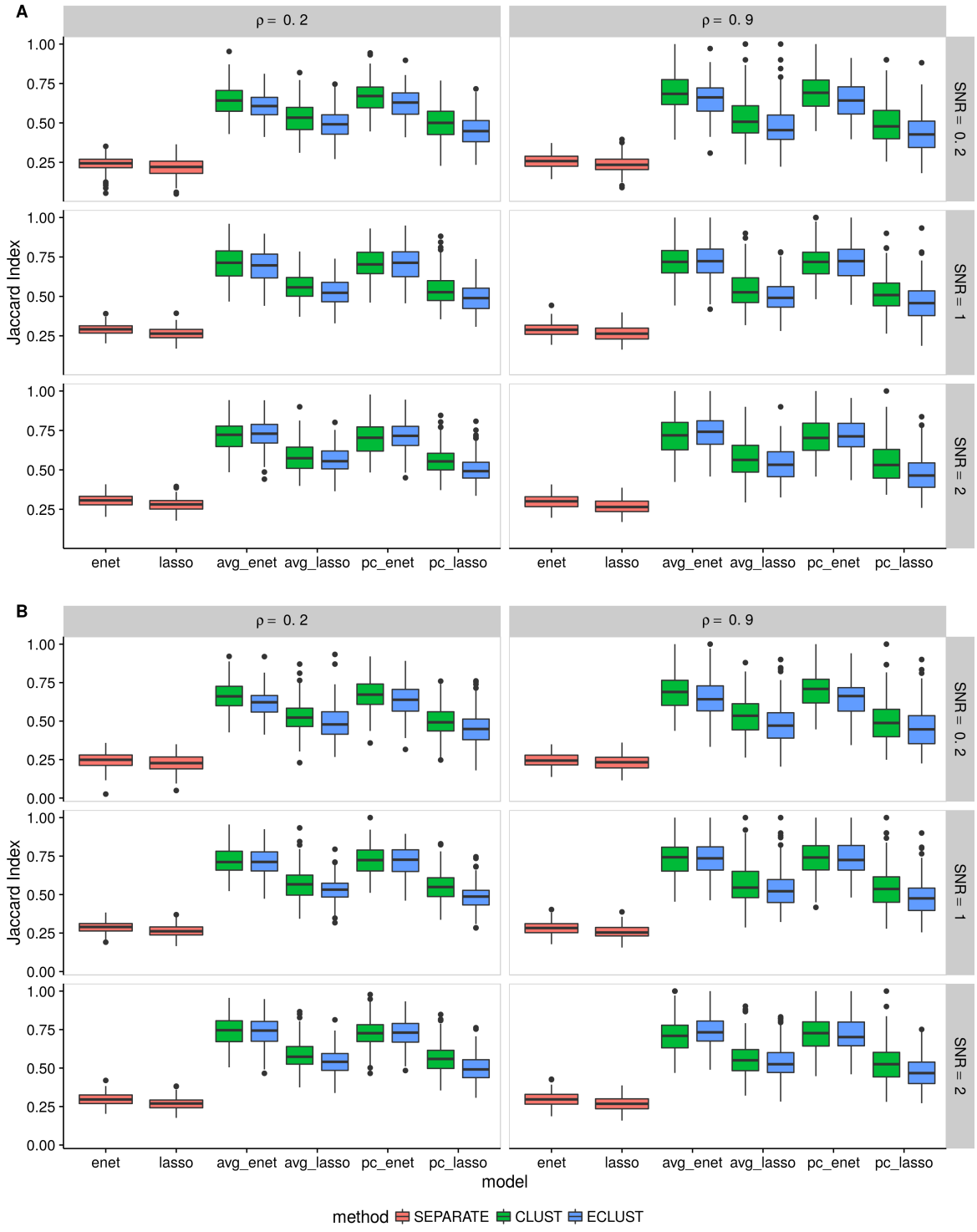


Figure C.35: Simulation 2 – Average Jaccard Index from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. We fit the model to each of the 10 CV folds resulting in 10 sets of selected predictors. We then calculate the Jaccard Index between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

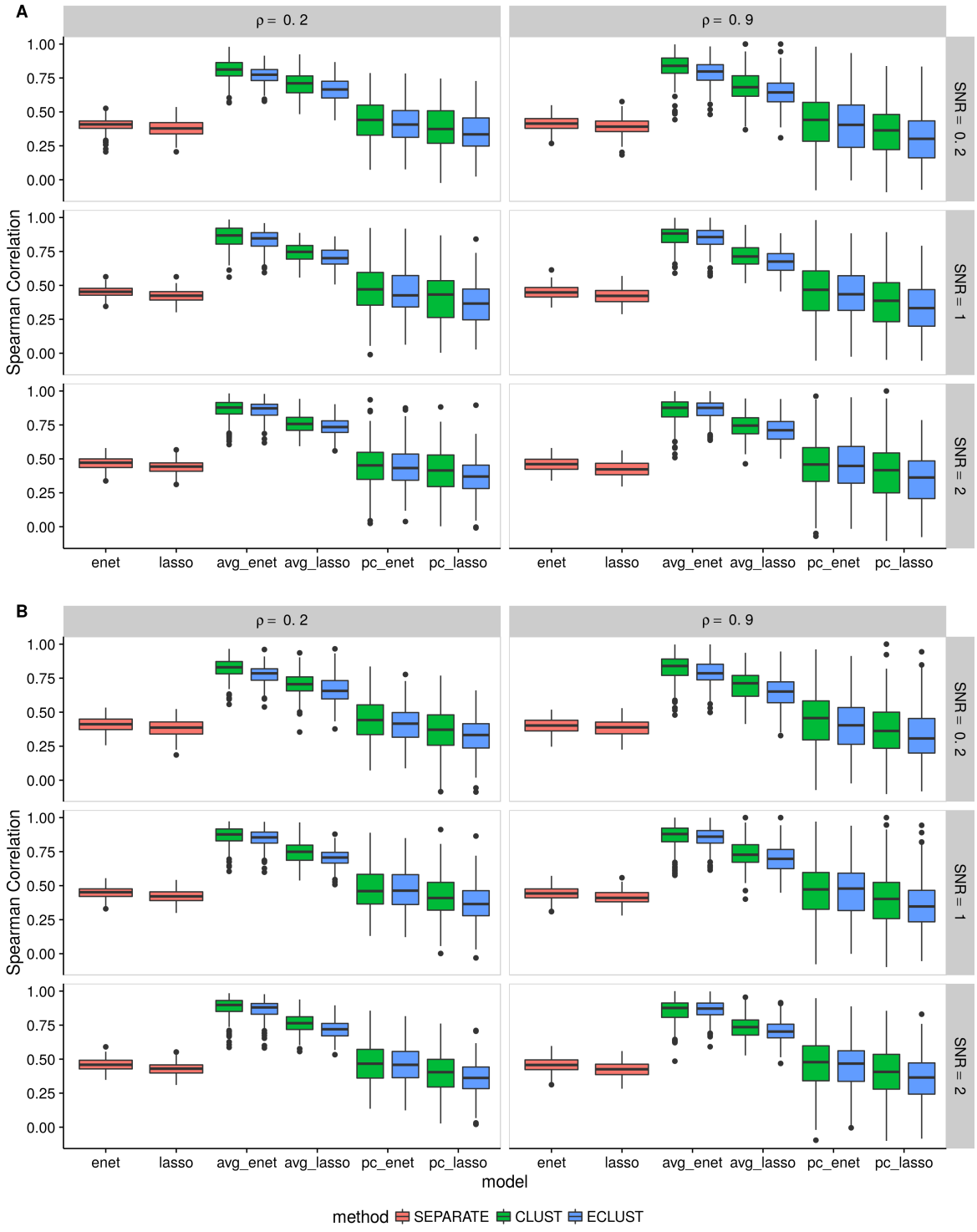


Figure C.36: Simulation 2 – Average Spearman correlation from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Spearman correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

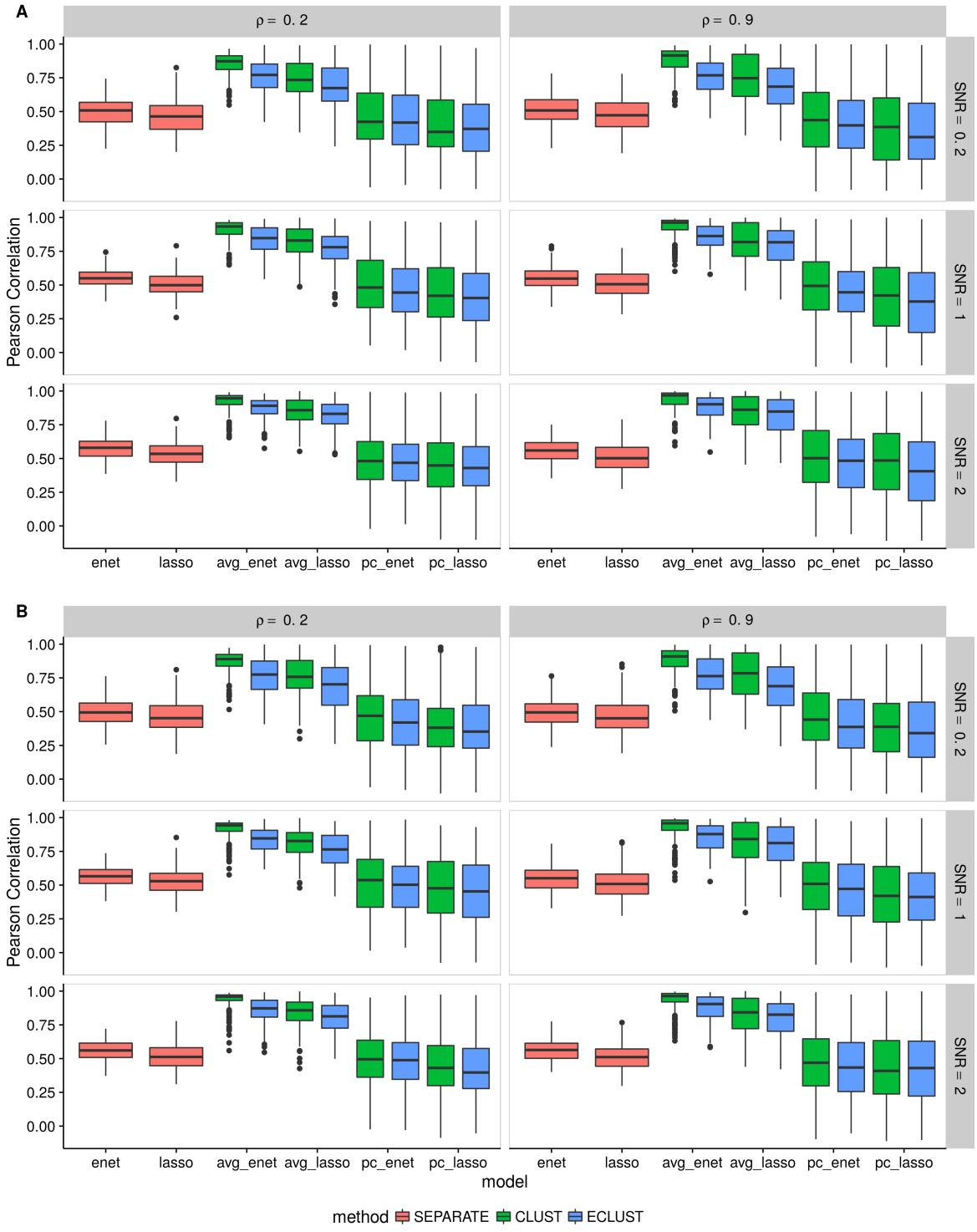


Figure C.37: Simulation 2 – Average Pearson correlation from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. We fit the model to each of the 10 CV folds resulting in 10 sets of estimated regression coefficients. We then calculate the Pearson correlation between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

Simulation 3

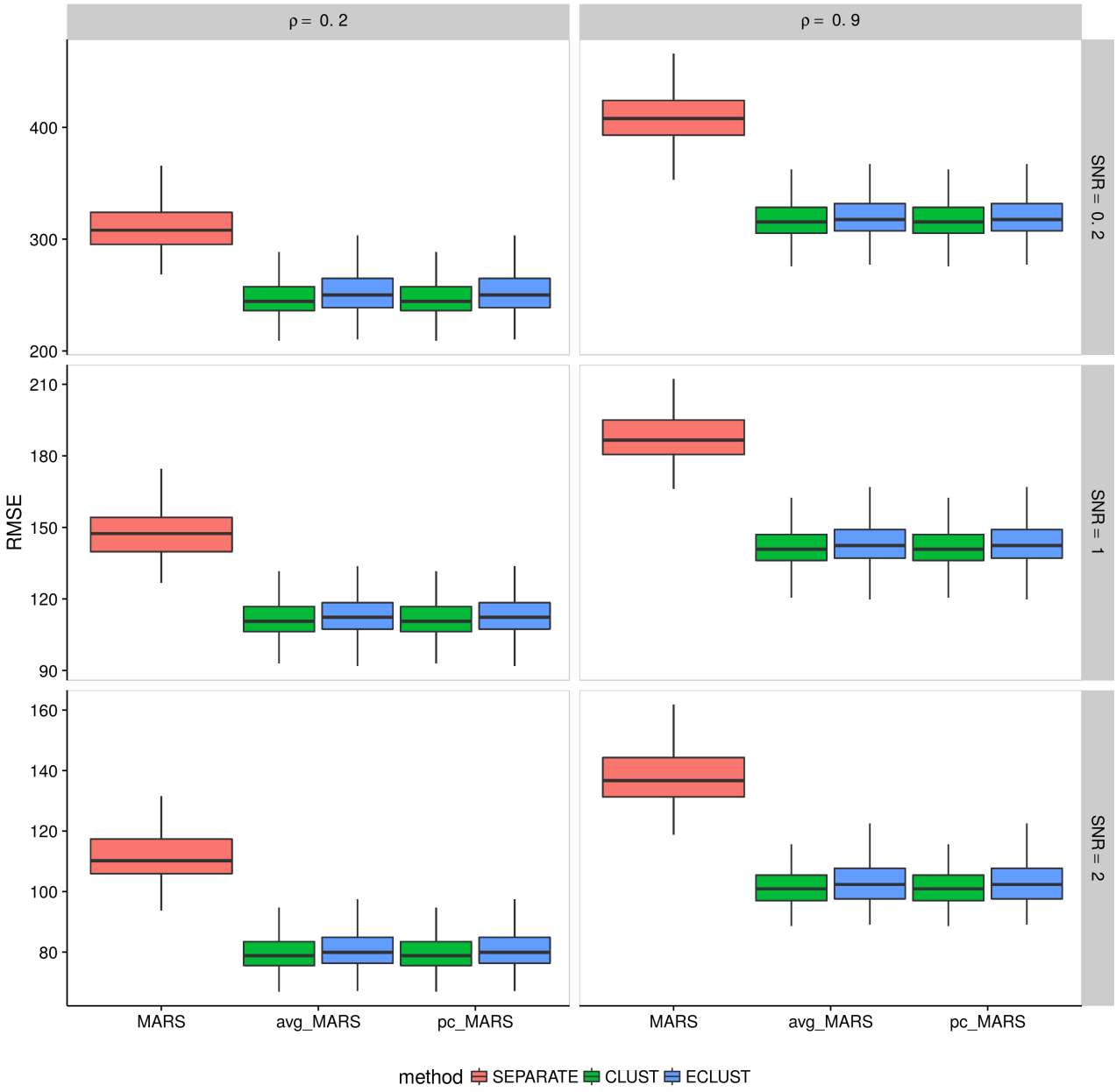


Figure C.38: Simulation 3 – Root mean squared error on an independent test set using the Pearson correlation as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

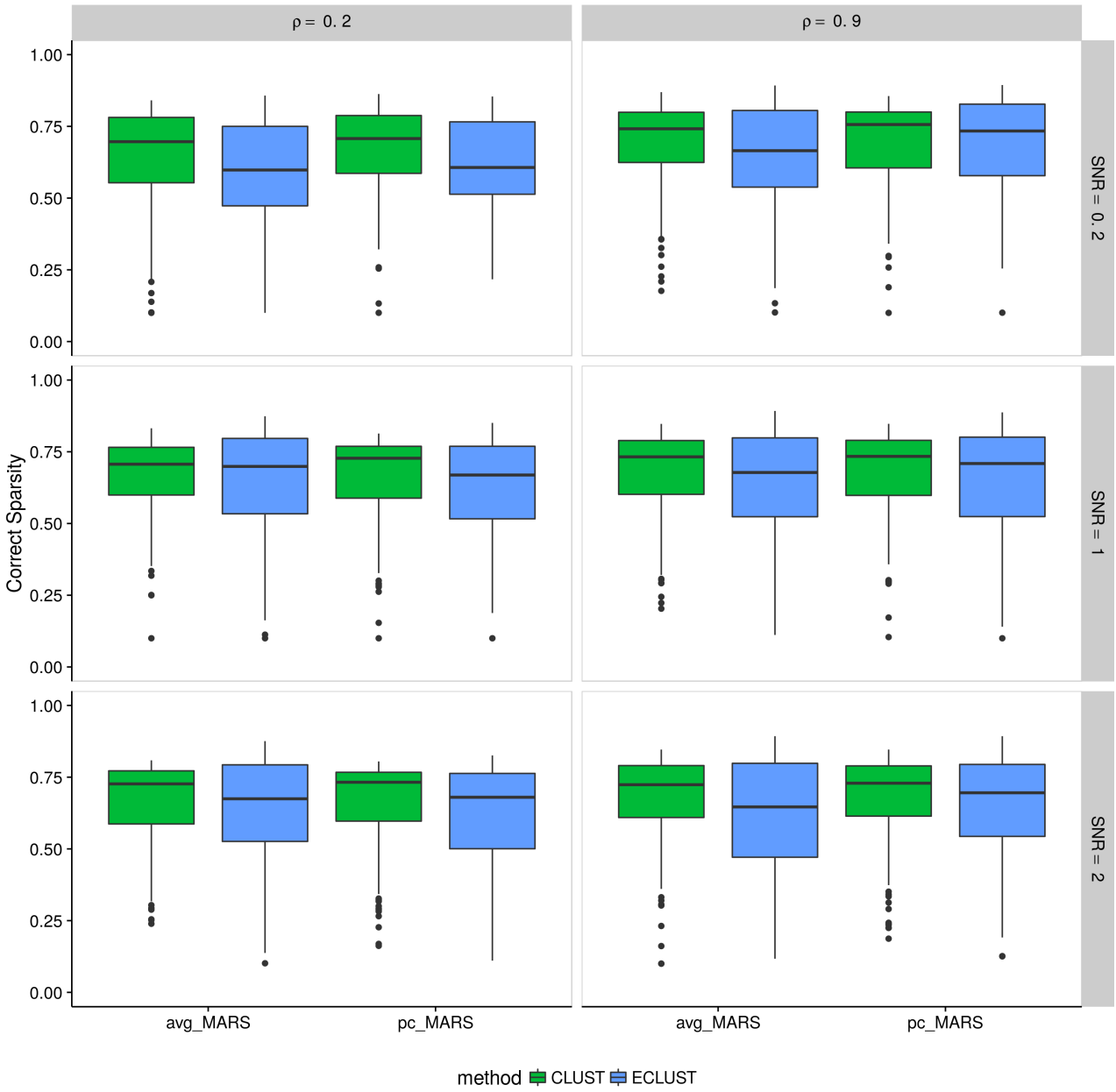


Figure C.39: Simulation 3 – Correct Sparsity based on the training set using the Pearson correlation as a measure of similarity from 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

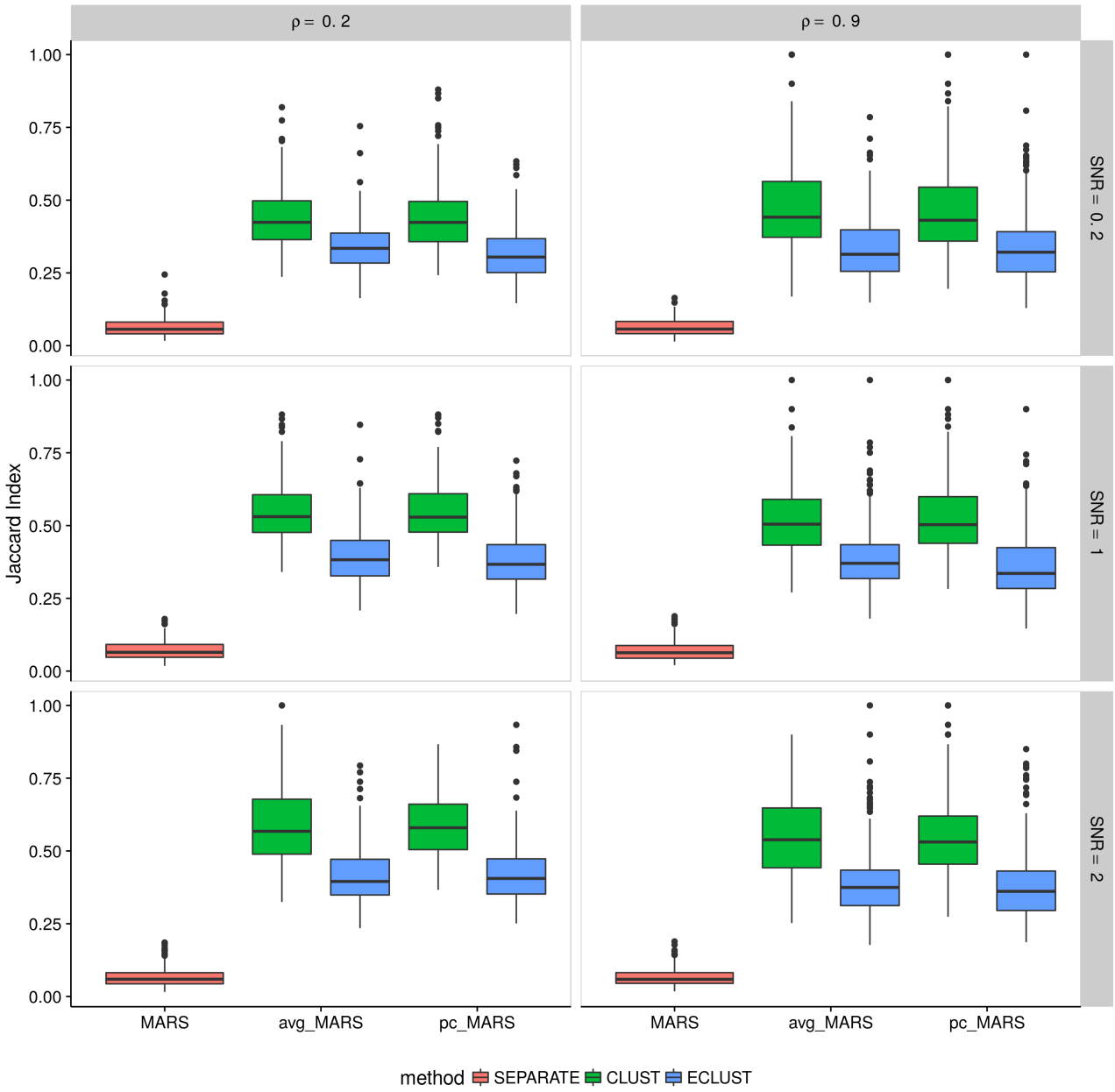
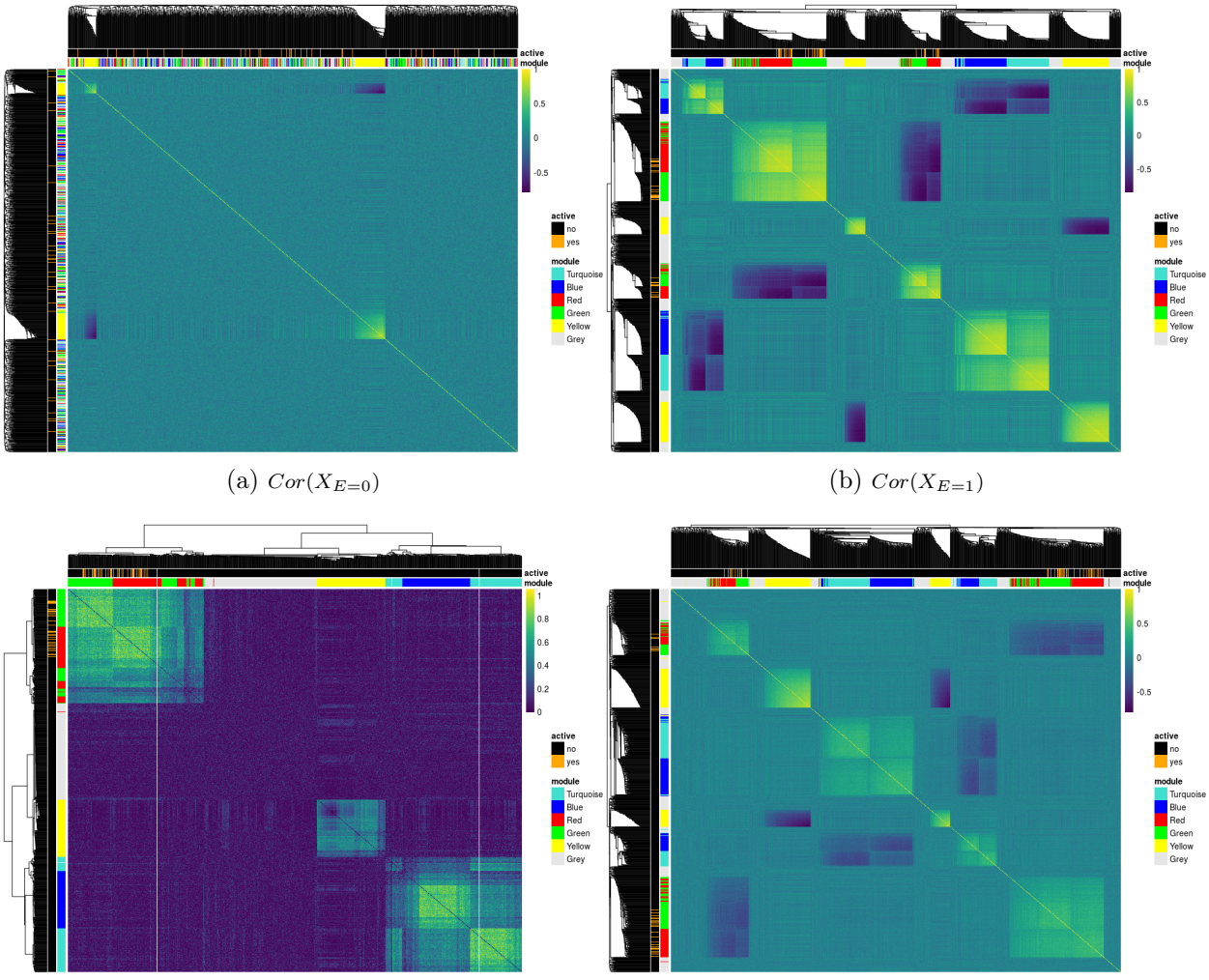


Figure C.41: Simulation 3 – Average Jaccard Index from 10 CV folds of the training set using the Pearson correlation as a measure of similarity. We fit the model to each of the 10 CV folds resulting in 10 sets of selected predictors. We then calculate the Jaccard Index between all $\binom{10}{2}$ possible combinations of these sets and take the average. This process is repeated for each of the 200 simulation runs. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

C.6 Visual Representation of Similarity Matrices

Pearson Correlation Matrix



(a) $Cor(X_{E=0})$

(b) $Cor(X_{E=1})$

(c) $|Cor(X_{E=1}) - Cor(X_{E=0})|$

(d) $Cor(X_{\text{all}})$

Figure C.42: Pearson correlation matrices of simulated predictors based on subjects with (a) $E = 0$, (b) $E = 1$, (c) their absolute difference and (d) all subjects. Dendrograms are from hierarchical clustering (average linkage) of one minus the correlation matrix for a, b, and d and the euclidean distance for c. The *module* annotation represents the true cluster membership for each predictor, and the *active* annotation represents the truly associated predictors with the response.

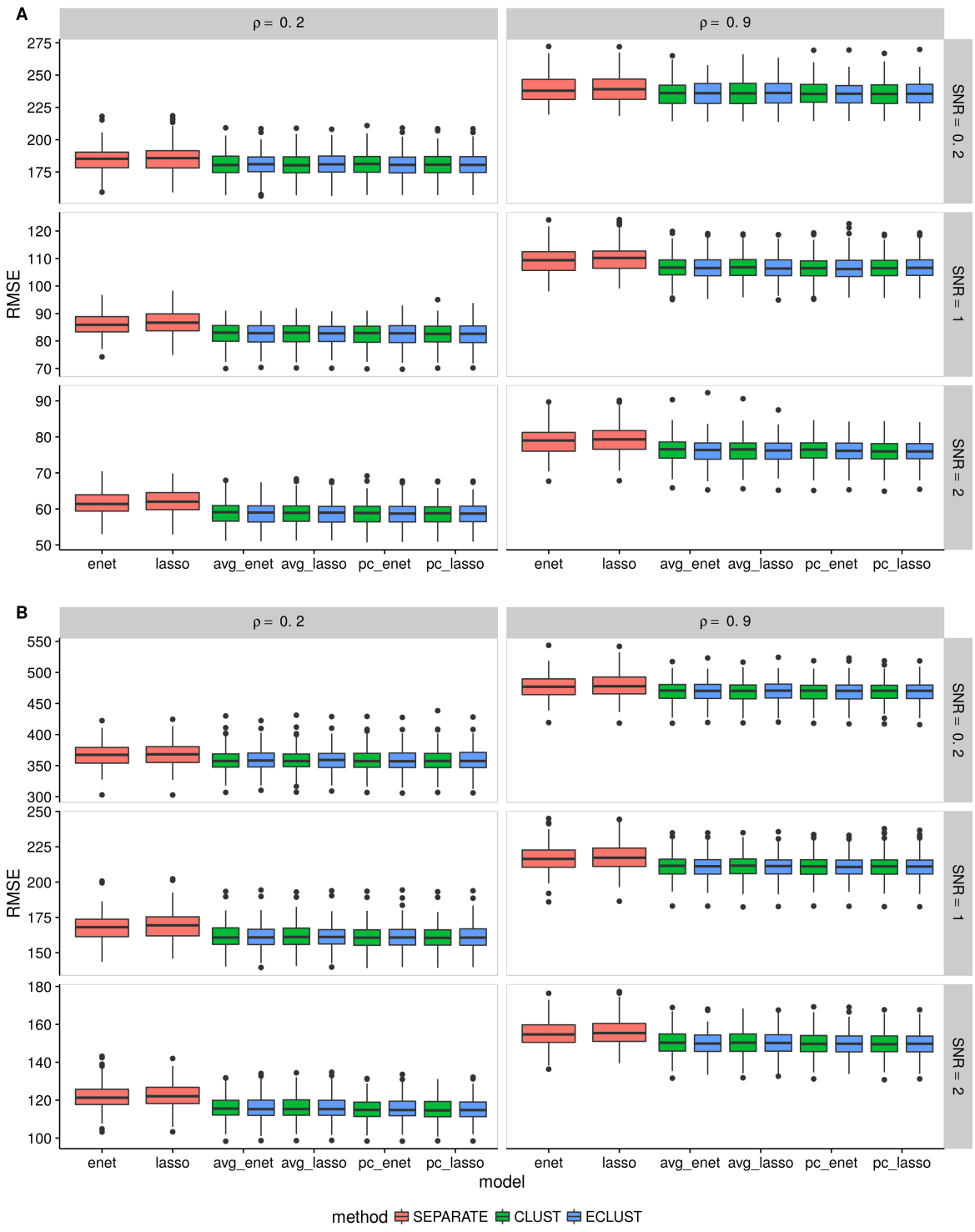


Figure C.32: Simulation 2 – Root mean squared error on an independent test set using the Pearson correlation as a measure of similarity from 200 simulation runs. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

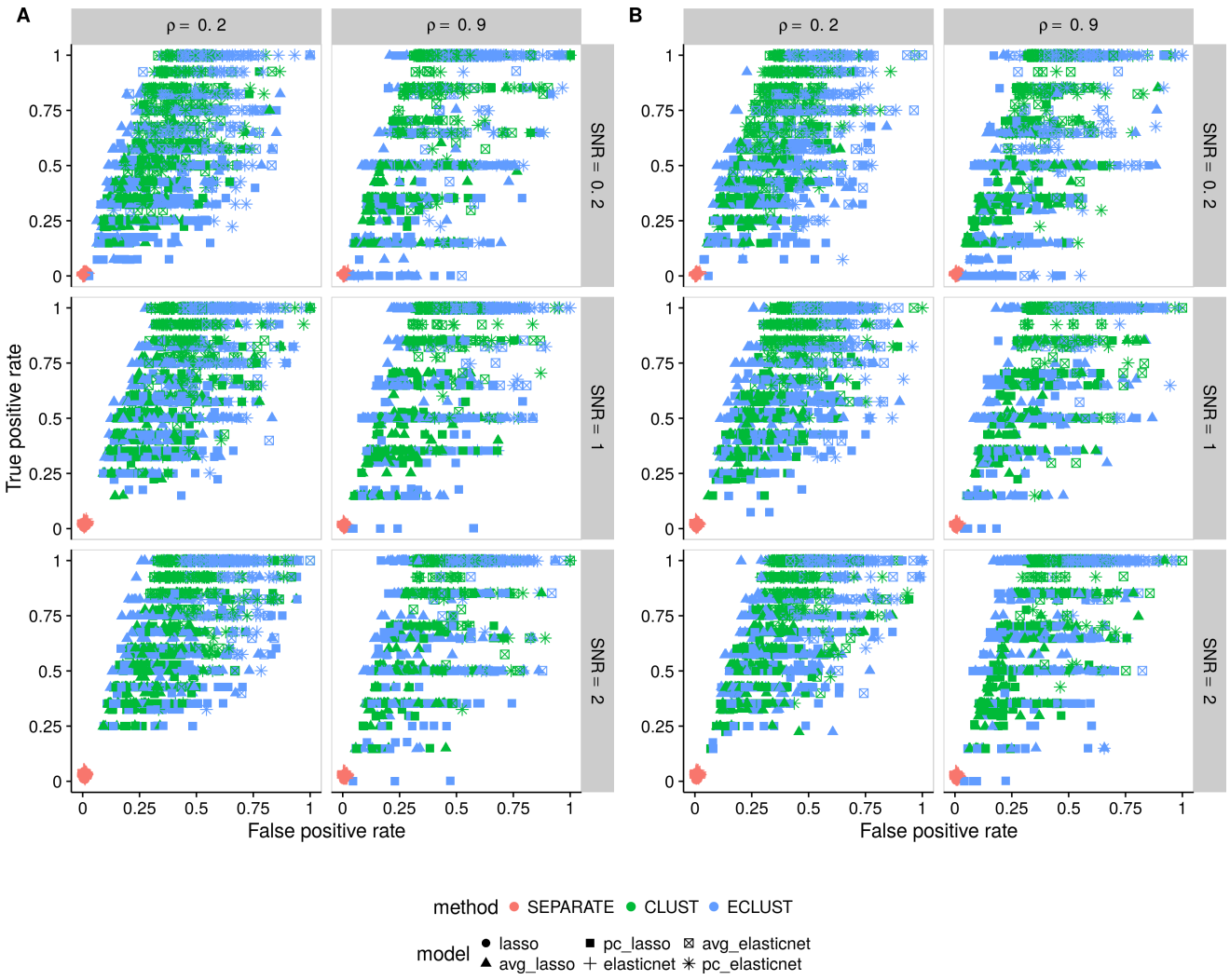


Figure C.34: Simulation 2 – True positive rate vs. false positive rate based on the training set using the Pearson correlation as a measure of similarity. (A) $\alpha_j \sim \text{Unif}\,[0.4, 0.6]$, (B) $\alpha_j \sim \text{Unif}\,[1.9, 2.1]$. Each point represents 1 simulation run (there are a total of 200 simulation runs). Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

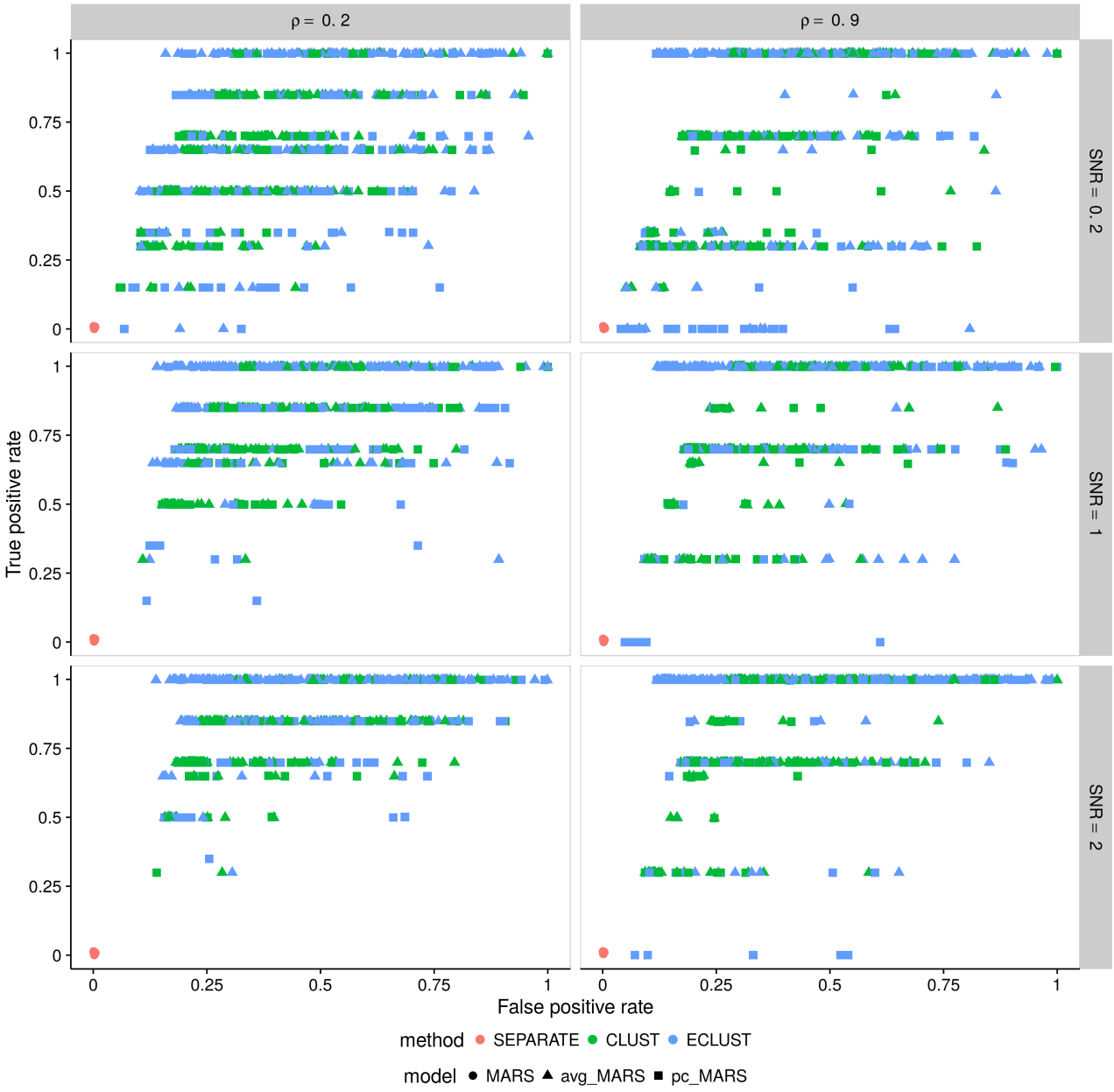


Figure C.40: Simulation 3 – True positive rate vs. false positive rate based on the training set using the Pearson correlation as a measure of similarity. Each point represents 1 simulation run (there are a total of 200 simulation runs). Vertical panels represent varying correlation between active clusters. Horizontal panels represent different signal-to-noise ratios.

References

Allen, N., Sudlow, C., Downey, P., Peakman, T., Danesh, J., Elliott, P., ... others (2012). Uk biobank: Current status and what it means for epidemiology. *Health Policy and Technology*, *1*(3), 123–126.

Astle, W., Balding, D. J., et al. (2009). Population structure and cryptic relatedness in genetic association studies. *Statistical Science*, *24*(4), 451–471.

Bach, F., Jenatton, R., Mairal, J., Obozinski, G., et al. (2012). Structured sparsity through convex optimization. *Statistical Science*, *27*(4), 450–468.

Bertsekas, D. P. (1999). *Nonlinear programming*. Athena scientific Belmont.

Bhatnagar, S. R. (2017). eclust: Environment based clustering for interpretable predictive models in high dimensional data [Computer software manual]. Retrieved from https://cran.r-project.org/package=eclust (R package version 0.1.0)

Bhatnagar, S. R., Yang, Y., Khundrakpam, B., Evans, A. C., Blanchette, M., Bouchard, L., & Greenwood, C. M. (2018). An analytic approach for interpretable predictive models in

high-dimensional data in the presence of interactions with exposures. *Genetic epidemiology*, *42*(3), 233–249.

Bibikova, M., Barnes, B., Tsan, C., Ho, V., Klotzle, B., Le, J. M., ... others (2011). High density dna methylation array with single cpg site resolution. *Genomics*, *98*(4), 288–295.

Bien, J., Taylor, J., Tibshirani, R., et al. (2013). A lasso for hierarchical interactions. *The Annals of Statistics*, *41*(3), 1111–1141.

Bondell, H. D., Krishna, A., & Ghosh, S. K. (2010). Joint variable selection for fixed and random effects in linear mixed-effects models. *Biometrics*, *66*(4), 1069–1077.

Bouchard, L., Hivert, M.-F., Guay, S.-P., St-Pierre, J., Perron, P., & Brisson, D. (2012). Placental adiponectin gene dna methylation levels are associated with mothers' blood glucose concentration. *Diabetes*, *61*(5), 1272–1280.

Bouchard, L., Thibault, S., Guay, S.-P., Santure, M., Monpetit, A., St-Pierre, J., ... Brisson, D. (2010). Leptin gene epigenetic adaptation to impaired glucose metabolism during pregnancy. *Diabetes care*, *33*(11), 2436–2441.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123–140.

Bühlmann, P., Rütimann, P., van de Geer, S., & Zhang, C.-H. (2013). Correlated variables in regression: clustering and sparse estimation. *Journal of Statistical Planning and Inference*, *143*(11), 1835–1858.

Bühlmann, P., & Van De Geer, S. (2011). *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, *16*(5), 1190–1208.

Chipman, H. (1996). Bayesian variable selection with related predictors. *Canadian Journal of Statistics*, *24*(1), 17–36.

Choi, N. H., Li, W., & Zhu, J. (2010). Variable selection with the strong heredity constraint and its oracle property. *Journal of the American Statistical Association*, *105*(489), 354–364.

Chouldechova, A., & Hastie, T. (2015). Generalized additive model selection. *arXiv preprint arXiv:1506.03850*.

Cordell, H. J., & Clayton, D. G. (2002). A unified stepwise regression procedure for evaluating the relative effects of polymorphisms within a gene using case/control or family data: application to hla in type 1 diabetes. *The American Journal of Human Genetics*, *70*(1), 124–141.

Cox, D. R. (1984). Interaction. *International Statistical Review/Revue Internationale de Statistique*, 1–24.

Dandine-Roulland, C. (2018). gaston: Genetic data handling (qc, grm, ld, pca) and linear mixed models [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=gaston (R package version 1.5.3)

Davey Smith, G., & Ebrahim, S. (2003). 'mendelian randomization': can genetic epidemiology contribute to understanding environmental determinants of disease? *International journal of epidemiology*, *32*(1), 1–22.

Ding, X., Su, S., Nandakumar, K., Wang, X., & Fardo, D. W. (2014). A 2-step penalized regression method for family-based next-generation sequencing association studies. In *Bmc proceedings* (Vol. 8, p. S25).

Dudbridge, F. (2013). Power and predictive accuracy of polygenic risk scores. *PLoS genetics*, *9*(3), e1003348.

Efron, B. (1983). Estimating the error rate of a prediction rule: improvement on cross-validation. *Journal of the American Statistical Association*, *78*(382), 316–331.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al. (2004). Least angle regression. *The Annals of statistics*, *32*(2), 407–499.

Ehrenberg, H. M., Mercer, B. M., & Catalano, P. M. (2004). The influence of obesity and diabetes on the prevalence of macrosomia. *American journal of obstetrics and gynecology*, *191*(3), 964–968.

Epstein, L. H., Paluch, R. A., Kilanowski, C. K., & Raynor, H. A. (2004). The effect of reinforcement or stimulus control to reduce sedentary behavior in the treatment of pediatric obesity. *Health Psychology*, *23*(4), 371.

Eu-Ahsunthornwattana, J., Miller, E. N., Fakiola, M., Jeronimo, S. M., Blackwell, J. M., Cordell, H. J., ... others (2014). Comparison of methods to account for relatedness in genome-wide association studies with family-based data. *PLoS Genet*, *10*(7), e1004445.

Evans, A. C., Group, B. D. C., et al. (2006). The nih mri study of normal brain development. *Neuroimage*, *30*(1), 184–202.

Fan, J., Han, F., & Liu, H. (2014). Challenges of big data analysis. *National science review*, *1*(2), 293–314.

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, *96*(456), 1348–1360.

Fan, Y., & Tang, C. Y. (2013). Tuning parameter selection in high dimensional penalized likelihood. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *75*(3), 531–552.

Felson, D. T., Zhang, Y., Hannan, M. T., & Anderson, J. J. (1993). Effects of weight and body mass index on bone mineral density in men and women: the framingham study. *Journal of Bone and Mineral Research*, *8*(5), 567–573.

Friedman, J., Hastie, T., Höfling, H., Tibshirani, R., et al. (2007). Pathwise coordinate optimization. *The Annals of Applied Statistics*, *1*(2), 302–332.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). Springer series in statistics Springer, Berlin.

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, *33*(1), 1.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, 1–67.

Gibson, G. (2009). Decanalization and the origin of complex disease. *Nature reviews. Genetics*, *10*(2), 134.

Gilmour, A. R., Thompson, R., & Cullis, B. R. (1995). Average information reml: an efficient algorithm for variance parameter estimation in linear mixed models. *Biometrics*, 1440–1450.

Hao, N., Feng, Y., & Zhang, H. H. (2018). Model selection for high-dimensional quadratic regression via regularization. *Journal of the American Statistical Association*, 1–11.

Haris, A., Shojaie, A., & Simon, N. (2016). Nonparametric regression with adaptive truncation via a convex hierarchical penalty. *arXiv preprint arXiv:1611.09972*.

Haris, A., Witten, D., & Simon, N. (2014). Convex modeling of interactions with strong heredity. *arXiv preprint arXiv:1410.3517*.

Haris, A., Witten, D., & Simon, N. (2016). Convex modeling of interactions with strong heredity. *Journal of Computational and Graphical Statistics*, *25*(4), 981–1004.

Hastie, T., & Tibshirani, R. (1987). Generalized additive models: some applications. *Journal of the American Statistical Association*, *82*(398), 371–386.

Hastie, T., Tibshirani, R., Botstein, D., & Brown, P. (2001). Supervised harvesting of expression trees. *Genome Biology*, *2*(1), 1–0003.

Hastie, T., Tibshirani, R., & Wainwright, M. (2015). *Statistical learning with sparsity: The lasso and generalizations*. CRC Press.

Helland, Å., Anglesio, M. S., George, J., Cowin, P. A., Johnstone, C. N., House, C. M., ... others (2011). Deregulation of mycn, lin28b and let7 in a molecular subtype of aggressive high-grade serous ovarian cancers. *PloS one*, *6*(4), e18064.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.

Hoggart, C. J., Whittaker, J. C., De Iorio, M., & Balding, D. J. (2008). Simultaneous analysis of all snps in genome-wide and re-sequencing association studies. *PLoS genetics*, *4*(7), e1000130.

Hosmer, D. W., Hosmer, T., Le Cessie, S., Lemeshow, S., et al. (1997). A comparison of goodness-of-fit tests for the logistic regression model. *Statistics in medicine*, *16*(9), 965–980.

Huang, J., Horowitz, J. L., & Wei, F. (2010). Variable selection in nonparametric additive models. *Annals of statistics*, *38*(4), 2282.

Jaccard, P. (1912). The distribution of the flora in the alpine zone. *New phytologist*, *11*(2), 37–50.

Jacob, L., Obozinski, G., & Vert, J.-P. (2009). Group lasso with overlap and graph lasso. In *Proceedings of the 26th annual international conference on machine learning* (pp. 433–440).

Jenatton, R., Obozinski, G., & Bach, F. (2009). Structured sparse principal component analysis. *arXiv preprint arXiv:0909.1440*.

Kalousis, A., Prados, J., & Hilario, M. (2007). Stability of feature selection algorithms: a study on high-dimensional spaces. *Knowledge and information systems*, *12*(1), 95–116.

Kanehisa, M., Araki, M., Goto, S., Hattori, M., Hirakawa, M., Itoh, M., ... others (2008). Kegg for linking genomes to life and the environment. *Nucleic acids research*, *36*(suppl 1), D480–D484.

Kang, H. M., Sul, J. H., Zaitlen, N. A., Kong, S.-y., Freimer, N. B., Sabatti, C., ... others (2010). Variance component model to account for sample structure in genome-wide association studies. *Nature genetics*, *42*(4), 348.

Kemp, J. P., Morris, J. A., Medina-Gomez, C., Forgetta, V., Warrington, N. M., Youlten, S. E., ... others (2017). Identification of 153 new loci associated with heel bone mineral density and functional involvement of gpc6 in osteoporosis. *Nature genetics*, *49*, 1468.

Kendall, M. (1957). *A course in multivariate analysis*. London: Griffin.

Khundrakpam, B. S., Reid, A., Brauer, J., Carbonell, F., Lewis, J., Ameis, S., ... others (2013). Developmental changes in organization of structural brain networks. *Cerebral Cortex*, *23*(9), 2072–2085.

Klein, K. O., Oualkacha, K., Lafond, M.-H., Bhatnagar, S., Tonin, P. N., & Greenwood, C. M. (2016). Gene coexpression analyses differentiate networks associated with diverse cancers harboring tp53 missense or null mutations. *Frontiers in Genetics*, *7*.

Kostka, D., & Spang, R. (2004). Finding disease specific alterations in the co-expression of genes. *Bioinformatics*, *20*(suppl 1), i194–i199.

Kuhn, M. (2008). Caret package. *Journal of Statistical Software*, *28*(5).

Kvålseth, T. O. (1985). Cautionary note about r 2. *The American Statistician*, *39*(4), 279–285.

Laird, N. M., & Ware, J. H. (1982). Random-effects models for longitudinal data. *Biometrics*, 963–974.

Lange, K., Hunter, D., & Yang, I. (2000). Optimization transfer using sur- rogate objective functions (with discussion). *Journal of Computational and Graphical Statistics*, *9*, 1-20.

Langfelder, P., & Horvath, S. (2007). Eigengene networks for studying the relationships between co-expression modules. *BMC systems biology*, *1*(1), 54.

Langfelder, P., & Horvath, S. (2008). Wgcna: an r package for weighted correlation network analysis. *BMC bioinformatics*, *9*(1), 1.

Langfelder, P., Zhang, B., & Horvath, S. (2008). Defining clusters from a hierarchical cluster tree: the dynamic tree cut package for r. *Bioinformatics*, *24*(5), 719–720.

Langfelder, P., Zhang, B., & with contributions from Steve Horvath. (2016). dynamictreecut: Methods for detection of clusters in hierarchical clustering dendrograms [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=dynamicTreeCut (R package version 1.63-1)

Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems* (pp. 556–562).

Leek, J. T., & Storey, J. D. (2008). A general framework for multiple testing dependence. *Proceedings of the National Academy of Sciences*, *105*(48), 18718–18723.

Li, J., Das, K., Fu, G., Li, R., & Wu, R. (2010). The bayesian lasso for genome-wide association studies. *Bioinformatics*, *27*(4), 516–523.

Lim, M., & Hastie, T. (2014). Learning interactions via hierarchical group-lasso regularization. *Journal of Computational and Graphical Statistics*(just-accepted), 00–00.

Lim, M., & Hastie, T. (2015). Learning interactions via hierarchical group-lasso regularization. *Journal of Computational and Graphical Statistics*, *24*(3), 627–654.

Lin, X., Lee, S., Christiani, D. C., & Lin, X. (2013). Test for interactions between a genetic marker set and environment in generalized linear models. *Biostatistics*, kxt006.

Lin, Y., Zhang, H. H., et al. (2006). Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, *34*(5), 2272–2297.

Lippert, C., Listgarten, J., Liu, Y., Kadie, C. M., Davidson, R. I., & Heckerman, D. (2011). Fast linear mixed models for genome-wide association studies. *Nature methods*, *8*(10), 833–835.

Manolio, T. A., Collins, F. S., Cox, N. J., Goldstein, D. B., Hindorff, L. A., Hunter, D. J., ... others (2009). Finding the missing heritability of complex diseases. *Nature*, *461*(7265), 747–753.

Marchini, J., Cardon, L. R., Phillips, M. S., & Donnelly, P. (2004). The effects of human population structure on large genetic association studies. *Nature genetics*, *36*(5), 512.

McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models* (Vol. 37). CRC press.

Meier, L., Van De Geer, S., & Bühlmann, P. (2008). The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *70*(1), 53–71.

Meinshausen, N. (2007). Relaxed lasso. *Computational Statistics & Data Analysis*, *52*(1), 374–393.

Milborrow. Derived from mda:mars by T. Hastie and R. Tibshirani., S. (2011). earth: Multivariate adaptive regression splines [Computer software manual]. Retrieved from http://CRAN.R-project.org/package=earth (R package)

Müllner, D. (2013). fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python. *Journal of Statistical Software*, *53*(9), 1–18. Retrieved from http://www.jstatsoft.org/v53/i09/

Ning, K., Chen, B., Sun, F., Hobel, Z., Zhao, L., Matloff, W., ... others (2018). Classifying alzheimer's disease with brain imaging and genetic data using a neural network framework. *Neurobiology of aging*.

Nishii, R. (1984). Asymptotic properties of criteria for selection of variables in multiple regression. *The Annals of Statistics*, 758–765.

Ochoa, A., & Storey, J. D. (2016a). FST and kinship for arbitrary population structures I: Generalized definitions. *bioRxiv*. doi: 10.1101/083915

Ochoa, A., & Storey, J. D. (2016b). FST and kinship for arbitrary population structures II: Method of moments estimators. *bioRxiv*. doi: 10.1101/083923

Osborne, M. R., Presnell, B., & Turlach, B. A. (2000). A new approach to variable selection in least squares problems. *IMA journal of numerical analysis*, *20*(3), 389–403.

Oualkacha, K., Dastani, Z., Li, R., Cingolani, P. E., Spector, T. D., Hammond, C. J., ... Greenwood, C. M. (2013). Adjusted sequence kernel association test for rare variants controlling for cryptic and family relatedness. *Genetic epidemiology*, *37*(4), 366–376.

Park, M. Y., Hastie, T., & Tibshirani, R. (2007). Averaged gene expressions for regression. *Biostatistics*, *8*(2), 212–227.

Pearson, K. (1895). Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 240–242.

Petersen, R. C., Aisen, P., Beckett, L., Donohue, M., Gamst, A., Harvey, D., ... others (2010). Alzheimer's disease neuroimaging initiative (adni) clinical characterization. *Neurology*, *74*(3), 201–209.

Pirinen, M., Donnelly, P., Spencer, C. C., et al. (2013). Efficient computation with a linear mixed model on large-scale data sets with applications to genetic studies. *The Annals of Applied Statistics*, *7*(1), 369–390.

Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., & Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nature genetics*, *38*(8), 904.

Qiu, H., Han, F., Liu, H., & Caffo, B. (2016). Joint estimation of multiple graphical models from high dimensional time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *78*(2), 487–504.

R Core Team. (2016). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria. Retrieved from https://www.R-project.org/

Radchenko, P., & James, G. M. (2010). Variable selection using adaptive nonlinear interaction structures in high dimensions. *Journal of the American Statistical Association*, *105*(492), 1541–1553.

Rakitsch, B., Lippert, C., Stegle, O., & Borgwardt, K. (2013). A lasso multi-marker mixed model for association mapping with population structure correction. *Bioinformatics*, *29*(2), 206–214.

Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N., & Barabási, A.-L. (2002). Hierarchical organization of modularity in metabolic networks. *science*, *297*(5586), 1551–1555.

Ravikumar, P., Lafferty, J., Liu, H., & Wasserman, L. (2009). Sparse additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *71*(5), 1009–1030.

Reid, S., & Tibshirani, R. (2016). Sparse regression and marginal testing using cluster prototypes. *Biostatistics*, *17*(2), 364–376.

Reid, S., Tibshirani, R., & Friedman, J. (2016). A study of error variance estimation in lasso regression. *Statistica Sinica*, 35–67.

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C., & Müller, M. (2011). proc: an open-source package for r and s+ to analyze and compare roc curves. *BMC Bioinformatics*, *12*, 77.

Ruchat, S.-M., Houde, A.-A., Voisin, G., St-Pierre, J., Perron, P., Baillargeon, J.-P., ... Bouchard, L. (2013). Gestational diabetes mellitus epigenetically affects genes predominantly involved in metabolic diseases. *Epigenetics*, *8*(9), 935–943.

Saltiel, A. R., & Kahn, C. R. (2001). Insulin signalling and the regulation of glucose and lipid metabolism. *Nature*, *414*(6865), 799–806.

Sathirapongsasuti, J. F. (2013). Copdsexualdimorphism.data: Data to support sexually dimorphic and copd differential analysis for gene expression and methylation. [Computer software manual]. (R package version 1.4.0)

Sato, J. R., Hoexter, M. Q., de Magalhães Oliveira, P. P., Brammer, M. J., Murphy, D., Ecker, C., ... others (2013). Inter-regional cortical thickness correlations are associated with autistic symptoms: a machine-learning approach. *Journal of psychiatric research*, *47*(4), 453–459.

Schadt, E. E. (2009). Molecular networks as sensors and drivers of common human diseases. *Nature*, *461*(7261), 218–223.

Schelldorfer, J., Bühlmann, P., DE, G., & VAN, S. (2011). Estimation for high-dimensional linear mixed-effects models using l1-penalization. *Scandinavian Journal of Statistics*, *38*(2), 197–214.

Shah, R. D. (2016). Modelling interactions in high-dimensional data with backtracking. *Journal of Machine Learning Research*, *17*(207), 1–31.

Shaw, P., Greenstein, D., Lerch, J., Clasen, L., Lenroot, R., Gogtay, N. e. a., ... Giedd, J. (2006). Intellectual ability and cortical development in children and adolescents. *Nature*, *440*(7084), 676–679.

She, Y., & Jiang, H. (2014). Group regularized estimation under structural hierarchy. *arXiv preprint arXiv:1411.4691*.

Song, M., Hao, W., & Storey, J. D. (2015). Testing for genetic associations in arbitrarily structured populations. *Nature genetics*, *47*(5), 550–554.

Spain, S. L., & Barrett, J. C. (2015). Strategies for fine-mapping complex traits. *Human molecular genetics*, *24*(R1), R111–R119.

Spearman, C. (1904). The proof and measurement of association between two things. *The American journal of psychology*, *15*(1), 72–101.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Tibshirani, R., & Friedman, J. (2017). A pliable lasso. *arXiv preprint arXiv:1712.00484*.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(1), 91–108.

Timpson, N. J., Greenwood, C. M., Soranzo, N., Lawson, D. J., & Richards, J. B. (2018). Genetic architecture: the shape of the genetic contribution to human traits and disease. *Nature Reviews Genetics*, *19*(2), 110.

Toloşi, L., & Lengauer, T. (2011). Classification with correlated features: unreliability of feature ranking and solutions. *Bioinformatics*, *27*(14), 1986–1994.

Tothill, R. W., Tinker, A. V., George, J., Brown, R., Fox, S. B., Lade, S., ... others (2008). Novel molecular subtypes of serous and endometrioid ovarian cancer linked to clinical outcome. *Clinical Cancer Research*, *14*(16), 5198–5208.

Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, *109*(3), 475–494.

Tseng, P., et al. (1988). Coordinate ascent for maximizing nondifferentiable concave functions.

Tseng, P., & Yun, S. (2009). A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, *117*(1), 387–423.

Tzourio-Mazoyer, N., Landeau, B., Papathanassiou, D., Crivello, F., Etard, O., Delcroix, N., ... Joliot, M. (2002). Automated anatomical labeling of activations in spm using a macroscopic anatomical parcellation of the mni mri single-subject brain. *Neuroimage*, *15*(1), 273–289.

Wakefield, J. (2013). *Bayesian and frequentist regression methods*. Springer Science & Business Media.

Wan, Y.-W., Allen, G. I., Anderson, M. L., & Liu, Z. (2015). Tcga2stat: Simple tcga data access for integrated statistical analysis in r [Computer software manual]. Retrieved from https://CRAN.R-project.org/package=TCGA2STAT (R package version 1.2)

Wang, D., Eskridge, K. M., & Crossa, J. (2011). Identifying qtls and epistasis in structured plant populations using adaptive mixed lasso. *Journal of agricultural, biological, and environmental statistics*, *16*(2), 170–184.

Wendland, E. M., Torloni, M. R., Falavigna, M., Trujillo, J., Dode, M. A., Campos, M. A., ... Schmidt, M. I. (2012). Gestational diabetes and pregnancy outcomes-a systematic review of the world health organization (who) and the international association of diabetes in pregnancy study groups (iadpsg) diagnostic criteria. *BMC pregnancy and childbirth*, *12*(1), 1.

Witten, D. M., Shojaie, A., & Zhang, F. (2014). The cluster elastic net for high-dimensional regression with unknown variable grouping. *Technometrics*, *56*(1), 112–122.

Witten, D. M., Tibshirani, R., & Hastie, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, kxp008.

Xie, Y. (2015). *Dynamic documents with r and knitr* (Vol. 29). CRC Press.

Yang, J., Benyamin, B., McEvoy, B. P., Gordon, S., Henders, A. K., Nyholt, D. R., ...

others (2010). Common snps explain a large proportion of the heritability for human height. *Nature genetics*, *42*(7), 565.

Yang, J., Zaitlen, N. A., Goddard, M. E., Visscher, P. M., & Price, A. L. (2014). Advantages and pitfalls in the application of mixed-model association methods. *Nature genetics*, *46*(2), 100.

Yang, Y., & Zou, H. (2014). gglasso: Group lasso penalized learning using a unified bmd algorithm. Retrieved from http://CRAN.R-project.org/package=gglasso (R package version 1.3)

Yang, Y., & Zou, H. (2015). A fast unified algorithm for solving group-lasso penalize learning problems. *Statistics and Computing*, *25*(6), 1129–1141.

Yu, J., Pressoir, G., Briggs, W. H., Bi, I. V., Yamasaki, M., Doebley, J. F., ... others (2006). A unified mixed-model method for association mapping that accounts for multiple levels of relatedness. *Nature genetics*, *38*(2), 203.

Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(1), 49–67.

Zhang, B., & Horvath, S. (2005). A general framework for weighted gene co-expression network analysis. *Statistical applications in genetics and molecular biology*, *4*(1).

Zhang, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, 894–942.

Zhao, P., Rocha, G., & Yu, B. (2009). The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 3468–3497.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, *101*(476), 1418–1429.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320.

Zou, H., Hastie, T., Tibshirani, R., et al. (2007). On the "degrees of freedom" of the lasso. *The Annals of Statistics*, *35*(5), 2173–2192.