New Penalized Regression Approaches to Analysis of Genetic and Genomic Data

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Sunkyung Kim

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Advised by Wei Pan, Ph.D

July, 2012

Acknowledgements

There are many people that have earned my gratitude for their contribution to my time in graduate school. First, I express my sincere gratitude to my thesis advisor, Dr. Wei Pan, who is a great researcher, teacher, and also advisor, for providing his excellent guidance throughout this research. I've been honored to work with him as one of his students, and to receive his care in students' research progress in his busy schedule.

Thanks to Dr. Baolin Wu, Dr. Xiaotong Shen, and Dr. Weihua Guan who are my committee members, for their constructive comments and time for a review of my thesis. I also appreciate Dr. Melanie M. Wall whose encouragement made me study happily in my second year right after I earned my son.

I am indebted to many of my friends in our department to support me. I will miss them and our chat in A450 a lot. I also thank University of Minnesota and our department for their help and service, which allowed me to devote to my study.

Finally, my heartful thanks go to my family members who are a consistent source of my joy in my lifetime. I can not imagine finishing my study without the support from my parents and parents-in-law. I also owe my thanks to my husband Eugene and lovely son Brian who are my best friends.

Abstract

Statistical analysis of high dimensional genetic data is challenging. Penalized regression represents a class of attractive approaches for its simultaneous and efficient variable selection and parameter estimation. While penalized regression has been shown to be advantageous in variable selection and outcome prediction over many other approaches, in this dissertation, we study several new penalized regression methods based on a new non-convex penalty, Truncated L_1-penalty (TLP), which is a surrogate of the (ideal but computationally infeasible) L_0-penalty.

First, we use the TLP for hypothesis testing to test for genetic association of multiple rare variants (RVs) (e.g. in a gene) with a quantitative trait. To deal with the low minor allele frequencies (MAFs) of RVs, we attempt a TLP-based grouping strategy to combine information across multiple RVs. Since there has been a paucity of literature [1] on penalized regression for hypothesis testing problem, we compare the performance of our proposed penalized regression method with other existing global tests.

Next, we apply the TLP-based grouping strategy to joint modeling of a large number of RVs across a genome. In particular, we apply the proposed method and several other existing penalized methods to the Genetic Analysis Workshop 17 (GAW17) exome-sequencing data.

Finally, we propose a novel TLP-based network penalty to smooth the regression coefficients over a given network that describes a priori relationships among the predictors (e.g. genes). In contrast to existing approaches that assume that the regression coefficients of neighboring nodes/predictors in a network are close in magnitude, we impose a much weaker prior assumption that the regression coefficients of neighboring nodes in a network are likely to be zero (or non-zero) at the same time, regardless of their specific magnitudes. Since the proposed penalty is not convex, we develop a computational algorithm based on difference convex programming.

For each of the above three problems, we conduct both simulation studies and real data applications to demonstrate the competitive performance and practical utility of the proposed methods.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

With the increasing availability of genetic and genomic data, analyzing a large number of variables has been a challenge in genome-wide association studies (GWAS). Since not all predictors are expected to be associated with a trait, selecting only informative variables is desirable to obtain a parsimonious model for precise prediction and parameter estimation. Penalized regression has received much attention as one attractive remedy to solve this "large p" problem since it is capable of simultaneous variable selection and parameter estimation by adding a penalty term in the objective function. Among existing penalties, e.g. L_0, L_1, L_2-penalties, ridge regression [2] using a squared L_2-norm predicts better than the ordinary least squares (OLS), but does not produce a parsimonious model since it tends to keep all the predictors. L_0-penalty is attractive because it directly penalizes the number of nonzero coefficients; however, when the number of variables is large, it is not computationally feasible. Thus, computationally efficient surrogates, L_1-type penalties, have been developed including Lasso [3], LARS [4], fused lasso [5], fused adaptive lasso [6], grouped lasso [7], relaxed lasso [8], and elastic net [9]. The main strength of an L_1-penalty such as Lasso is that it produces a sparse coefficient vector, which means that many of the coefficients are estimated as 0. The resulting non-zero estimates in the sparse vector can be very helpful, but the selection of highly correlated predictors is problematic. In other words, only a part of informative predictors are included in the model while others are not, leading to inaccurate variable selection and outcome prediction. To solve the problem of L_1-penalty and computational infeasibility of L_0-penalty, [10] suggested a novel regularizer, Truncated L_1 penalty (TLP). TLP is an approximation to the L_0-penalty with the degree of approximation dependent on the data.

Rather than just selecting variables, grouping has been recommended because, for example, genes sharing a biological pathway are expected to work together to influence a disease trait if the gene pathway is indeed functional. Therefore, embedding this *prior* knowledge of the gene pathway into the model is statistically and biologically desired to enhance a model's interpretation and prediction. This type of information is available from the Biomolecular Interaction Network Database(BIND) [11], the Human Protein Reference Database (HPRD) [12], biological pathways in the Kyoto Encyclopedia of Genes and Genomes (KEGG) [13], and gene functional

annotations in the Gene Ontology [14]. Currently, there exist group penalties (pre-mentioned fused lasso, grouped lasso, and elastic net are such examples) in penalized regression. For instance, the fused lasso [5] encourages sparsity in differences between coefficients, and [15] proposes a non-convex form of group penalty. While these two methods considered subgroups with the same coefficient values, the same absolute coefficient values were considered in OSCAR [16], weighted L_γ [17], and an L_1-group regularizer using the sign of pairwise correlation of predictors [18]. As another approach to resolving simultaneous grouping and selection of predictors in the estimation process over an undirected graph, we attempt a Truncated L_1 group penalty, where both selection and group penalties are included in the objective function.

TLP enjoys the strengths of L_0-penalty (sparsity & unbiasedness) while overcoming the weaknesses of L_1-penalty (biased estimation & inaccurate variable selection). Therefore, the scope of its application to other statistical genetic problems is quite open and worthwhile to investigate.

Chapter 2 deals with the global hypothesis testing problem by applying TLP for rare variant (RV) analysis. Variants with minor allele frequency (MAF) less than 1% are considered. From gene-centric data in Chapter 2, Chapter 3 extends the application of the method to genome-wide analysis, where grouping of variants is allowed within a gene. Chapter 4 introduces a new network-based penalty built on a TLP approximation. Lastly, Chapter 5 discusses the limitations of the proposed approaches and suggests further work to improve. A brief summary of each chapter follows.

1.1 A new penalized regression approach to testing for quantitative trait-rare variant association

Although penalized regression methods have shown many advantages in variable selection and outcome prediction over other approaches, there is a paucity of literature on its application to hypothesis testing, in particular, in genetic association analysis. In this study, we apply a new penalized regression method with a Truncated L_1-penalty (TLP) [10] for either variable selection, or both grouping in a data-adaptive

way to test for association between a quantitative trait and a group of rare variants (e.g. in a gene). TLP-based tests' performances are compared with some existing tests via simulations and application to real sequence data from the Genetic Analysis Workshop 17 (GAW17). They are competitive with other methods in certain cases gaining a limited but identifiable amount in power. The possible problems utilizing penalized regression methods in genetic hypothesis testing are also discussed.

1.2 New penalized regression methods for genome-wide selection of RVs associated with a quantitative trait.

Next-generation sequencing technologies have allowed us to explore disease causal rare variants. However, detection of rare variants needs more advanced statistical tools because of their low-frequency. In this chapter, using Genetic Analysis Workshop 17 (GAW17) data consisting of 697 subjects in a high dimensional setting, we compare several penalized regression methods with grouping pursuit within a gene to detect the informative genetic markers on a trait. The methods include Lasso, graph fused lasso (gflasso), graph OSCAR (goscar), and two different versions of a recently developed Truncated L_1-penalty (TLP) [10] on grouping (ncTLF and ncTFGS). To fit these models we use the Feature Grouping and Selection Over an Undirected Graph (FGSG) package created by [19]. FGSG is a C library with interface to MATLAB. Note that Chapter 2 treats single gene-based approach, and Chapter 3 treats genome-wide, or multiple genes-based approach.

1.3 A network-based penalized regression method with application to genomic data

New penalized regression methods have been introduced to utilize network structures of predictors, e.g. gene networks, to improve parameter estimation and variable selection. All the existing network-based penalized methods are based on an assumption that parameters, e.g. regression coefficients, of neighboring nodes in a

network are close in magnitude; however, it may not hold. In this paper we propose a novel penalized regression method based on a weaker prior assumption that the parameters of neighboring nodes in a network are likely to be zero or non-zero at the same time, regardless of their specific magnitudes. We propose a novel non-convex penalty function to incorporate this prior assumption and an algorithm based on difference convex programming. We use simulated data and a gene expression dataset to demonstrate the advantages of the proposed method over some existing methods.

Chapter 2

A new penalized regression approach to testing for quantitative trait-rare variant association

2.1 Introduction

Genome-wide association studies (GWAS) have uncorvered many common variants (CVs) associated with complex diseases, but the proportion of variance explained by the identified CVs is still low [20]. Alternatively, with the development of sequencing technologies, analysis of rare variants (RVs) have become feasible. Recent studies have demonstrated that some RVs are associated with complex disease. For example, Kotowski *et al.* (2006) found that mutiple RVs in gene PCSK9 are associated with plasma levels of low-density lipoprotein cholesterol.

In this study, we propose new penalized regression methods to 1) test for association between a quantitative trait and multiple RVs, and 2) investigate how a penalized method works in hypothesis testing in a low dimensional setting. One basic statistical test is the F-test in linear regression. For example, in simple regression, the trait, Y, is regressed on each of multiple variants sequentially. However, because of the extremely low minor allele frequency (MAF) of a RV, a test to detect the association between a trait and single RV might be low powered. Also, this approach seems not ideal since the test might be very conservative due to a stringent control for multiple testing, e.g. by the Bonferroni correction to control the probability of false positives. In addition, ultimately, complex diseases are expected to be affected by a combination of genetic variants. Thus an analysis in which a group of variants are tested simultaneously for their joint effects on the trait is preferred. In multiple regression, to assess any association between a trait and k RVs, all k RVs are added to a regression model.

However, when k is large, the statistical power might decrease due to the cost of large degrees of freedom (DF), k. To avoid the large DF issue and also to aggregate information from multiple RVs, one common strategy is to pool or collapse across the multiple RVs (Liu and Leal 2010; Madsen and Browning 2009). One such attempt is the Sum test [21], which was developed to utilize joint effects of multiple variants while reducing the DF at the same time. With only 1 DF, the Sum test enhances power under some scenarios [22] [23]. However, it is noted that the performance of the Sum test depends on the directions of the variants' associations with a trait. Thus, in an extreme case where a half of the variants are positively associated with the trait and the other half are negatively associated with similar effect sizes,

the positive and negative effects may cancel out, and the Sum test would perform poorly. In addition, in the Sum test, combining all predictors into just one group while ignoring variants' varying effect sizes would work well only when they are almost equally associated with the trait in the same direction, i.e when regression coefficients β_js are all close to each other. Also, the Sum test works poorly if many neutral or null RVs are present [24]. Consequently, in a non-ideal situation, the Sum test might have low power.

Recently, to deal with a large number of variables in genetic and genomic studies, penalized regression has received much attention, e.g, Lasso [3], LARS [4], fused lasso [5], fused adaptive lasso [6], grouped lasso [7], relaxed lasso [8], and elastic net [9]. It has been considered as one attractive remedy since it is capable of simultaneous model selection and parameter estimation by imposing a desired prior on the parameters. One of very recently developed penalties in a high dimensional setting is Truncated L_1-penalty (TLP) of [10], where the dimension or predictor number p exceeds sample size n ($p > n$). The TLP approximates the L_0-penalty while attempting to reduce the estimation bias of the L_1-penalty. To investigate whether an application of TLP would boost statistical power in hypothesis testing for genetic association in cases of $p < n$, in this study, we apply TLP with only variable selection, denoted TLPS, with both variable selection and parameter grouping, denoted TLPSG, to data-adaptively select and group predictors to reduce the DF as in the Sum test, while reducing the bias of the penalized estimates based on an L_1-type penalty. As the competitors, we compare TLPS and TLPSG to Lasso and graph fused lasso [18] ($gflasso$). TLPS and TLPSG shrinks a pair of predictors' regression coefficients towards each other to realize $|\beta_j - r(j, j')\beta_{j'}|$, where $r(j, j')$ is the sign of correlation between two feature vectors $X_{\cdot j}$ and $X_{\cdot j'}$. Note the main difference between TLPSG and $gflasso$ is that the former aims to smoothe two (absolute) regression coefficients ($|\beta_j|, |\beta_{j'}|$) only when their difference is less than a pre-fixed threshold τ.

This chapter is organized in four sections. Section 2 provides a brief review of some existing association tests to be compared, and then introduces TLP-based tests. In section 3, the comparison results from simulation studies and application to the Genetic Analysis Workshop 17 (GAW17) [25] data are presented. Finally, the discussion section summarizes the results, and suggests some potential problems for

future study.

2.2 Methods

2.2.1 Some existing OLS based tests

We briefly review some existing global tests which are based on the ordinary least square (OLS) estimates. Given n independent observations (Y_i, X_i), $i = 1, ..., n$, with Y_i as a quantitative trait and a vector $X_i = (X_{i1}, ..., X_{ik})$ as genotypes of k variants for subject i, we would like to test for any possible association between the trait and genotypes. We use the dosage coding for X_{ij}: X_{ij}=0, 1 or 2, representing the count number of one of the two alleles present in variant j of subject i. A multi-locus association analysis is based on fitting a linear model,

$$Y_i = \beta_0 + \sum_{j=1}^{k} X_{ij}\beta_j + \epsilon_i \tag{2.1}$$

A global test of any possible association between the trait and k variants can be formulated as testing on the multiple parameters β_js for $j = 1, ..., k$ with null hypothesis $H_0 : \beta = (\beta_1, ..., \beta_k)' = 0$ by an F-test based on the OLS estimates of minimizing residual sum of squares. A potential problem with the test is the power loss due to the large variance of $\hat{\beta}_j$ since MAFs of RVs are small.

We also apply four different tests: the Score, the sum of squared score (SSU), its weighted version SSUw [21], and the univariate minP (UminP) tests as Score is popular in general statistics; UminP is most popular in GWAS for CVs.; [24] showed that SSU and SSUw test were powerful in RVs association testing in case-control study design. The derived score vector U and its covariance matrix V from (2.1) under the null are

$$U = \sum_{i=1}^{n} (Y_i - \bar{Y}) X_i,$$

$$V = Cov(U) = \hat{\sigma}_o^2 \sum_{i=1}^{n} (X_i - \bar{X})(X_i - \bar{X})^T,$$

where $\bar{Y} = \sum_{i=1}^{n} Y_i/n$, $\bar{X} = \sum_{i=1}^{n} X_i/n$, and $\hat{\sigma}_o^2$ $(=\sum_{i=1}^{n}(Y_i - \bar{Y})^2/(n-1))$ is the sample variance estimate from the null model. The corresponding four test statistics are

$$
\begin{aligned}
&T_{Score} = U^T V^{-1} U, \\
&T_{SSU} = U^T U, \\
&T_{SSUw} = U^T V_d^{-1} U \quad \text{with} \quad V_d = Diag(V), \\
&T_{UminP} = \max_{j=1}^{k} U_j^2 / v_j,
\end{aligned}
$$

where U_j is the jth element of U and v_j is the (j,j)th diagonal element of V. Under H_0, asymptotically T_{Score} has a χ_k^2 distribution. T_{SSU} and T_{SSUw} have approximate chi-squared distributions [21], and the p-value of T_{UminP} can be numerically obtained [26].

Next, we apply the Sum [21] test to the case with a quantitative trait, and its modified version, a data-adaptive Sum (aSum) test [27]. The Sum test was originated to model multiple parameters jointly and also reduce the resulting large DF: while using all the variants within the model, it assumes that the variants are associated with the trait with a common association effect, β_c, as following:

$$
Y_i = \beta_{c,0} + \sum_{j=1}^{k} X_{ij} \beta_c + \epsilon_i \tag{2.2}
$$

Fitting (2.2) is equivalent to conducting a simple regression of Y on a new covariate, the sum of the genotypes over the multiple variants. To address the question of whether any association between the disease and the variants exists, one simply needs to test a null $H_0 : \beta_c = 0$, without multiple test adjustment. The main advantage of the Sum test is that, because it tests on only one parameter β_c, there will be no power loss due to the large DF. The common association parameter β_c is a weighted average of the individual $\beta_{M,1}, ..., \beta_{M,k}$ [21] in marginal model $Y_i = \beta_{M,0} + X_{ij}\beta_{M,j} + \epsilon_{ij}$ for $j = 1, ..., k$. On the other hand, the main problem of the Sum test is its dependence on the signs of $\beta_{M,j}$s or on the codings of each variant (i.e. which allele is chosen as the reference category). If the signs are not the same, the test may have a quite small $\hat{\beta}_c$ and thus low power. To overcome the limitation of the Sum test, [27] proposed the aSum test for a case-control study

design: flip the coding of variant j, $X^*_{\cdot j} = 2 - X_{\cdot j}$ if $\hat{\beta}_{M,j} < 0$ and its p-value $p_{M,j} \leq \alpha_0$ in the marginal model. Then fit the model, (2.2). To test H_0 in the aSum test, we use a permutation-based log-likelihood ratio test (LRT), which is asymptotically equivalent to the score test. For the choice of α_0, we use the same value as recommended in the paper [27], 0.1, to prevent low power by the heavy tailed null distribution when α_0 is 1.

While the F-test is based on OLSE, in next section, we apply some penalized regression methods, Lasso, $gflasso$, and a recently developed L_0-approximation TLP method with only variable selection (TLPS) and also with grouping (TLPSG). Specially, the $gflasso$ and TLPSG methods are applied to check whether the predictors' grouping accompanying variable selection contribute to improving the testing power by resolving the cost of large DF in joint model (2.1).

2.2.2 Penalized regression based tests

Parameter estimation from penalized regression

Given response $Y = (Y_1, ..., Y_n)'$, and a matrix of k predictors $X = \{X_{\cdot 1}, ..., X_{\cdot k}\}$, the Lasso estimate of β is based on the penalized least squares function:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda \sum_{j=1}^{k} |\beta_j|, \tag{2.3}$$

where a large λ automatically yields some components of $\hat{\beta}$ as 0. While Lasso does effective variable selection, to reduce Lasso's estimation bias, [10] suggested a truncated Lasso(L_1)-penalty (TLP) $J_\tau(|x|) = min(\frac{|x|}{\tau}, 1)$ which, as $\tau \to 0^+$, approximates the L_0-norm, $I(|x| \neq 0)$, and only penalizes $|\beta_j|$ that is less than a threshold τ. The amount of approximation in TLP is controlled by a tuning parameter, τ. See Figure 2.1 for a display over the different values of τ. Then, the $\hat{\beta}$ with TLP is obtained from

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{k} J_\tau(|\beta_j|), \tag{2.4}$$

and we denote (2.4) as TLPS. While Lasso and TLPS both consider only variable selection, an alternative way to reduce model complexity is grouping. To investigate the predictors' grouping effect on testing power, we apply two recent penalized grouping methods, $gflasso$ and TLPSG. The β estimate from $gflasso$ is based on the following function:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{k} |\beta_j| + \lambda_2 \sum_{j<j'} |\beta_j - r(j,j')\beta_{j'}|, \tag{2.5}$$

where $r(j,j')$ is the sign of the correlation between two predictors $X_{\cdot j}$ and $X_{\cdot j'}$ to target $|\beta_j| = |\beta_{j'}|$, denoted $gflasso_{r=cor}$. The first penalty is to select variables, and the second one is to encourage grouping of two coefficients with the consideration of their association directions by multiplying $r(j,j')$ in one component of a pair. We also set $r(j,j')$ simply as 1, denoted $gflasso_{r=1}$, to simply target $\beta_j = \beta_{j'}$. As next, the β with TLP grouping, TLPSG, comes from:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{k} J_\tau(|\beta_j|) + \lambda_2 \sum_{j'<j} J_\tau(\big||\beta_j| - |\beta_{j'}|\big|), \tag{2.6}$$

The second penalty shrinks the difference of $|\beta_j|$s if the difference is within the upper bound τ and the number of groups is a decreasing function of λ_2. Thus, some properly selected tuning parameters $(\lambda_1, \lambda_2, \tau)$ are expected to provide a balance between the model complexity and model goodness of fit, possibly contributing to enhanced power. Note that in the Sum test all predictors had to belong to one group even when the variants' associations with the trait are quite different both in effect sizes and directions, but the TLPSG method attempts to conduct a more precise grouping over all variants with a data-adaptive grouping.

To compute β in Lasso, $gflasso$, TLPS and TLPSG, we used the Feature Grouping and Selection Over an Undirected Graph (FGSG) package in [19], which is a C library with interface to MATLAB and is quite fast to run. Its computing efficiency allowed us to estimate separate tuning parameters for each permuted dataset to control the type I error as explained in the next section.

Testing the null hypothesis, $H_0 : \beta = 0$

To test the null $H_0 : \beta = 0$ in (2.1), we conduct a permutation-based test, in which the p-value is calculated by comparing a test statistic (T) applied to the original dataset to the ones ($T_0^{(b)}$) applied to the b^{th} permuted dataset, b=1,...,B. A permutation step is added accordingly to control the Type I error. The testing procedure follows:

Step 1. With the original data $\{(Y_i, X_i)\}$, we solve a penalized regression problem to obtain $\hat{\beta}$.

Step 2. Calculate a corresponding test statistic $T(\hat{\beta})$.

Step 3. By repeatedly permuting the observed Y of the original data, we obtain B sets of permuted data $\{(Y_i^{(b)}, X_i)\}$ for $b = 1, ..., B$. For each permuted data set, $\{(Y_i^{(b)}, X_i)\}$, we repeat the Step 1 and 2, obtaining the test statistic $T_0^{(b)}(\hat{\beta})$ for $H_0 : \beta = 0$. A permutation-based p-value is calculated as $p = \sum_{d=1}^{D} \sum_{b=1}^{B} I(T < T_{0d}^{(b)})/(DB)$.

In Step 2, in choosing a test statistic T, first, commonly over all methods, we calculate a new 1-df F-statistic (1-df) by fitting

$$Y_i = \alpha_0 + (X_i\hat{\beta})\alpha + \epsilon_i,$$

where $\hat{\beta}$ is from Step 1. We test $H_0 : \beta = 0$ via testing $H_0' : \alpha = 0$. This 1-df test uses model selection information from the corresponding penalized method while allowing testing with 1 df only. For TLPSG, we also apply the corresponding SSU and SSUw tests, where the test statistics T_{SSU} and T_{SSUw} are all based on the selected variables from its corresponding estimates. Specifically,

$$T_{SSU} = U^{*'}U^*,$$
$$T_{SSUw} = U^{*'}(V_d^*)^{-1}U^* \quad \text{with} \quad V_d^* = Diag(V^*),$$

where U^* is a sub-component vector of the score vector U corresponding to $|\hat{\beta}_j| \neq 0$, and $|\hat{\beta}_j| > 0.001$ is considered as non-zero. Similarly, V^* is the corresponding sub-matrix of the covariance matrix V.

Selection of tuning parameters

To estimate the tuning parameters, we apply a grid-search with Akaike's information criterion (AIC) [28]; AIC=$-2L + 2p$, where L=$\left(\frac{-n}{2}log(\hat{\sigma}^2) - \frac{n-p-1}{2}\right)$ and $\hat{\sigma}^2$ (=$\sum_{i=1}^{n}(Y_i - \beta_0 - X_i\hat{\beta})^2/(n-p-1)$) is the sample variance estimate under the corresponding model. The p of AIC is computed as the number of non-zero $|\hat{\beta}_j|$ for Lasso and TLPS, the number of non-zero unique $\hat{\beta}_j$ for $gflasso_{r=1}$, and the number of non-zero unique $|\hat{\beta}_j|$ for $gflasso_{r=cor}$ and TLPSG. For λ of Lasso, the one resulting in the smallest AIC out of 50 equally spaced points in [0.001,10] is selected. Similarly the values of $(\lambda_1, \lambda_2, \tau)$ in other methods are in grid search of 5 x 5 x 5 equally spaced grid points of [0.001,1] x [0.001,0.5] x [0.001,0.5]. For each permuted data $(Y_i^{(b)}, X_i)$ for $b = 1, ..., B$, we also estimate its own $(\lambda_1^{(b)}, \lambda_2^{(b)}, \tau^{(b)})$ to properly control the type I error.

2.3 Results

2.3.1 Simulation study

We consider two simulation schemes. In the first simulation set-up, we generate only RVs with a total of 200 replicates and n=400 in each replicate. The permutation size is set as B=100. For each replicate, to generate k variants including 6 causal ones in linkage disequilibrium (LD), two latent vectors from multivariate normal distribution MVN(0,R) are simulated, where R has a first order auto regressive (AR1) structure; the association between any two elements of the latent vector decreases by ρ=0.8 times as 1 lag increases. Then, the vector is dichotomized to yield a haplotype with the minor allele frequency (MAF) randomly chosen between 0.005 and 0.01. The genotype data $X_i = (X_{i1}, ..., X_{ik})'$ for sample i is obtained by adding two haplotypes together. Finally, Y_i is generated from the randomly located 6 causal variants with σ^2=2 in model (2.1), where the intercept β_0 is set as 0.3

throughout the simulations. The considered 3 cases are:

$$\text{Case 1: } \beta = (\underbrace{0.9, 0.9, 0.9, 0.9, 0.9, 0.9}_{6}, \underbrace{0, ..., 0}_{k-6})'$$

$$\text{Case 2: } \beta = (\underbrace{1.2, 1.2, 1.2, -1.2, -1.2, -1.2}_{6}, \underbrace{0, ..., 0}_{k-6})'$$

$$\text{Case 3: } \beta = (\underbrace{1.4, 1.3, -1.2, 1.2, -1.3, 1.4}_{6}, \underbrace{0, ..., 0}_{k-6})'.$$

In each case, we vary the number of non-causal RVs k-6 from 0 to 24 so that the total number of RVs, k, ranges from 6 to 30. The Type I error is computed from the Y under $H_0 : \beta = (0, ..., 0)'$.

In the second simulation set-up, multiple RVs and two CVs are generated as predictors to mimic the GAW17 data we use later. The frequency of one allele for CVs is randomly distributed between 0.2 and 0.7, and CVs may or may not be chosen as causal variant in each replicate. When a CV is randomly selected as causal variant, the effect size of it β_j is scaled down to $\beta_j/10$ in the following cases to prevent its dominating association with the outcome. The considered 3 cases for RVs+CVs case are:

$$\text{Case 1: } \beta = (\underbrace{1, 1, 1, 1, 1, 1}_{6}, \underbrace{0, ..., 0}_{k-6})'$$

$$\text{Case 2: } \beta = (\underbrace{1.5, 1.5, 1.5, -1.5, -1.5, -1.5}_{6}, \underbrace{0, ..., 0}_{k-6})'$$

$$\text{Case 3: } \beta = (\underbrace{1.1, 1.3, -1.2, 1.2, -1.3, 1.1}_{6}, \underbrace{0, ..., 0}_{k-6})',$$

Figure 2.2 displays the TLPS and TLPSG solution paths of $|\hat{\beta}_j|$ over a tuning parameter given other(s), where two true horizontal solution lines are at 1.2 and 0 for Case 2 of the RVs only set-up. Inferred from the closed form solution provided in [10] but for orthogonal predictor sets in X, for TLPS, when λ_1 increases in TLPS given τ in (a), $\hat{\beta}_j$=0 for $|\hat{\beta}_j^{ols}| \leq \frac{\lambda_1}{\tau}$. When τ increases given λ_1 in (b), $\hat{\beta}_j$=$(|\hat{\beta}_j^{ols}| - \frac{\lambda_1}{\tau})sign(\hat{\beta}_j^{ols})$, which is as same as $|\hat{\beta}_j|$=$|\hat{\beta}_j^{ols}|$+constant. On the other hand, in TLPSG plot (d), when λ_2 increases given λ_1 and τ, $|\hat{\beta}_j| - |\hat{\beta}_{j'}|$=0

for $\left||\hat{\beta}_j^{ols}| - |\hat{\beta}_{j'}^{ols}|\right| \leq \frac{\lambda_2}{\tau}$, i.e. grouping effect. In (e), when τ increases given λ_1 and λ_2, $|\hat{\beta}_j| - |\hat{\beta}_{j'}|= (\left||\hat{\beta}_j^{ols}| - |\hat{\beta}_{j'}^{ols}|\right| - \frac{\lambda_2}{\tau})sign(|\hat{\beta}_j^{ols}| - |\hat{\beta}_{j'}^{ols}|)$, which is as same as $\left||\hat{\beta}_j| - |\hat{\beta}_{j'}|\right| = \left||\hat{\beta}_j^{ols}| - |\hat{\beta}_{j'}^{ols}|\right|$+constant, i.e again no grouping.

Table 2.1 presents the simulation results of the RVs only set-up. Type I error seems to be properly controlled under the null for all cases, though there are some slightly inflated numbers such as 0.055 at maximum, possibly due to the relatively small number of replicates. Under the non-null, in Case 1 when the causal RVs' associations are all in the same direction, the Sum or aSum test beat other methods. Within penalized regression based methods, TLPSG with SSU and SSUw test statistics are more powerful, and TLPSG with SSUw beat the F-test regardless of the number of non-causal RVs included. There seems to be little gain with grouping in TLPSG compared to no grouping in TLPS, and the 1df-test of TLPSG works better than $gflasso_{r=cor}$ until the number of non-causal RVs is moderate at 16. Overall, penalized regression based methods does not improve power over Sum and aSum. In Case 2 and 3 where the causal RVs' effect directions are mixed, the Sum test works poorly as expected, while the aSum test increases the power. The TLPSG does not improve power over SSU and SSUw tests. Again, the comparison of TLPS and TLPSG reveals that the predictors' grouping might or might not contribute to boost power. The results of the RVs+CVs set-up are in Table 2.2. The similar pattern is observed as in the previous set-up. The SSUw test is overall winner.

Table 2.3 and 2.4 show the model error (ME) calculated as $(\beta - \hat{\beta})'X'X(\beta - \hat{\beta})$, and its standard deviation (sd). Except in the lowest dimension of k=6 (when the number of non-causal RVs is 0), the ME of TLPS or TLPSG is less than of OLS estimates, but they are larger than those from Lasso or $gflasso$.

In Table 2.5 and 2.6, we also investigate the model selection performance of the methods by calculating the mean number of true positives (TP) and false positives (FP), where a $|\hat{\beta}_j| > 0.001$ is considered as positive. The OLS estimates show no variable selection with the mean TP and mean FP are all close to possible maximum values. In the null cases, $gflasso_{r=cor}$ yields most sparse model, but its conservativeness is quite same in all non-null cases too. The Lasso's FP rate is the second lowest overall, but the pattern is similar in detecting TP. While TLPSG removes more noise variables compared to TLPS, it also seems to detect less true

positive ones.

The methods' parameter estimation performances are summarized in Table 2.7 and 2.8. As expected, OLS estimates have least biased estimates with the largest mean squared error (MSE) due to the larger variance. The TLPS and TLPSG's MSEs are between OLS and Lasso/$gflasso$. For causal variants (β_c), Lasso and $gflasso$ shrink them more toward 0, and TLP based methods' estimation bias are much less than theirs as expected.

2.3.2 Real sequence data: GAW17

For real data analysis, we apply the methods to the real sequence data from GAW17 [25]. The data set consists of 3,205 autosomal genes with 24,487 variants on 697 subjects. The genotypes were obtained from the sequence alignment files provided by the 1000 Genomes Project for the pilot 3 study. The GAW17 data also included two hundred replicates of three quantitative traits named Q1, Q2, and Q4, where only Q1 and Q2 were influenced by genetic factors. We use Q2 here which is influenced by 72 variants in 13 genes. The true effect sizes of all variants range from 0.2 to 1.2, so all variants are positively associated in quite differential magnitudes with the trait.

In this study, we test over all causal genes (PLAT, SREBF1, SIRT1, VLDLR, VNN3, PDGFD, BCHE, INSIG1,LPL,RARB,VNN1,VWF) except GCKR which has just one SNP. The number of causal variants (nC) in each gene affecting Q2, and some summary statistics of MAF and pairwise correlations (COR) are listed in Table 2.9. Within each gene, most variants were RVs, but a few were common variants (CVs) with their MAFs larger than 5%. First, we test any association between Q2 and all variants gene by gene in Table 2.10 and 2.11, and then test without the CVs in Table 2.12 and 2.13. To compute Type I error, outcome Y generated from the intercept only model with β_0=0.3 and σ^2=1 is used.

Type I error seems to be controlled overall as observed in Table 2.10 and 2.12, though there are some exceptions as the value of 0.080. In Table 2.11, when including all variants within a gene, the identity of the winner differs across the genes: The F-test is the winner for VLDLR, VNN3, PDGFD and LPL. For gene VLDLR, BCHE, VNN1 and VWF, SSU or SSUw test are the best. The two $gflasso$ tests work quite

similarly over all genes. The TLP based methods perform well in SREBF1, VNN1 and INSIG1. After removing a few CVs in Table 2.13, the SSU test recovers good power in PDGFD, BCHE and LPL. The Sum test is winner in BCHE, and the F-test from OLS estimates perform best in VNN3 and PDGFD. For VNN3, the TLPSG with SSU is winner. Finally, Table 2.14 shows the model selection performance in GAW17 data. Overall, in GAW17 data, the Lasso or $gflasso$ seems to produce sparse coefficient vector compared to TLPS and TLPSG.

In conclusion, the TLP methods' application to the GAW17 data shows some mixed results: they work better in some genes with a slight margin, but other pre-existing methods beat them in other genes. In terms of penalized regression methods' power, overall, the gain is quite small when it works, and there is no uniform winner.

2.4 Discussion

In this study, we conducted global hypothesis testing for quantitative trait-genetic variants association with new penalized regression methods, which smoothes the variants' absolute effect sizes $|\beta_i| \approx |\beta_j|$ in a data-adaptive way for both simulated data and GAW17 data. Though many penalized regression methods have been developed recently with their good performance in outcome prediction reported in the literature, there is a lack of studies to investigate their performance in hypothesis testing, in which their advantages are not clear. Motivated from that, we used a new penalty TLP for variable selection (TLPS) and for parameter grouping (TLPSG), in the estimation of regression coefficients.

Out of several penalty candidates, we chose the TLP-grouping for the following reasons. First, it overcomes the main limitation of the Sum test, which is its decreasing power when the association directions of the variants with the trait are different; TLP-grouping works by grouping the regression coefficients in absolute values. Second, it aggregates genetic information of variants by grouping, which also reduces the cost of large DF. Third, it determines the group number and group memberships data-adaptively, rather than pre-fixed as in the Sum/aSum tests. Lastly, it is one of the most recently developed penalized methods.

The statistical power for the simulation data and GAW17 data seemed to be

increased by TLP based tests in some cases as compared to other methods, but the power was limited compared to the existing SSU or SSUw test on our simulation settings. One possible reason why the TLP based tests did not perform well might be due to its non-optimal tuning parameter selection based on the model selection criteria we adopted (AIC). As an example, in a simulated dataset, when we set the tuning parameters that properly group the variants, the estimates were quite close to the true values, but the corresponding AIC was less desirable, leading to choosing non-optimal tuning parameters. Relevantly, there is a lack of rigorous theory justifying the applicability of AIC in the current context of penalized regression, where AIC usually measures goodness of fit for the maximum likelihood estimate (MLE), but our TLP estimate is not. We also set the number of parameters as the unique number coefficient groups in absolute magnitudes in AIC for the TLPSG, which also might be questionable. Alternatively, we tried another model selection criterion, multi-fold cross validation. But for RVs as considered here, if we divide the genotype data into multi-fold, it may result in several monomorphic variants (i.e, all values are 0 in a sub sample), which causes computing issues in model fitting.

In conclusion, to test a quantitative trait-genetic variant association, the TLP-penalized regression method was competitive to the OLS in some genes for the GAW17 data and some simulated data. However, its full benefit in outcome prediction did not seem to directly translate into substantially improved testing power. In addition to this one, there exist three recent reports [29],[1],[30] questioning the effectiveness of penalized regression in hypothesis testing as we also showed similar conclusions for Lasso and $gflasso$. We do not claim here that any penalized regression method would not outperform exiting global tests; Further investigation on enhanced tuning parameter selection and better test statistics is warranted to explore the use of penalized regression to improve statistical power in hypothesis testing.

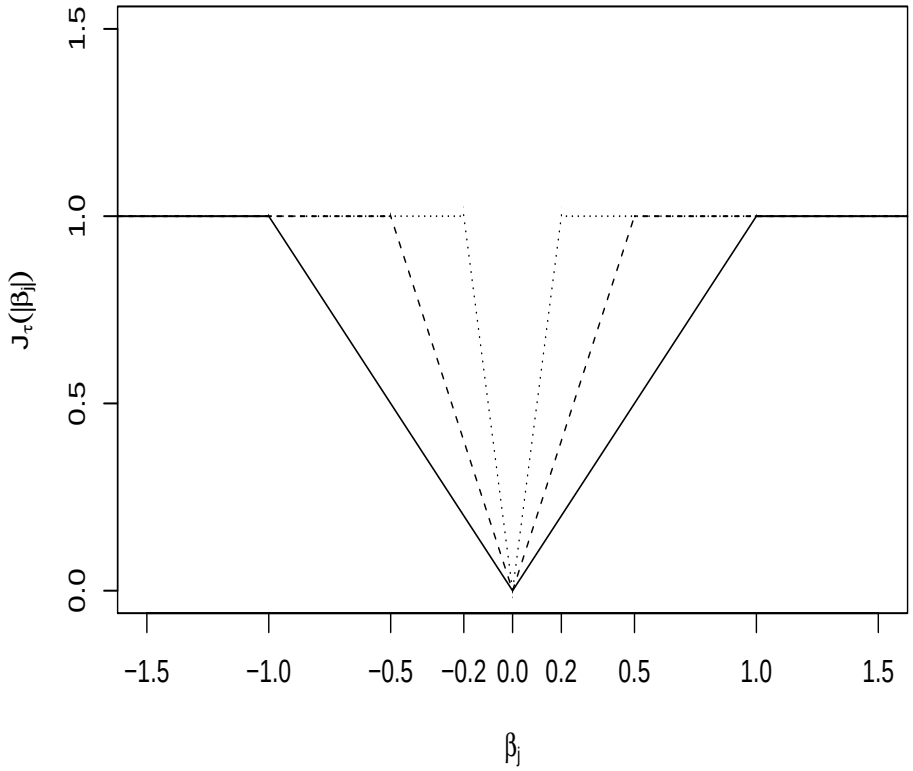
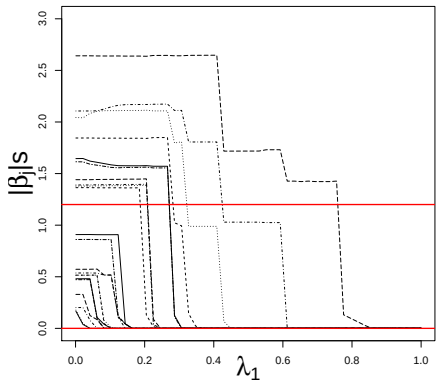
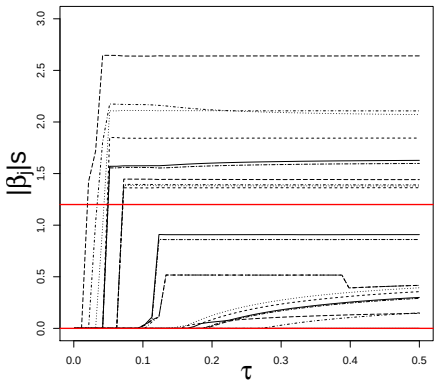


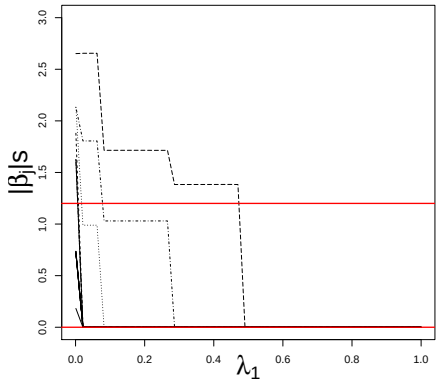
Figure 2.1: Truncated L_1-penalty (TLP) function $J_\tau(|\beta_j|)$ with τ=0.2, 0.5, 1.



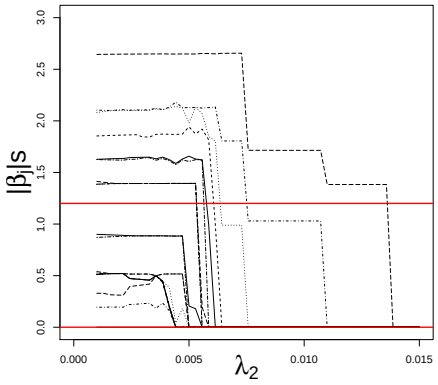
(a) TLPS: $\tau = 0.15$



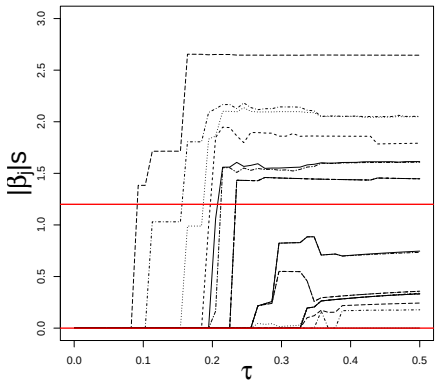
(b) TLPS: $\lambda_1 = 0.1$



(c) TLPSG:(λ_2, τ)=(0.01,0.15)



(d) TLPSG:(λ_1, τ)=(0.02,0.15)



(e) TLPSG:(λ_1, λ_2)=(0.1,0.01)

Figure 2.2: Solution path of $|\beta|$ of a dataset of k=22 in Case 2 of the RVs only set-up for TLPS and TLPSG over the values of a tuning parameter given other(s). The true horizontal solution lines are at 1.2 and 0.

Table 2.1: Empirical Type I error and Power at the nominal level α=0.05 based on 200 replicates for the RVs only set-up with 6 causal RVs and a varying number of non-causal RVs.

		# of non-causal RVs				# of non-causal RVs			
Model	Test	0	8	16	24	0	8	16	24
Selection	Statistics	Null				Case 1			
OLS	F-test	0.030	0.080	0.040	0.060	0.715	0.480	0.340	0.260
OLS	Score	0.030	0.080	0.035	0.055	0.710	0.470	0.320	0.245
OLS	SSU	0.030	0.060	0.045	0.045	0.830	0.660	0.510	0.405
OLS	SSUw	0.035	0.080	0.055	0.060	0.810	0.625	0.500	0.380
OLS	UminP	0.045	0.070	0.050	0.035	0.675	0.445	0.360	0.310
OLS	Sum	0.055	0.075	0.040	0.075	**0.915**	0.685	0.525	0.460
OLS	aSum	0.035	0.065	0.035	0.060	0.910	**0.715**	**0.575**	**0.520**
Lasso	1df	0.055	0.075	0.050	0.080	0.710	0.415	0.325	0.270
$gflasso_{r=cor}$	1df	0.035	0.080	0.050	0.090	0.690	0.415	0.240	0.295
$gflasso_{r=1}$	1df	0.035	0.070	0.050	0.075	0.685	0.375	0.225	0.275
TLPS	1df	0.050	0.085	0.050	0.075	0.720	0.450	0.305	0.255
TLPSG	1df	0.055	0.085	0.055	0.070	0.700	0.450	0.290	0.250
TLPSG	SSU	0.055	0.080	0.040	0.060	0.700	0.520	0.440	0.390
TLPSG	SSUw	0.040	0.075	0.045	0.070	0.790	0.500	0.365	0.320
		Case 2				Case 3			
OLS	F-test	**0.635**	0.515	0.440	0.455	0.745	0.640	0.550	0.490
OLS	Score	0.625	0.500	0.425	0.395	0.745	0.635	0.525	0.470
OLS	SSU	0.590	0.530	**0.505**	0.445	0.710	0.645	**0.595**	**0.555**
OLS	SSUw	0.570	0.505	0.475	0.445	0.715	**0.660**	0.570	0.525
OLS	UminP	0.450	0.410	0.400	0.310	0.665	0.595	0.425	0.425
OLS	Sum	0.145	0.125	0.145	0.100	0.485	0.310	0.260	0.215
OLS	aSum	0.450	0.430	0.355	0.340	0.665	0.590	0.535	0.500
Lasso	1df	0.615	0.465	0.405	0.390	**0.765**	0.585	0.465	0.435
$gflasso_{r=cor}$	1df	0.620	0.530	0.435	**0.480**	**0.765**	0.600	0.480	0.520
$gflasso_{r=1}$	1df	0.615	**0.535**	0.435	0.425	0.750	0.585	0.475	0.495
TLPS	1df	0.615	0.505	0.455	0.425	0.760	0.630	0.530	0.475
TLPSG	1df	0.615	0.485	0.445	0.415	0.755	0.605	0.450	0.450
TLPSG	SSU	0.565	0.470	0.460	0.445	0.705	0.605	0.510	0.525
TLPSG	SSUw	0.585	0.505	0.460	0.415	0.745	0.585	0.485	0.475

Table 2.2: Empirical Type I error and Power at the nominal level α=0.05 based on 200 replicates for the RVs+CVs set-up with 6 causal variants and a varying number of non-causal ones.

Model Selection	Test Statistics	# of non-causal variants				# of non-causal variants			
		0	8	16	24	0	8	16	24
		Null				Case 1			
OLS	F-test	0.025	0.045	0.065	0.050	0.760	0.520	0.355	0.385
OLS	Score	0.020	0.045	0.065	0.035	0.760	0.515	0.345	0.350
OLS	SSU	0.060	0.050	0.090	0.030	0.490	0.210	0.125	0.110
OLS	SSUw	0.040	0.035	0.060	0.035	**0.845**	**0.695**	**0.510**	**0.510**
OLS	UminP	0.030	0.055	0.060	0.025	0.715	0.540	0.380	0.410
OLS	Sum	0.055	0.060	0.075	0.045	0.695	0.450	0.315	0.315
OLS	aSum	0.050	0.060	0.065	0.045	0.665	0.435	0.325	0.340
Lasso	1df	0.030	0.045	0.060	0.045	0.750	0.515	0.360	0.375
$gflasso_{r=cor}$	1df	0.030	0.030	0.070	0.015	0.760	0.450	0.275	0.415
$gflasso_{r=1}$	1df	0.030	0.030	0.070	0.015	0.765	0.455	0.290	0.385
TLPS	1df	0.035	0.050	0.050	0.030	0.750	0.540	0.360	0.370
TLPSG	1df	0.035	0.035	0.065	0.045	0.750	0.515	0.335	0.315
TLPSG	SSU	0.075	0.060	0.055	0.065	0.495	0.230	0.140	0.105
TLPSG	SSUw	0.030	0.055	0.055	0.045	**0.845**	0.675	0.435	0.375
		Case 2				Case 3			
OLS	F-test	0.800	**0.765**	**0.720**	0.650	0.655	0.585	0.415	0.375
OLS	Score	0.800	0.755	0.710	0.630	0.645	0.580	0.400	0.360
OLS	SSU	0.275	0.175	0.155	0.160	0.200	0.140	0.110	0.105
OLS	SSUw	0.715	0.705	0.715	**0.665**	0.640	**0.615**	**0.485**	**0.415**
OLS	UminP	0.640	0.615	0.550	0.505	0.530	0.510	0.370	0.345
OLS	Sum	0.190	0.120	0.125	0.100	0.195	0.150	0.090	0.110
OLS	aSum	0.345	0.275	0.270	0.315	0.290	0.225	0.195	0.210
Lasso	1df	**0.805**	0.695	0.640	0.585	0.580	0.555	0.415	0.360
$gflasso_{r=cor}$	1df	0.810	0.725	0.625	0.655	0.595	0.570	0.420	**0.415**
$gflasso_{r=1}$	1df	**0.805**	0.725	0.620	0.655	0.590	0.570	0.435	0.395
TLPS	1df	0.790	0.730	0.680	0.615	0.600	0.570	0.395	0.390
TLPSG	1df	0.795	0.730	0.620	0.600	0.600	0.555	0.400	0.310
TLPSG	SSU	0.310	0.185	0.165	0.210	0.205	0.120	0.125	0.120
TLPSG	SSUw	0.750	0.720	0.650	0.550	**0.675**	0.560	0.460	0.390

Table 2.3: Mean(sd) of ME out of 200 replicates of the RVs only set-up.

Method	# of non-causal RVs			
	0	8	16	24
	Null			
OLS	12.2(6.1)	29.0(11.7)	43.5(11.7)	61.0(16.7)
Lasso	2.5(4.4)	3.7(6.5)	3.2(5.7)	5.4(11.9)
$gflasso_{r=cor}$	3.5(4.5)	3.9(7.6)	2.5(6.6)	3.6(11.8)
$gflasso_{r=1}$	3.7(4.5)	4.8(7.8)	3.5(6.7)	4.7(12.0)
TLPS	7.2(7.8)	16.8(11.8)	22.8(12.0)	31.0(16.3)
TLPSG	7.7(7.0)	14.9(13.1)	18.9(14.4)	25.3(20.1)
	Case 1			
OLS	11.6(6.8)	27.1(10.3)	43.4(13.4)	58.2(15.1)
Lasso	10.1(6.0)	13.6(6.1)	15.1(6.8)	16.6(8.1)
$gflasso_{r=cor}$	7.7(4.8)	14.5(6.7)	16.1(7.9)	17.6(9.5)
$gflasso_{r=1}$	5.4(5.2)	11.5(6.6)	13.1(7.7)	15.7(9.5)
TLPS	11.5(6.1)	20.3(9.8)	27.7(11.6)	34.0(14.0)
TLPSG	10.6(7.3)	20.7(9.4)	26.7(12.2)	31.7(15.2)
	Case 2			
OLS	11.9(7.2)	26.6(8.9)	43.1(11.7)	59.1(14.4)
Lasso	12.3(5.8)	16.2(6.2)	18.2(7.3)	19.8(7.7)
$gflasso_{r=cor}$	9.9(5.4)	15.5(6.2)	18.8(7.8)	20.7(9.2)
$gflasso_{r=1}$	10.5(5.5)	16.2(6.5)	19.5(7.8)	21.3(9.0)
TLPS	13.4(6.3)	21.8(8.3)	28.9(10.7)	35.5(12.7)
TLPSG	12.3(7.9)	22.8(8.2)	28.5(11.3)	35.0(13.0)
	Case 3			
OLS	12.0(6.8)	27.1(10.6)	44.6(12.5)	60.5(16.7)
Lasso	11.7(5.7)	17.3(7.3)	19.6(7.6)	22.1(9.0)
$gflasso_{r=cor}$	9.7(5.2)	17.3(7.6)	20.0(8.1)	24.6(11.6)
$gflasso_{r=1}$	10.3(5.2)	17.7(7.6)	20.8(8.2)	25.1(11.5)
TLPS	13.6(5.7)	22.1(10.0)	30.8(11.3)	37.1(14.4)
TLPSG	11.5(7.2)	23.2(9.7)	30.4(11.1)	36.7(14.7)

Table 2.4: Mean(sd) of ME out of 200 replicates of the RVs+CVs set-up.

Method	# of non-causal variants			
	0	8	16	24
	Null			
OLS	16.6(13.2)	34.0(17.2)	50.2(21.4)	63.7(17.3)
Lasso	8.9(11.2)	10.4(12.8)	12.0(17.3)	9.1(10.8)
$gflasso_{r=cor}$	11.2(12.2)	10.0(12.6)	10.2(18.5)	4.8(8.0)
$gflasso_{r=1}$	11.3(12.1)	10.6(12.6)	11.4(18.7)	6.6(9.4)
TLPS	12.7(13.2)	22.8(16.5)	31.4(21.1)	36.7(17.6)
TLPSG	9.0(14.4)	17.2(18.6)	24.2(24.7)	27.2(22.5)
	Case 1			
OLS	18.8(20.2)	35.7(22.4)	46.7(16.6)	66.0(20.2)
Lasso	17.9(19.0)	23.6(21.0)	22.7(12.6)	24.8(13.5)
$gflasso_{r=cor}$	16.7(19.3)	23.6(21.0)	24.1(13.7)	25.8(13.4)
$gflasso_{r=1}$	16.1(19.4)	21.8(20.3)	22.0(13.8)	24.6(13.8)
TLPS	19.1(19.8)	29.6(22.0)	33.7(16.5)	45.4(19.3)
TLPSG	20.0(20.1)	29.7(22.2)	32.8(16.8)	42.2(20.7)
	Case 2			
OLS	23.0(48.3)	41.1(45.0)	49.8(16.2)	67.3(21.8)
Lasso	24.0(49.7)	32.3(42.0)	28.6(13.7)	33.8(17.1)
$gflasso_{r=cor}$	22.3(48.3)	32.2(42.6)	29.7(14.7)	36.0(17.1)
$gflasso_{r=1}$	22.5(48.2)	32.5(42.5)	30.5(14.9)	36.9(17.3)
TLPS	24.5(48.5)	37.1(43.5)	37.8(15.5)	48.8(20.8)
TLPSG	23.9(47.0)	38.4(42.5)	39.1(17.4)	47.7(21.8)
	Case 3			
OLS	24.0(35.5)	36.4(28.6)	48.4(17.0)	61.9(18.7)
Lasso	24.1(34.8)	25.9(28.7)	25.5(12.1)	26.0(13.1)
$gflasso_{r=cor}$	22.8(35.2)	25.8(28.5)	26.3(12.5)	27.3(14.0)
$gflasso_{r=1}$	22.9(35.2)	26.1(28.5)	26.6(12.4)	27.9(14.3)
TLPS	24.6(35.1)	31.5(28.7)	36.8(16.7)	42.3(17.5)
TLPSG	25.2(34.6)	32.0(28.7)	36.5(16.1)	41.1(18.9)

Table 2.5: Mean numbers of TP(sd)/FP(sd) of the methods in the RVs only set-up. TP is 0 in null case, denoted as '.' in front of '/'. In non-null cases, when k=6, FP is also 0, denoted as '.' after '/'.

Method	Null			
	./6	./14	./22	./30
OLS	./5.9(0.3)	5.9(0.4)/7.8(0.4)	5.9(0.3)/15.7(0.6)	5.9(0.3)/23.5(0.7)
Lasso	./0.8(1.2)	0.6(1.1)/0.8(1.4)	0.4(0.8)/1.1(1.9)	0.5(0.9)/1.9(3.2)
$gflasso_{r=cor}$	./3.4(2.3)	1.4(2.1)/1.9(2.8)	0.6(1.4)/1.6(3.6)	0.5(1.2)/2.1(4.7)
$gflasso_{r=1}$	./2.8(2.2)	2.3(2.5)/3.0(3.4)	1.7(2.5)/4.4(6.6)	1.5(2.4)/6.1(9.7)
TLPS	./3.4(1.3)	2.9(1.3)/3.6(1.6)	2.6(1.2)/6.9(2.4)	2.6(1.3)/10.5(3.0)
TLPSG	./2.0(2.1)	1.3(1.7)/1.5(2.0)	0.8(1.0)/2.4(2.2)	0.9(1.0)/3.6(3.2)
	Case 1			
	6/0	6/8	6/16	6/24
OLS	5.9(0.3)/.	5.9(0.4)/7.8(0.4)	5.9(0.3)/15.7(0.6)	5.9(0.3)/23.5(0.6)
Lasso	3.3(1.7)/.	2.6(1.6)/1.7(1.7)	2.0(1.7)/2.4(2.5)	1.7(1.6)/2.9(3.4)
$gflasso_{r=cor}$	5.1(1.3)/.	3.8(2.1)/4.4(3.1)	2.4(2.3)/4.6(5.5)	1.7(1.9)/4.1(6.2)
$gflasso_{r=1}$	5.2(1.2)/.	5.1(1.6)/6.1(2.6)	4.3(2.3)/10.0(6.6)	4.0(2.5)/14.5(10.7)
TLPS	4.2(1.2)/.	3.7(1.3)/3.5(1.5)	3.3(1.3)/6.9(2.3)	3.5(1.2)/9.9(3.2)
TLPSG	4.2(1.8)/.	3.0(2.0)/3.0(3.1)	2.1(1.3)/2.7(2.5)	1.9(1.2)/3.6(2.8)
	Case 2			
	6/0	6/8	6/16	6/24
OLS	5.9(0.3)/.	5.9(0.4)/7.9(0.4)	5.9(0.3)/15.7(0.5)	5.9(0.3)/23.5(0.7)
Lasso	3.0(2.0)/.	2.5(1.8)/1.9(1.9)	2.5(1.7)/3.1(3.2)	2.3(1.7)/3.5(3.6)
$gflasso_{r=cor}$	4.7(1.5)/.	4.0(2.2)/4.2(2.9)	3.0(2.3)/5.8(5.9)	2.5(2.3)/6.6(8.1)
$gflasso_{r=1}$	4.2(1.7)/.	4.0(2.0)/4.4(2.9)	3.6(2.3)/7.6(6.6)	3.3(2.5)/10.4(10.1)
TLPS	4.1(1.1)/.	3.8(1.3)/3.6(1.6)	3.9(1.2)/6.8(2.3)	4.0(1.3)/10.3(3.3)
TLPSG	4.2(2.0)/.	3.4(2.0)/3.3(3.0)	2.8(1.5)/3.4(3.5)	2.6(1.4)/4.2(2.9)
	Case 3			
	6/0	6/8	6/16	6/24
OLS	5.9(0.3)/.	5.9(0.4)/7.9(0.4)	5.9(0.3)/15.7(0.5)	5.9(0.3)/23.5(0.7)
Lasso	3.7(1.8)/.	2.9(1.7)/1.9(1.9)	2.6(1.8)/3.1(2.8)	2.5(1.8)/4.1(4.0)
$gflasso_{r=cor}$	5.1(1.3)/.	4.2(1.9)/4.3(2.9)	3.3(2.3)/7.0(5.9)	2.9(2.3)/6.9(7.5)
$gflasso_{r=1}$	4.7(1.5)/.	4.3(1.9)/4.6(2.9)	4.1(2.2)/9.2(6.3)	3.7(2.3)/11.3(9.9)
TLPS	4.5(1.2)/.	4.1(1.2)/3.6(1.5)	3.9(1.3)/7.1(2.4)	4.0(1.2)/9.9(3.1)
TLPSG	4.6(1.7)/.	3.5(1.9)/3.1(3.2)	2.7(1.7)/3.5(3.7)	2.8(1.4)/4.1(2.7)

Table 2.6: Mean numbers of TP(sd)/FP(sd) of the methods in the RVs+CVs set-up. TP is 0 in null case, denoted as '.' in front of '/'. In non-null cases, when k=6, FP is also 0, denoted as '.' after '/'.

Method	Null			
	./6	./14	./22	./30
OLS	./5.7(1.1)	5.9(0.3)/7.9(0.4)	5.9(0.4)/15.7(0.5)	6.0(0.2)/23.5(0.7)
Lasso	./1.5(1.2)	0.9(1.0)/1.3(1.2)	0.7(1.0)/1.8(2.0)	0.6(0.9)/2.2(2.6)
$gflasso_{r=cor}$	./3.9(1.8)	2.0(2.1)/2.8(2.6)	0.9(1.5)/2.4(3.8)	0.3(0.9)/1.5(2.5)
$gflasso_{r=1}$	./3.6(1.7)	2.5(2.4)/3.3(3.0)	2.4(2.6)/6.4(6.9)	2.2(2.8)/8.7(10.8)
TLPS	./4.1(1.3)	3.3(1.3)/4.3(1.4)	2.8(1.2)/7.6(2.1)	2.9(1.3)/11.0(3.0)
TLPSG	./1.1(1.7)	1.3(1.7)/1.7(2.2)	1.1(1.3)/2.6(3.2)	1.1(1.3)/4.0(4.2)
	Case 1			
	6/0	6/8	6/16	6/24
OLS	5.9(0.3)/.	5.9(0.3)/7.9(0.4)	5.9(0.3)/15.7(0.5)	6.0(0.2)/23.5(0.7)
Lasso	3.9(1.4)/.	3.0(1.7)/2.6(1.8)	2.4(1.6)/3.2(2.8)	2.3(1.6)/4.4(4.1)
$gflasso_{r=cor}$	5.2(0.9)/.	4.5(1.8)/5.1(2.7)	2.8(2.1)/5.8(5.8)	2.2(1.9)/5.3(6.1)
$gflasso_{r=1}$	5.2(1.0)/.	5.3(1.3)/6.2(2.2)	4.6(2.1)/11.1(6.2)	4.6(2.2)/16.2(9.4)
TLPS	5.0(1.0)/.	4.2(1.1)/4.3(1.6)	3.8(1.3)/7.2(2.3)	3.7(1.2)/10.7(3.0)
TLPSG	4.3(1.8)/.	3.6(2.0)/3.6(3.1)	2.7(1.6)/4.2(4.2)	2.6(1.4)/5.1(3.9)
	Case 2			
	6/0	6/8	6/16	6/24
OLS	5.9(0.2)/.	5.9(0.3)/7.9(0.4)	5.9(0.3)/15.7(0.5)	6.0(0.2)/23.5(0.7)
Lasso	4.4(1.7)/.	3.7(1.6)/2.9(2.1)	3.5(1.7)/4.7(3.3)	3.2(1.7)/5.9(4.2)
$gflasso_{r=cor}$	5.4(1.0)/.	4.8(1.5)/5.1(2.4)	4.1(2.0)/8.0(5.0)	3.5(2.2)/9.3(7.5)
$gflasso_{r=1}$	5.2(1.1)/.	4.5(1.6)/4.8(2.4)	4.3(1.9)/8.9(5.4)	4.1(2.1)/12.3(8.7)
TLPS	5.4(0.9)/.	4.7(1.1)/4.3(1.5)	4.4(1.1)/7.5(2.1)	4.3(1.1)/11.0(2.9)
TLPSG	4.7(2.0)/.	4.3(1.8)/4.2(3.2)	3.6(1.6)/5.3(4.5)	3.5(1.5)/6.3(5.0)
	Case 3			
	6/0	6/8	6/16	6/24
OLS	5.5(1.5)/.	5.9(0.3)/7.9(0.4)	5.9(0.3)/15.7(0.5)	5.9(0.2)/23.5(0.7)
Lasso	3.5(2.0)/.	3.1(1.8)/2.6(1.9)	2.6(1.7)/4.0(3.3)	2.4(1.8)/4.4(3.9)
$gflasso_{r=cor}$	4.8(1.7)/.	4.6(1.6)/5.0(2.5)	3.1(2.2)/6.2(5.3)	2.5(2.2)/6.2(7.0)
$gflasso_{r=1}$	4.6(1.7)/.	4.5(1.8)/4.9(2.7)	3.8(2.2)/8.4(6.0)	3.6(2.4)/11.5(10.0)
TLPS	4.8(1.6)/.	4.4(1.1)/4.3(1.5)	3.9(1.1)/7.6(2.3)	4.1(1.2)/10.7(2.9)
TLPSG	3.7(2.3)/.	3.8(1.9)/3.8(3.0)	3.0(1.6)/4.7(4.1)	3.0(1.5)/5.4(4.3)

Table 2.7: Mean, sd, and MSE of causal (β_{cs}) and noncausal (β_{ncs}) RVs' regression coefficient estimates when k=30 in the RVs only set-up.

Methods	Mean	sd	MSE	Mean	sd	MSE	Mean	sd	MSE
	Null								
	β_{ncs}=0			β_{ncs}=0			β_{ncs}=0		
OLS	0.10	1.27	3.25	-0.13	1.29	3.35	0.01	1.23	3.03
Lasso	0.00	0.27	0.15	-0.01	0.18	0.06	-0.04	0.28	0.16
$gflasso_{r=cor}$	-0.00	0.23	0.10	-0.02	0.26	0.13	-0.02	0.20	0.08
$gflasso_{r=1}$	0.00	0.24	0.11	-0.01	0.27	0.15	-0.01	0.22	0.09
TLPS	0.06	0.71	1.01	-0.02	0.64	0.81	-0.07	0.72	1.04
TLPSG	0.08	0.64	0.83	-0.05	0.56	0.63	-0.04	0.63	0.80
	Case1								
	β_{cs}=0.9			β_{cs}=0.9			β_{ncs}=0		
OLS	0.97	1.35	3.64	0.70	1.34	3.60	0.11	1.39	3.89
Lasso	0.29	0.54	0.96	0.24	0.49	0.91	0.08	0.36	0.26
$gflasso_{r=cor}$	0.19	0.47	0.95	0.14	0.38	0.87	0.06	0.30	0.18
$gflasso_{r=1}$	0.27	0.43	0.77	0.24	0.36	0.70	0.17	0.31	0.22
TLPS	0.71	0.90	1.67	0.51	0.83	1.53	0.11	0.78	1.21
TLPSG	0.56	0.89	1.70	0.46	0.80	1.47	0.13	0.69	0.97
	Case2								
	β_{cs}=1.2			β_{cs}=-1.2			β_{ncs}=0		
OLS	1.01	1.12	2.53	-1.33	1.55	4.79	-0.14	1.28	3.30
Lasso	0.32	0.52	1.32	-0.39	0.59	1.34	-0.01	0.31	0.20
$gflasso_{r=cor}$	0.26	0.48	1.35	-0.30	0.51	1.33	0.00	0.25	0.12
$gflasso_{r=1}$	0.26	0.49	1.35	-0.29	0.52	1.37	0.01	0.26	0.13
TLPS	0.73	0.87	1.73	-0.82	0.94	1.91	-0.02	0.68	0.91
TLPSG	0.62	0.88	1.89	-0.72	0.95	2.04	0.02	0.61	0.75
	Case3								
	β_{cs}=1.4			β_{cs}=1.3			β_{ncs}=0		
OLS	1.36	1.18	2.77	1.33	1.56	4.88	-0.02	1.23	3.00
Lasso	0.46	0.65	1.72	0.50	0.76	1.64	0.01	0.30	0.18
$gflasso_{r=cor}$	0.44	0.67	1.81	0.45	0.72	1.61	-0.01	0.33	0.22
$gflasso_{r=1}$	0.46	0.66	1.74	0.47	0.71	1.54	0.03	0.35	0.24
TLPS	0.98	1.03	2.30	0.94	1.06	2.31	-0.01	0.55	0.61
TLPSG	0.85	1.05	2.49	0.87	1.07	2.40	-0.01	0.50	0.49

Table 2.8: Mean, sd, and MSE of causal (β_{cs}) and noncausal (β_{ncs}) variants' regression coefficient estimates when k=30 in the RVs+CVs set-up.

Methods	Mean	sd	MSE	Mean	sd	MSE	Mean	sd	MSE
	Null								
	β_{ncs}=0			β_{ncs}=0			β_{ncs}=0		
OLS	0.11	1.28	3.29	0.20	1.43	4.13	-0.22	1.33	3.56
Lasso	0.03	0.49	0.48	0.14	0.45	0.42	-0.14	0.59	0.71
$gflasso_{r=cor}$	-0.05	0.25	0.13	0.05	0.20	0.08	0.02	0.16	0.05
$gflasso_{r=1}$	-0.01	0.17	0.06	0.03	0.15	0.04	0.02	0.11	0.03
TLPS	0.10	0.60	0.72	0.19	0.77	1.23	-0.06	0.80	1.29
TLPSG	0.02	0.84	1.39	0.35	1.01	2.13	-0.03	0.92	1.68
	Case1								
	β_{cs}=1.0			β_{cs}=1.0			β_{ncs}=0		
OLS	0.97	1.33	3.52	1.05	1.40	3.91	-0.04	1.57	4.89
Lasso	0.59	0.66	1.04	0.67	0.70	1.09	0.04	0.43	0.37
$gflasso_{r=cor}$	0.50	0.68	1.18	0.53	0.69	1.18	0.06	0.45	0.41
$gflasso_{r=1}$	0.42	0.56	0.96	0.43	0.57	0.97	0.12	0.35	0.26
TLPS	0.84	1.06	2.27	0.96	1.03	2.11	0.03	0.93	1.74
TLPSG	1.01	1.07	2.27	1.06	1.02	2.09	0.03	0.83	1.37
	Case2								
	β_{cs}=1.5			β_{cs}=1.5			β_{ncs}=0		
OLS	1.59	1.26	3.16	1.54	1.53	4.69	-0.04	1.37	3.77
Lasso	0.93	0.87	1.82	0.84	0.81	1.74	0.01	0.47	0.45
$gflasso_{r=cor}$	0.88	0.93	2.11	0.80	0.86	1.96	0.01	0.57	0.64
$gflasso_{r=1}$	0.83	0.92	2.15	0.74	0.86	2.05	0.02	0.55	0.61
TLPS	1.35	1.14	2.60	1.25	1.15	2.70	-0.03	0.85	1.45
TLPSG	1.28	1.16	2.72	1.29	1.15	2.70	0.01	0.85	1.44
	Case3								
	β_{cs}=1.1			β_{cs}=1.3			β_{ncs}=0		
OLS	1.12	1.31	3.42	1.27	1.40	3.90	0.00	1.29	3.32
Lasso	0.63	0.72	1.26	0.72	0.70	1.30	0.02	0.29	0.17
$gflasso_{r=cor}$	0.56	0.82	1.63	0.52	0.69	1.55	0.02	0.30	0.18
$gflasso_{r=1}$	0.53	0.78	1.55	0.50	0.66	1.51	0.05	0.30	0.18
TLPS	0.99	1.07	2.31	1.09	1.10	2.46	-0.02	0.72	1.05
TLPSG	1.11	1.10	2.42	1.13	1.17	2.74	0.01	0.69	0.94

Table 2.9: MAF(%) and Correlation(COR) in the values of (min, mean, max) for the 12 genes influencing the quantitative trait Q2 in the GAW17 data.

Gene		All	Causal	non-Causal
PLAT	MAF	(0.072,2.098,45.12)	(0.072,0.206,0.574)	(0.072,2.855,45.12)
SREBF1		(0.072,0.699,7.747)	(0.072,0.222,0.43)	(0.072,1.04,7.747)
SIRT1		(0.072,0.858,16.71)	(0.072,0.12,0.215)	(0.072,1.332,16.71)
VLDLR		(0.072,1.047,9.469)	(0.072,0.126,0.287)	(0.072,1.435,9.469)
VNN3		(0.072,4.429,40.53)	(0.072,2.06,9.828)	(0.072,6.501,40.53)
PDGFD		(0.072,4.115,31.56)	(0.072,0.287,0.861)	(0.072,6.303,31.56)
BCHE		(0.072,0.625,14.56)	(0.072,0.105,0.287)	(0.072,1.076,14.56)
INSIG1		(0.072,0.775,3.587)	(0.072,0.072,0.072)	(0.072,1.829,3.587)
LPL		(0.072,1.854,14.490)	(0.072,0.598,1.578)	(0.072,2.076,14.490)
RARB		(0.072,0.352,1.363)	(0.072,0.287,0.502)	(0.072,0.367,1.363)
VNN1		(0.072,2.675,17.070)	(0.574,8.824,17.070)	(0.072,0.215,0.359)
VWF		(0.072,0.944,2.080)	(0.072,0.323,0.574)	(0.359,1.255,2.080)
PLAT	COR	(-0.143,0.002,0.753)	(-0.008,-0.003,-0.001)	(-0.143,0.007,0.753)
SREBF1		(-0.038,0.007,0.635)	(-0.009,-0.004,-0.001)	(-0.038,0.024,0.635)
SIRT1		(-0.044,0.004,0.707)	(-0.004,0.007,0.33)	(-0.044,0.002,0.499)
VLDLR		(-0.135,-0.001,0.331)	(-0.003,-0.002,-0.001)	(-0.135,0.001,0.331)
VNN3		(-0.422,-0.002,0.59)	(-0.104,-0.01,0.072)	(-0.422,-0.001,0.341)
PDGFD		(-0.156,-0.007,0.276)	(-0.007,-0.004,-0.001)	(-0.156,-0.007,0.276)
BCHE		(-0.044,0.001,0.499)	(-0.005,0.004,0.499)	(-0.044,-0.002,0.075)
INSIG1		(-0.010,0.009,0.128)	(-0.001,-0.001,-0.001)	(0.128,0.128,0.128)
LPL		(-0.138,-0.002,0.215)	(-0.010,-0.006,-0.002)	(-0.138,-0.002,0.215)
RARB		(-0.025,-0.003,0.073)	(-0.004,-0.004,-0.004)	(-0.025,-0.005,-0.001)
VNN1		(-0.046,0.038,0.945)	(0.055,0.055,0.055)	(-0.005,0.091,0.945)
VWF		(0.113,0.316,0.564)	(0.265,0.265,0.265)	(0.127,0.246,0.466)

Table 2.10: Empirical Type I error based on the GAW17 data from 200 replicates of Q2, k and nC denote the numbers of the total and causal variants in the data.

		Gene(k, nC)					
Model Selection	Test Stats	PLAT (28,8)	SREBF1 (24,10)	SIRT1 (23,9)	VLDLR (27,8)	VNN3 (15,7)	PDGFD (11,4)
OLS	F-test	0.065	0.065	0.055	0.045	0.045	0.070
OLS	Score	0.055	0.055	0.055	0.040	0.045	0.070
OLS	SSU	0.050	0.025	0.055	0.025	0.040	0.040
OLS	SSUw	0.060	0.040	0.040	0.045	0.060	0.050
OLS	UminP	0.065	0.040	0.035	0.040	0.050	0.045
OLS	Sum	0.050	0.040	0.055	0.070	0.075	0.060
OLS	aSum	0.050	0.045	0.065	0.045	0.075	0.045
Lasso	1df	0.045	0.065	0.065	0.045	0.030	0.035
$gflasso_{r=cor}$	1df	0.045	0.065	0.065	0.045	0.030	0.035
$gflasso_{r=1}$	1df	0.050	0.060	0.065	0.045	0.030	0.035
TLPS	1df	0.080	0.055	0.060	0.040	0.045	0.055
TLPSG	1df	0.000	0.030	0.025	0.005	0.025	0.035
TLPSG	SSU	0.040	0.020	0.050	0.020	0.045	0.045
TLPSG	SSUw	0.005	0.015	0.015	0.010	0.010	0.030
		BCHE (28,13)	INSIG1 (5,3)	LPL (20,3)	RARB (11,2)	VNN1 (7,2)	VWF (6,2)
OLS	F-test	0.070	0.035	0.035	0.020	0.055	0.080
OLS	Score	0.065	0.035	0.035	0.020	0.055	0.080
OLS	SSU	0.030	0.050	0.040	0.040	0.045	0.075
OLS	SSUw	0.060	0.045	0.045	0.020	0.045	0.060
OLS	UminP	0.035	0.050	0.030	0.025	0.030	0.060
OLS	Sum	0.020	0.045	0.025	0.035	0.055	0.055
OLS	aSum	0.030	0.030	0.045	0.060	0.060	0.070
Lasso	1df	0.075	0.035	0.055	0.015	0.050	0.080
$gflasso_{r=cor}$	1df	0.075	0.035	0.055	0.015	0.050	0.080
$gflasso_{r=1}$	1df	0.080	0.040	0.050	0.015	0.050	0.080
TLPS	1df	0.065	0.040	0.045	0.020	0.055	0.080
TLPSG	1df	0.015	0.055	0.010	0.005	0.045	0.070
TLPSG	SSU	0.030	0.050	0.050	0.030	0.050	0.040
TLPSG	SSUw	0.015	0.050	0.000	0.005	0.045	0.045

Table 2.11: Empirical Power based on the GAW17 data from 200 replicates of Q2, k and nC denote the numbers of the total and causal variants in the data.

		Gene(k, nC)					
Model Selection	Test Stats	PLAT (28,8)	SREBF1 (24,10)	SIRT1 (23,9)	VLDLR (27,8)	VNN3 (15,7)	PDGFD (11,4)
OLS	F-test	0.070	0.275	0.360	**0.155**	**0.640**	**0.340**
OLS	Score	0.060	0.260	0.355	**0.155**	**0.640**	0.335
OLS	SSU	0.040	0.025	0.355	0.055	0.185	0.060
OLS	SSUw	0.035	0.245	0.445	**0.155**	0.555	0.320
OLS	UminP	0.065	0.185	0.420	0.120	0.555	0.310
OLS	Sum	0.040	0.075	0.560	0.065	0.410	0.055
OLS	aSum	0.070	0.130	**0.565**	0.095	0.415	0.075
Lasso	1df	0.100	0.270	0.285	0.110	0.595	0.300
$gflasso_{r=cor}$	1df	**0.085**	0.195	0.225	0.135	0.555	0.290
$gflasso_{r=1}$	1df	**0.085**	0.215	0.225	0.135	0.570	0.300
TLPS	1df	0.065	**0.290**	0.330	0.130	0.630	0.325
TLPSG	1df	0.025	0.090	0.165	0.075	0.410	0.195
TLPSG	SSU	0.040	0.015	0.355	0.080	0.220	0.070
TLPSG	SSUw	0.015	0.085	0.225	0.055	0.330	0.205
		BCHE (28,13)	INSIG1 (5,3)	LPL (20,3)	RARB (11,2)	VNN1 (7,2)	VWF (6,2)
OLS	F-test	0.375	0.065	**0.305**	0.135	0.750	0.110
OLS	Score	0.365	0.065	0.295	0.135	0.740	0.110
OLS	SSU	0.040	0.090	0.050	0.100	**0.945**	0.170
OLS	SSUw	**0.405**	0.055	0.300	0.130	0.715	**0.210**
OLS	UminP	0.300	0.060	0.285	0.110	0.820	0.170
OLS	Sum	0.180	0.080	0.030	0.145	0.925	**0.210**
OLS	aSum	0.120	0.100	0.090	0.145	0.935	**0.210**
Lasso	1df	0.315	0.050	0.205	0.135	0.655	0.090
$gflasso_{r=cor}$	1df	0.300	0.055	0.220	0.120	0.720	0.110
$gflasso_{r=1}$	1df	0.300	0.055	0.215	0.125	0.695	0.110
TLPS	1df	0.355	0.060	0.270	**0.160**	0.720	0.110
TLPSG	1df	0.135	0.080	0.115	0.095	0.665	0.080
TLPSG	SSU	0.045	**0.110**	0.040	0.070	**0.945**	0.140
TLPSG	SSUw	0.155	0.075	0.135	0.085	0.675	0.145

Table 2.12: Empirical Type I based on the GAW17 data without CVs from 200 replicates of Q2, k and nC denote the numbers of the total and causal RVs in the data.

		Gene(k, nC)				
Model Fitting	Test Stats	PLAT (26,8)	SREBF1 (23,10)	SIRT1 (22,9)	VLDLR (24,8)	VNN3 (12,6)
OLS	F-test	0.055	0.065	0.055	0.045	0.045
OLS	Score	0.045	0.060	0.050	0.045	0.045
OLS	SSU	0.055	0.045	0.050	0.045	0.060
OLS	SSUw	0.040	0.045	0.040	0.045	0.045
OLS	UminP	0.055	0.035	0.025	0.040	0.055
OLS	Sum	0.035	0.035	0.080	0.060	0.040
OLS	aSum	0.055	0.060	0.055	0.050	0.045
Lasso	1df	0.050	0.065	0.050	0.050	0.040
$gflasso_{r=cor}$	1df	0.050	0.065	0.060	0.030	0.035
$gflasso_{r=1}$	1df	0.055	0.065	0.060	0.030	0.035
TLPS	1df	0.070	0.050	0.045	0.040	0.050
TLPSG	1df	0.000	0.050	0.050	0.005	0.020
TLPSG	SSU	0.020	0.045	0.075	0.040	0.060
TLPSG	SSUw	0.000	0.040	0.040	0.005	0.015
		PDGFD (9,4)	BCHE (27,13)	INSIG1 (4,3)	LPL (17,3)	VNN1 (6,1)
OLS	F-test	0.040	0.060	0.060	0.040	0.055
OLS	Score	0.040	0.060	0.055	0.040	0.055
OLS	SSU	0.055	0.055	0.055	0.040	0.020
OLS	SSUw	0.040	0.055	0.055	0.045	0.055
OLS	UminP	0.040	0.035	0.050	0.035	0.030
OLS	Sum	0.065	0.020	0.045	0.040	0.065
OLS	aSum	0.055	0.065	0.030	0.050	0.040
Lasso	1df	0.045	0.065	0.050	0.030	0.055
$gflasso_{r=cor}$	1df	0.040	0.070	0.065	0.030	0.055
$gflasso_{r=1}$	1df	0.045	0.070	0.055	0.025	0.055
TLPS	1df	0.045	0.060	0.065	0.055	0.060
TLPSG	1df	0.030	0.060	0.065	0.005	0.050
TLPSG	SSU	0.040	0.065	0.065	0.025	0.030
TLPSG	SSUw	0.025	0.055	0.055	0.005	0.055

Table 2.13: Empirical power based on the GAW17 data without CVs from 200 replicates of Q2, k and nC denote the numbers of the total and causal RVs in the data.

		Gene(k, nC)				
Model Fitting	Test Stats	PLAT (26,8)	SREBF1 (23,10)	SIRT1 (22,9)	VLDLR (24,8)	VNN3 (12,6)
OLS	F-test	0.070	**0.295**	0.305	0.135	**0.435**
OLS	Score	0.065	0.295	0.305	0.135	0.430
OLS	SSU	0.040	0.095	**0.430**	0.075	0.225
OLS	SSUw	0.055	0.270	0.375	0.130	0.390
OLS	UminP	0.060	0.190	0.390	0.125	0.410
OLS	Sum	0.055	0.260	0.350	0.105	0.265
OLS	aSum	0.085	0.295	0.380	**0.155**	0.270
Lasso	1df	**0.105**	0.255	0.265	0.140	0.275
$gflasso_{r=cor}$	1df	**0.105**	0.195	0.220	0.110	0.255
$gflasso_{r=1}$	1df	0.100	0.225	0.210	0.110	0.280
TLPS	1df	0.065	0.280	0.295	0.175	0.355
TLPSG	1df	0.020	0.150	0.215	0.045	0.250
TLPSG	SSU	0.035	0.090	0.350	0.070	0.265
TLPSG	SSUw	0.000	0.125	0.235	0.035	0.215
		PDGFD (9,4)	BCHE (27,13)	INSIG1 (4,3)	LPL (17,3)	VNN1 (6,1)
OLS	F-test	**0.395**	0.380	0.035	0.340	0.145
OLS	Score	**0.395**	0.380	0.035	0.335	0.145
OLS	SSU	0.200	0.430	0.035	**0.450**	0.195
OLS	SSUw	0.385	0.405	0.035	0.340	0.155
OLS	UminP	0.330	0.305	0.050	0.305	0.115
OLS	Sum	0.155	**0.505**	0.035	0.145	0.035
OLS	aSum	0.195	0.465	**0.075**	0.245	0.135
Lasso	1df	0.315	0.320	0.030	0.230	0.115
$gflasso_{r=cor}$	1df	0.310	0.305	0.040	0.255	0.130
$gflasso_{r=1}$	1df	0.335	0.305	0.035	0.260	0.135
TLPS	1df	0.360	0.345	0.040	0.320	0.115
TLPSG	1df	0.250	0.430	0.060	0.140	0.110
TLPSG	SSU	0.175	0.450	0.055	0.440	**0.220**
TLPSG	SSUw	0.250	0.455	0.055	0.155	0.150

Table 2.14: Mean numbers of TP(sd)/FP(sd) in the GAW17 data, where $q1$ and $q0$ denote the number of causal RVs and the number of non-causal variants in each gene.

	Lasso	$gflasso_{r=cor}$	$gflasso_{r=1}$	TLPS	TLPSG
Gene($q1/q0$)			all		
PLAT(8/20)	0.8(1.4)/2.1(2.7)	0.6(1.5)/1.9(2.6)	1.5(2.8)/3.8(6.4)	5.0(1.6)/11.4(2.9)	1.7(2.1)/3.9(4.5)
SREBF1(10/14)	2.1(2.6)/2.6(3.0)	2.4(3.1)/3.5(3.9)	6.5(4.4)/8.8(6.1)	6.6(1.6)/8.5(2.4)	2.6(2.1)/3.1(2.2)
SIRT1(9/14)	2.0(1.9)/2.5(2.2)	1.9(2.3)/2.3(3.2)	4.9(3.7)/6.9(5.8)	5.0(1.6)/7.2(2.5)	2.0(1.2)/2.3(1.3)
VLDLR(8/19)	0.8(1.5)/2.7(3.0)	0.8(1.7)/2.6(3.7)	2.1(3.2)/5.4(7.1)	5.0(1.5)/11.7(2.6)	1.5(1.7)/3.8(3.7)
VNN3(7/8)	2.7(1.3)/2.5(1.7)	3.5(1.6)/3.9(2.1)	4.4(2.1)/4.6(2.6)	5.0(1.2)/5.6(1.4)	2.8(1.4)/2.3(1.9)
PDGFD(4/7)	1.2(1.1)/2.3(1.9)	2.1(1.3)/4.2(2.0)	2.5(1.4)/4.4(2.2)	2.9(1.0)/5.3(1.3)	1.2(1.1)/2.0(2.0)
BCHE(13/15)	3.4(3.0)/2.8(2.8)	3.0(3.5)/3.1(3.5)	7.1(5.2)/7.7(6.0)	7.5(2.1)/7.5(2.3)	4.1(3.1)/3.4(3.4)
INSIG1(3/2)	0.2(0.6)/0.4(0.6)	1.0(1.1)/1.1(0.7)	0.8(1.2)/1.1(0.7)	1.6(0.8)/1.4(0.5)	0.7(1.1)/0.7(0.8)
LPL(3/17)	1.0(0.8)/2.9(3.0)	1.1(0.9)/4.0(4.1)	1.4(1.0)/5.4(5.7)	2.5(0.7)/10.8(2.4)	1.2(0.8)/3.2(3.4)
RARB(2/9)	0.7(0.7)/1.2(1.8)	0.8(0.7)/2.1(2.7)	1.0(0.8)/3.2(3.6)	1.6(0.5)/5.1(1.7)	0.8(0.7)/1.4(1.8)
VNN1(2/5)	1.5(0.5)/0.6(1.0)	1.8(0.4)/2.4(1.7)	1.7(0.5)/1.7(1.8)	1.9(0.3)/2.8(1.3)	1.5(0.5)/1.1(1.7)
VWF(2/4)	0.2(0.5)/1.0(1.1)	1.0(0.8)/2.7(1.2)	1.0(0.8)/2.7(1.2)	1.5(0.6)/3.5(0.7)	0.4(0.6)/1.2(1.2)
			w.o CVs		
PLAT(8/18)	1.0(1.6)/1.4(2.5)	0.9(1.8)/1.3(2.7)	1.8(2.9)/3.5(6.2)	5.0(1.6)/9.3(2.7)	1.6(1.6)/2.3(2.4)
SREBF1(10/13)	2.1(2.4)/2.2(2.6)	2.3(2.9)/2.9(3.4)	6.7(4.3)/8.5(5.6)	6.5(1.5)/7.4(2.3)	2.7(2.0)/2.7(1.9)
SIRT1(9/13)	2.0(1.9)/2.0(2.4)	1.9(2.3)/2.1(3.0)	4.9(3.7)/6.5(5.6)	5.0(1.6)/6.7(2.1)	2.0(1.5)/2.1(1.8)
VLDLR(8/16)	1.3(1.7)/1.8(2.6)	0.9(1.7)/1.5(3.0)	2.4(3.3)/4.5(6.4)	4.8(1.5)/8.8(2.5)	1.6(1.4)/2.3(2.2)
VNN3(6/6)	1.8(1.3)/1.2(1.3)	2.4(1.5)/2.0(1.8)	2.9(1.9)/2.4(2.1)	3.8(1.2)/3.7(1.3)	1.8(1.2)/1.2(1.3)
PDGFD(4/5)	1.4(1.1)/1.4(1.5)	2.0(1.3)/2.5(1.5)	2.5(1.4)/2.7(1.9)	2.9(0.9)/3.5(1.2)	1.6(1.2)/1.4(1.6)
BCHE(13/14)	3.6(3.1)/2.2(2.7)	3.1(3.6)/2.5(3.4)	8.1(5.0)/8.1(5.6)	7.4(2.0)/6.4(2.2)	3.8(2.3)/2.2(2.1)
INSIG1(3/1)	0.3(0.7)/0.2(0.4)	0.7(1.1)/0.2(0.4)	0.6(1.0)/0.2(0.4)	1.6(0.8)/0.5(0.5)	1.0(1.2)/0.4(0.5)
LPL(3/14)	1.2(0.8)/2.1(2.8)	1.4(1.0)/3.4(4.3)	1.6(1.0)/4.5(5.5)	2.4(0.7)/7.9(2.4)	1.4(0.8)/2.3(2.3)
VNN1(1/5)	0.5(0.5)/0.6(1.0)	0.8(0.4)/2.1(1.8)	0.7(0.4)/1.7(1.9)	0.9(0.2)/2.7(1.3)	0.6(0.5)/1.2(1.8)

2.5 Programs

[R code]

```
###################################
# simX simulates Rep datasets
# of predictor set (@ dat$ind)
# and records causal variants'
# location (@ dat$nCloc)
# for RVs only case.

# Input
# n   : # of subjects
# p   : # of predictors
# rho : correlation in AR1 matrix
# nC  : # of causal variants
###################################

simXall<-function(Rep,n,p,rho,nC){

 dat=matrix(0,Rep*n,p)

 for(d in 1:Rep){

  set.seed(d)

  #AR1 correlation matrix
  R<-matrix(0, nrow=p, ncol=p)
   for(i in 1:p)
    for(j in 1:p)
     R[i, j]<-rho^(abs(i-j))
    svd.R0<-svd(R)
    R1<-svd.R0$u %*% diag(sqrt(svd.R0$d))

MAF<-runif(p, 0.0025, 0.005)
cutoff<-qnorm(MAF)

Xtmp<-matrix(0, nrow=n, ncol=p)
for( i in 1:n){
  X0<-rnorm(p, 0, 1)
  X1<-R1 %*% X0
  X2<-ifelse(X1<cutoff, 1, 0)
  X0<-rnorm(p, 0, 1)
  X1<-R1 %*% X0
  X3<-ifelse(X1<cutoff, 1, 0)
  X4<-X2+ X3
  Xtmp[i, ]<-X4
  }

#randomly locate causal snps
 indmat=matrix(0,nrow=Rep, ncol=nC)
 posi=sample(1:p,nC)
 indmat[d,]=posi
 X=matrix(0,n,p)
 X[,posi]=Xtmp[,posi]
 X[,-posi]=Xtmp[,-posi]

#prevent monormorphic column
 for(m in 1:p)
  {if(max(X[,m])==0){X[sample(c(1:n),1),m]=1}}

dat[(1+n*(d-1)):(n*d),]=X

}

return(list(nCloc=indmat,dat=dat))

}

###################################
# simY simulates Rep datasets
# of quantitative outcome Y.

# Input
# tX : Rep datasets of predictors
#      of size n*Rep x p
###################################

simY<-function(Rep,tX,b0,b,n,var,nCloc){
 p=ncol(tX)
 dat=matrix(0,Rep*n,1)
 for(j in 1:Rep){
 X=tX[(1+n*(j-1)):(n*j),]
 nCpos=nCloc[j,]

 Y<-rep(0, n);
   for (i in 1:n){
   Y[i]<-rnorm(1,b0+sum(b*X[i,nCpos]),sqrt(var))
   }
 dat[(1+n*(j-1)):(n*j),]=Y
}
return(dat)
}

###################################
# Data generate example
# of p=30 for all cases
###################################

nC=6
#X
```

```
X=simXall(Rep=200,n=400,p=30,rho=0.8,nC=6)
tX=X$dat
nCloc=X$ncloc
bn=c(rep(0,nC))
b1=c(rep(0.9,nC))
b2=c(rep(1.2,nC/2),rep(-1.2,nC/2)))
b3=c(1.4,1.3,-1.2,1.2,-1.3,1.4)

#Null
Y=simY(Rep=200,tX,b0=0.3,bn,n,var=2,nCloc)
#Case1
Y=simY(Rep=200,tX,b0=0.3,b1,n,var=2,nCloc)
#Case2
Y=simY(Rep=200,tX,b0=0.3,b2,n,var=2,nCloc)
#Case3
Y=simY(Rep=200,tX,b0=0.3,b3,n,var=2,nCloc)
```

[MATLAB]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  HT produces empirical
%  power, and the summary of TP/FP
%  for lasso, gflasso_cor, gflasso_r=1,
%  TLPS, and TLPSG.

%  Input
%  dat   : matrix of total data (Y,X)
%  Rep   : # of replicates
%  pm    : number of permutation size B
%  pair0 : matrix of # of edge x 2
%  nCloc : position of causal variants
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function[pow,TPFP]=
 HT(dat,Rep,pm,pair0,nCloc)

tY=dat(:,1);
n=length(tY)/Rep;
%add int
tX=[ones(Rep*n,1) dat(:,2:end)];
op=zeros(Rep,7);
p=size(tX,2);

%consider int
nCloc=1+nCloc;
pair=pair0';

%Test Stats matrix of lasso
%set similarly for other methods.
TS_lasso=zeros(Rep,1+pm);

%TPFP
TP=zeros(Rep,5);
FP=zeros(Rep,5);

%Grid search range
gd=5;
lamf=linspace(0.001,1,gd);
lams=linspace(0.001,0.5,gd);
ltau=linspace(0.001,0.5,gd);

 for d =1:Rep

 disp(d)
 nCpos=nCloc(d,:);
 Y=tY((1+n*(d-1)):(n*d));
 X=tX((1+n*(d-1)):(n*d),:);

% correlation for gflasso_cor
nedge=size(pair0,1);
sn=zeros(nedge,1);
for j=1:size(pair0,1)
sn(j,1)=sign(corr(X(:,pair0(j,1)),
        X(:,pair0(j,2))));
end

% AIC matrix to select model

 [AIC_lasso,AIC_gflasso1,AIC_gflasso2,
  AIC_TLPS,AIC_TLPSG]=
 AIC_FGSG(gd,Y,X,lamf,lams,ltau,pair,sn);

 % F-stat from new 1 df test for lasso

 ind= AIC_lasso==min(AIC_lasso);
 lamlasso=linspace(0.001,10,50);
 opts=[];
 blasso=gflasso(X,Y,pair,lamlasso(ind),0,opts);
 X2=[ones(n,1) X*blasso];
 [bsum,bint,r,rint,stats]=regress_nowarn(Y,X2);

 if bsum(2)==0, TS_lasso(d,1)=0;
 else TS_lasso(d,1)=stats(2); end

% TP and FP
 TP(d,1)= sum(abs(blasso(nCpos))>0.001);
 FP(d,1)= sum(abs(blasso(setxor(nCpos,2:p)))>0.001);

% For other methods, set similarly but
% use the selected lam1/lam2/tau from
% corresponding AIC matrix.

% Note that before run final model
% via the estimated tuning parameters,
```

```
% for gflasso_cor, set, opts.wt=sn;
% for TLPS and TLPSG, set, opts.tau=tau;

% TLPSG SSU/SSUw test stats
 [SSU,SSUw]=Smsq(Y,X,bTLPSG);
 TS_TLPSG_SSU(d,1)=SSU;
 TS_TLPSG_SSUw(d,1)=SSUw;

%% permutation
 for B=1:pm

 Yb=Y(randperm(n));
 % same procedure as above with Yb,
 % just add test stat into TS_*(d,1+B)

 end

end

for t=1:Rep
op(t,1)=sum(sum(TS_lasso(t,1)
                <TS_lasso(:,2:end)));
%similarly for other methods
end

%% power where op/(Rep*pm) is pvalue
 pow=mean(op/(Rep*pm)<=0.05,1);
 TPFP=[mean(TP,1) mean(FP,1);
       sqrt(var(TP,0)) sqrt(var(FP,0))];

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% AIC_FGSG produces AIC matrix for
% all methods.
% For simplicity, only the one of TLPSG
% is presented.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function[AIC_TLPSG]
 =AIC_FGSG(gd,Y,X,lamf,lams,ltau,pair,sn)

AIC_TLPSG=zeros(gd,gd*gd);

opts=[];
for s=1:gd
for t=1:gd
 for u=1:gd
 opts.tau=ltau(u);
 x0=ncTFGS(X,Y,pair,lamf(s),lams(t),opts);
 trss=norm(Y-X*x0)^2;
 % up to 4 decimal digits
 ngrp=length(find(unique(round(
             abs(x0(2:end))
             *10000)/10000)));
 var=trss/(n-ngrp-1);
 ll=-(n/2)*log(var)-trss/(2*var);
 AIC_TLPSG(s,(t-1)*gd+u)=-2*ll+2*ngrp;
 end
end
end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Smsq produces SSU/SSUw test stats
% from bhat of TLPSG
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[Tssu,Tssuw]=Smsq(Y,X,bhat)

%exclude intercept
pos= find(abs(bhat(2:end))>0.001);
X2=X(:,2:end);

if isempty(pos)==0

 Xg=X2;
 Xbar=mean(Xg);
 Xgb=Xg;
 for i=1:size(Xg,1)
  Xgb(i,:)=Xg(i,:)-Xbar;
 end
 var0=var(Y);
 U=1/var0*Xg'*(Y-mean(Y));

%Sub U and V
U2=U(pos);
%SumSqU
Tssu=U2'*U2;
%cov of the score stats;
CovS=1/var0*(Xgb'*Xgb);
CovS2=CovS(pos,pos);
%SSUw
 Tssuw=U2'*diag(ones(length(pos),1)./diag(CovS2))*U2;
else
  Tssu=0;
  Tssuw=0;
end

end
```

Chapter 3

New penalized regression methods for genome-wide selection of RVs associated with a quantitative trait.

3.1 Introduction

GWAS have been successful in discovering CVs associated with complex disease and traits, but the explained proportion still remains quite low. In the mean time, the availability of rare variants (RVs), thanks to the advance in sequencing technologies, enables to find RVs associated with common diseases [31] [32] [33]. However, statistically, the detection of the causal RVs is challenging due to their extremely low frequencies.

One representative remedy to improve detection power of causal RVs is to collapse them to strengthen the association signal. Though it might increase the statistical power in some situations, forced collapsing with the possible presence of noise variants together may also reduce power. To avoid this problem, collapsing on only selected variants might be helpful to increase the detection power. With this point in mind, recently, some penalized regression methods conducting variable selection accompanying grouping for a gene have been applied in GWAS [34] [35] [36] [37] [38]. They have been shown to be effective to detect the informative genetic markers related to a trait.

While the methods applied in previous studies are mostly based on the L_1 or L_2-penalty, this study explores some newly developed non-convex penalty TLP [10] to select informative markers in high dimensional setting ; TLP is a truncated L_1-penalty aiming to reduce regression coefficient estimation bias in L_1-norm by only penalizing the coefficients less than a threshold τ: $J_\tau(|x|) = min(\frac{|x|}{\tau}, 1)$, where τ is determined in a data-adaptive way. In grouping pursuit in a gene, it smoothes a pair of variants' (absolute) regression coefficient values only when they are close within a predetermined degree τ.

Computationally, optimization with the non-convex TLP can be transformed into a convex sub-problem using Difference of Convex (DC) programming [10], and then fitted using a convex solver, e.g. MATLAB package `CVX`. However, the computing time is not well suited for high dimensional problem. Very recently, [19] implemented a quite fast feature grouping solver in C, called Feature Grouping and Selection Over an undirected graph (FGSG) with several grouping functions (graph-OSCAR, graph-Fused Lasso, and etc), and it includes several versions of TLP.

We apply the FGSG package in this study to compare grouping-penalized regression methods' performance when applied to a large number of the genes for the GAW17 data. While our focus is on feature selection, we also compare prediction ability for all traits. Feature selection performance is evaluated in either variant or gene level. First, we evaluate it in the SNP level to see whether a method finds the causal variants which directly influences the trait. Second, we also evaluate it in the gene level since though the method does not find the causal variants which might be hard especially in large scale data, the grouping penalty would encourage the neighbors' selection. Thus, if a gene includes any selected variant, we count the gene as detected. All analyses are based on two quantitative traits Q1 and Q2 over 200 replicates of the GAW17 dataset.

3.2 Method

3.2.1 Penalized regression methods

For a quantitative response vector $Y = (Y_1, ..., Y_n)$, the genotypes of p rare variants $X = (X_1, ...X_p)$ where each component $X_j = (X_{1j}, ..., X_{nj})'$ is a j^{th} rare variant vector, $j = 1, ..., p$, a linear model fits:

$$Y_i = \beta_0 + \sum_{j=1}^{p} X_{ij}\beta_j + \epsilon_i, \;\; i = 1, ..., n, \tag{3.1}$$

where $\beta = (\beta_1, ..., \beta_p)'$ is a regression coefficient vector. However, (3.1) causes some computational issues in a high dimensional setting with $p > n$ which is a common problem in genetic studies. While several penalized regression methods have been developed to resolve these issues, in this study, we apply six methods where five methods are implemented in the FGSG package of [19]: 1) Lasso [3], 2) graph fused lasso (gflasso) [18], 3) graph OSCAR (goscar), 4) non-convex truncated feature grouping and selection (ncTFGS), and its variant, non-convex truncated fused lasso grouping and selection(ncTLF) [19]. Another method is Ridge [34] fitted via glmnet package in R. All methods except Lasso and Ridge attempts grouping.

As first, $\hat{\beta}$ in Lasso is obtained by,

$$\hat{\beta}_{lasso} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j|, \tag{3.2}$$

where the number of non-zero β_js is a decreasing function of λ_1. With only a selection penalty, Lasso focuses on variable selection, and in FGSG, it is fitted by setting λ_2 as 0 in the gflasso function of the following.

The $\hat{\beta}$ in Ridge is obtained by,

$$\hat{\beta}_{ridge} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_2 \sum_{j=1}^{p} \beta_j^2/2. \tag{3.3}$$

Since Ridge does not involve variable selection, we show only its prediction ability later.

Gflasso adds a L_1-group penalty for a gene g, which shrinks the difference of β_j and $\beta_{j'}$ for $(j, j') \in E_g$, $g = 1, ..., G$, where E_g is a set of a gene g's non-directed pairwise edges.

$$\hat{\beta}_{gflasso} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{g=1}^{G} \sum_{j<j',(j,j')\in E_g} |\beta_{jg} - \beta_{j'g}|. \tag{3.4}$$

The gflasso encourages the predictors' regression coefficients in the same group (or gene in this study) to be similarly estimated to reflect their dependency such as SNPs' linkage disequilibrium within a gene.

As another grouping method, we apply goscar in a following form:

$$\hat{\beta}_{goscar} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{g=1}^{G} \sum_{j<j',(j,j')\in E_g} \max\{|\beta_{jg}|, |\beta_{j'g}|\}, \tag{3.5}$$

where the L_∞-norm is used to shrink the regression coefficients' absolute values in a gene g towards each other to accommodate the coefficients' opposite signs. Note that Lasso, gflasso, and goscar are all common in convex forms, but as next, we apply two grouping functions, ncTLF and ncTFGS in non-convex forms. The ncTLF is

a non-convex version of gflasso in which each L_1-norm is replaced by non-convex truncated L_1-norm, $J_\tau(|x|) = min(\frac{|x|}{\tau}, 1)$:

$$\hat{\beta}_{ncTLF} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{p} J_\tau(|\beta_j|) + \lambda_2 \sum_{g=1}^{G} \sum_{j<j',(j,j')\in E_g} J_\tau(\big|\beta_{jg} - \beta_{j'g}\big|), \tag{3.6}$$

where the use of τ attempts to reduce the estimation bias from the L_1-penalty by only penalizing $|\beta_j| < \tau$ in the first selection penalty and $|\beta_j - \beta_{j'}| < \tau$ for $(j, j') \in E_g$ in the second grouping penalty.

The ncTLF encourage $\beta_j \approx \beta_{j'}$, but this might not work well if there exist the coefficients with opposite signs. The ncTFGS considers this problem by taking absolutes of the coefficients in the grouping penalty:

$$\hat{\beta}_{ncTFGS} = \underset{\beta}{\operatorname{argmin}} \frac{1}{2}\|Y - X\beta\|^2 + \lambda_1 \sum_{j=1}^{p} J_\tau(|\beta_j|) + \lambda_2 \sum_{g=1}^{G} \sum_{j<j',(j,j')\in E_g} J_\tau(\big||\beta_{jg}| - |\beta_{j'g}|\big|). \tag{3.7}$$

3.2.2 Data

The GAW17 [25] data set consists of 3,205 autosomal genes with 24,487 variants sequenced on 697 subjects. The genotypes were obtained from the sequence alignment files provided by the 1000 Genomes Project pilot 3 study. The GAW17 data includes 200 replicates of three quantitative traits Q1, Q2, and Q4, where Q1 and Q2 were directly influenced by genetic factors. The quantitative trait Q1 was influenced by 39 SNPs in 9 genes and three environmental factors (age/sex/smoking status), and the genes primarily came from the VEGF pathway. The 13 genes (72 SNPs) influencing Q2 were related to cardiovascular disease risk and inflammation, and Q2 was not influenced by age, sex, and smoking status. The true effect sizes of all genetic variants range from 0.2 to 1.2, so all causal variants were associated with a trait in the same direction though the effect sizes were quite different.

We run two models for each trait; The first model includes only genotypes as covariates (genotypes only model), and the second model adds age/sex/smoking status (combined model). The genotype set consists of all the SNPs in the corresponding

causal genes and also the non-causal SNPs in one hundred randomly chosen non-causal genes; For Q2, as the predictors, we include the 211 and 659 SNPs from 13 causal and 100 non-causal genes respectively. For Q1, we include 125 from 9 causal and 773 SNPs from 100 non-causal genes.

3.2.3 Analysis

Before analyzing data, we center the data; the j^{th} variant X_j, $j = 1, ..., p$ is centered with $X_j - (\sum_{i=1}^{n} X_{ij}/n)\underline{1}_n$. Outcome Y is centered with the mean value of Y_{tr} of training set. The three environmental factors are included in the variant set X for the combined model where each factor is considered as an independent group. Using the FGSG package of [19], the five different methods are mainly evaluated on the feature selection in both the SNP level and gene level. In the SNP level, the $|\hat{\beta}_j|$ larger than 10^{-4} is considered as non-zero. On the other hand, in the gene level, if a gene includes at least one non-zero estimate for one or more SNPs, the gene is considered as non-zero. Out of 200 replicates in the GAW17 dataset, for d=1,...,198, we use d^{th} set as training set to fit the model, and the next (d+1th) set as the tuning set to select the tuning parameters as the ones minimizing the predictive residual sum of squares, and use the following (d+2th) set as test set to evaluate the method's prediction performance. For 199^{th} (200^{th}) replicate, 200^{th} and 1^{st} (1^{st} and 2^{nd}) replicates are used as the tuning and test sets.

The $(\lambda_1, \lambda_2, \tau)$ are searched in 5 x 5 x 5 equally spaced grid points of [0.001,10] x [0.001,10] x [0.01, 0.5u], where u is the maximum component of the Lasso estimate.

3.3 Result

The analysis results are listed in Tables 3.1-3.7. In Table 3.1, in the SNP level, ncTLF and ncTFGS on average identify the true causal SNPs (TPsnp) as often as other methods while yielding false positive SNPs (FPsnp) much less than others. The ratio of the mean TPsnp over the mean FPsnp's (Rsnp) in the 4th column shows that it is around 2.5 times higher for Q1 in the genotypes only model, e.g. 0.1 vs 0.26 for the convex methods vs ncTLF. The ratio increases to 4 times higher in Q2 in the same model, e.g. 0.08 vs 0.30 for the convex methods vs ncTLF. We can see that

TLP grouping reduces false positive numbers since the weakly or non-associated SNPs to a trait in a gene might be estimated all as 0 (grouping effect).

When including three environmental factors in the combined model, for Q1, Rsnp of ncTLF and ncTFGS are 5 times higher than those of the convex methods, Lasso, gflasso, and goscar. While TPsnp is similar to the one in the genotypes only model of Q1, FPsnp is quite less in the combined model since the variation of Q1 is also explained by three environmental factors. For Q2 in the same model, Rsnp is similar to that of the genotypes only model since Q2 is not affected by three environmental factors. Thus, including the latter 3 might not help to reduce residual variance in the model.

The pattern is similar when we count TP and FP in the gene level as shown in Table 3.2. While the number of the detected true causal genes are similar or little less, the number of false positives is quite reduced. Comparison of ncTLF and ncTFGS reveals that the former is quite similar or slightly better.

Table 3.3 presents the prediction error (PE) calculated as $\text{PE}=\sum_{i=1}^{n}(Y_i-\sum_{j=1}^{p}X_{ij}\hat{\beta}_j)^2$ in the test set of Y. Ridge yields the inflated PE for Q1, but this case, overall, the PE is quite similar over the methods in any model.

Table 3.4 shows results the top 6 most frequently selected SNPs in the genotypes only model. As an example, for Q1, the causal SNP C13S523 in gene FLT1 is always selected over 200 replicates in all methods. Though the counts detected are little bit different, the SNP lists in Lasso, gflasso, and goscar are the same, and 5 SNPs out of top 9 lists are causal ones in Q1. Interestingly, the most often selected variants by ncTLF and ncTFGS are all causal ones, which means that these methods detect the right ones most often. For Q2, all methods select causal SNP C6S5380 most often. However, Lasso, gflasso, and goscar select non-causal SNPs also quite often. On the contrary, ncTLF and ncTFGS include all causal ones in their top 6 list.

For the combined model in Table 3.5, the pattern to detect the causal variants is similar to the one of the previous model. In Q1, age and smoking status are always detected over all methods. For Q2, non-causal covariate sex is detected in gflasso and goscar 121 times while Lasso, ncTLF, and ncTFGS do not include this in their top 7 lists.

In the gene level, among the top 5 gene lists of Q1 and Q2 as shown in Table

3.6 for the genotypes only model, three genes are causal for each method in Q1, but top 3 (FLT1, KDR, and ARNT) in ncTLF and ncTFGS are all causal ones while others also found non-causal gene KIF17 in all 200 replicates. For Q2, ncTLF and ncTFGS also identify the causal genes more often than others. For the combined model (Table 3.7), while the performances of the methods for Q1 are similar, ncTLF and ncTFGS detect the causal genes more often.

3.4 Discussion

In this study, we used the new FGSG package to apply some recently developed grouping-penalized regression methods, where we aimed to improve feature selection performance by combining the variants in the same gene accompanying variable selection to remove noise variants. Specifically we applied five different methods in FGSG; Lasso, gflasso, goscar, ncTLF and ncTFGS, to find the informative markers for two quantitative traits Q1 and Q2 in the GAW17 data. For each trait, we ran two different models, genotypes only model and combined model, where latter additionally included three environmental factors into the former.

To summarize results, the two non-convex methods ncTLF and ncTFGS detected the similar number of true causal SNPs in both the SNP and gene levels as the rest of the methods, but their false detection rates were lower. Additionally, ncTLF and ncTFGS included more true causal variants in the most frequently selected variant or gene lists than other methods. No method showed any advantage in its predictive ability for two traits.

This study suggested some improved feature selection performance with a reduced false positive rate by grouping the SNPs within genes with the newly developed penalty TLP. However, the improvement seemed to be limited, especially with only a similar or slightly smaller true-positive rate as compared to other methods. Therefore, more powerful study schemes to effectively explore associated genetic markers with a trait are desired.

Table 3.1: Mean[Median](sd) of TP and FP out of 200 replicates in the SNP level. The ratio of the mean[Median] TP over the mean[Median] FP in the SNP level is also in last column.

Model	Method	TPsnp	FPsnp	TPsnp/FPsnp
		Q1		
genotypes only	Lasso	5.5[6](1.0)	54.5[49.0](20.5)	0.10[0.12]
	gflasso	5.5[6](1.0)	53.3[50.0](19.1)	0.10[0.12]
	goscar	5.4[6](1.0)	53.2[50.0](18.9)	0.10[0.12]
	ncTLF	4.3[4](1.1)	16.8[9.0](17.4)	0.26[0.44]
	ncTFGS	4.4[4](1.1)	17.7[8.5](19.9)	0.25[0.47]
combined	Lasso	5.2[5](1.0)	38.2[36](10.4)	0.1[0.1]
	gflasso	5.1[5](1.0)	37.2[35](9.7)	0.1[0.1]
	goscar	5.1[5](1.0)	37.2[35](9.7)	0.1[0.1]
	ncTLF	4.1[4](1.1)	9.0[4](11.9)	0.5[1.0]
	ncTFGS	4.2[4](1.4)	9.3[4](15.2)	0.5[1.0]
		Q2		
genotypes only	Lasso	3.8[4](1.5)	44.9[42](14.1)	0.08[0.10]
	gflasso	3.2[3](2.3)	38.8[36.5](25.8)	0.08[0.08]
	goscar	2.6[2](1.7)	31.1[33](18.4)	0.08[0.06]
	ncTLF	3.2[3](5.0)	10.8[5](56.4)	0.30[0.60]
	ncTFGS	2.8[3](1.1)	8.0[5](10.9)	0.35[0.60]
combined	Lasso	3.7[4](1.3)	43.5[41.0](11.3)	0.1[0.1]
	gflasso	3.2[3](2.3)	38.3[36.5](26.0)	0.1[0.1]
	goscar	2.6[2](1.6)	30.2[33.0](18.3)	0.1[0.1]
	ncTLF	3.1[3](5.0)	10.2[4.0](56.3)	0.3[0.8]
	ncTFGS	2.7[3](1.0)	6.8[4.0](9.8)	0.4[0.8]

Table 3.2: Mean[Median](sd) of TP and FP out of 200 replicates in the gene level. The ratio of the mean[Median] TP over the mean[Median] FP in the gene level is also in last column.

Model	Method	TPgene	FPgene	TPgene/FPgene
		Q1		
genotypes only	Lasso	5.0[5](0.9)	31.6[30.0](8.1)	0.16[0.17]
	gflasso	5.0[5](0.9)	31.1[30.0](8.0)	0.16[0.17]
	goscar	5.0[5](0.9)	31.0[29.5](7.9)	0.16[0.17]
	ncTLF	3.0[3](1.4)	11.2[7.0](10.5)	0.27[0.43]
	ncTFGS	3.2[3](1.6)	13.6[9.5](15.9)	0.24[0.32]
combined	Lasso	4.5[4](1.0)	24.8[24](5.4)	0.2[0.2]
	gflasso	4.5[4](1.0)	24.3[24](5.3)	0.2[0.2]
	goscar	4.5[4](1.0)	24.3[24](5.3)	0.2[0.2]
	ncTLF	2.6[2](1.2)	6.4[4](7.7)	0.4[0.5]
	ncTFGS	2.8[3](1.2)	7.5[4](10.0)	0.4[0.8]
		Q2		
genotypes only	Lasso	6.9[7](1.8)	26.5[26](6.0)	0.26[0.27]
	gflasso	4.2[3](2.9)	21.6[22](9.4)	0.19[0.14]
	goscar	4.2[3](3.0)	22.3[23](11.2)	0.19[0.27]
	ncTLF	3.3[3](1.5)	5.8[4](8.1)	0.57[0.75]
	ncTFGS	3.7[3](2.0)	9.1[6](14.9)	0.41[0.50]
combined	Lasso	6.8[7.0](1.7)	25.9[25](5.1)	0.3[0.3]
	gflasso	4.1[3.5](2.9)	21.4[22](9.6)	0.2[0.2]
	goscar	4.2[3.0](2.9)	21.3[22](9.6)	0.2[0.1]
	ncTLF	3.2[3.0](1.4)	5.4[4](8.0)	0.6[0.8]
	ncTFGS	3.4[3.0](1.7)	7.1[4](12.1)	0.5[0.8]

Table 3.3: Mean[Median](sd) of PE out of 200 replicates.

Method	genotypes only	combined
	Q1	
Lasso	591.1[591.4](16.6)	548.0[549.0](21.8)
Ridge	646.6[646.3](7.1)	575.7[576.1](13.9)
gflasso	591.2[591.5](16.6)	547.9[549.0](21.7)
goscar	591.2[591.5](16.6)	547.9[549.0](21.7)
ncTLF	585.8[583.0](18.5)	549.3[551.3](24.5)
ncTFGS	586.1[583.6](18.6)	549.3[551.3](24.5)
	Q2	
Lasso	688.1[688.3](13.3)	688.9[689.2](13.3)
Ridge	686.4[686.2](5.1)	687.4[686.9](5.6)
gflasso	687.7[687.2](10.6)	688.9[689.1](10.9)
goscar	688.2[687.9](10.7)	689.3[689.3](10.8)
ncTLF	690.8[690.3](17.8)	690.9[690.5](16.6)
ncTFGS	690.3[689.8](16.7)	690.4[689.5](16.1)

Table 3.4: Feature selection on Q1 and Q2 in the SNP level for the genotypes only model.

Method	Q1				Q2			
	Gene	SNP	count	causal	Gene	SNP	count	causal
Lasso	FLT1	C13S523	200	Y	VNN1	C6S5380	200	Y
	FLT1	C13S522	199	Y	VNN3	C6S5441	195	Y
	KDR	C4S1878	192	Y	C10orf65	C10S4927	130	N
	FLT1	C13S431	191	Y	VNN3	C6S5427	124	N
	FLT1	C13S370	146	N	ZNF26	C12S7056	123	N
	KIF17	C1S1595	146	N	MFAP1	C15S1016	120	N
	FLT1	C13S370	146	N	LPL	C8S442	119	Y
	KIF17	C1S1595	146	N				
	ARNT	C1S6533	140	Y				
gflasso	FLT1	C13S523	200	Y	VNN1	C6S5380	188	Y
	FLT1	C13S522	199	Y	PHYHIPL	C10S2973	121	N
	KDR	C4S1878	192	Y	CHST10	C2S2053	110	N
	FLT1	C13S431	190	Y	PGBD3	C10S2617	110	N
	FLT1	C13S370	147	N	CHST10	C2S2053	110	N
	KIF17	C1S1595	147	N	PGBD3	C10S2617	110	N
	FLT1	C13S370	147	N	ZHX2	C8S3863	109	N
	KIF17	C1S1595	147	N	POLR2J2	C7S2544	105	N
	ARNT	C1S6533	142	Y	PGBD3	C10S2632	102	N
goscar	FLT1	C13S523	200	Y	VNN1	C6S5380	184	Y
	FLT1	C13S522	199	Y	PHYHIPL	C10S2973	122	N
	KDR	C4S1878	192	Y	CHST10	C2S2053	114	N
	FLT1	C13S431	190	Y	ZHX2	C8S3863	108	N
	FLT1	C13S370	147	N	POLR2J2	C7S2544	105	N
	KIF17	C1S1595	147	N	ZNF117	C7S1632	103	N
	FLT1	C13S370	147	N				
	KIF17	C1S1595	147	N				
	ARNT	C1S6533	141	Y				
ncTLF	FLT1	C13S523	200	Y	VNN1	C6S5380	189	Y
	FLT1	C13S522	197	Y	VNN3	C6S5441	156	Y
	FLT1	C13S431	179	Y	LPL	C8S442	73	Y
	KDR	C4S1878	126	Y	VNN3	C6S5449	65	Y
	ARNT	C1S6533	92	Y	GCKR	C2S354	38	Y
	KDR	C4S1884	66	Y	PDGFD	C11S5292	36	Y
ncTFGS	FLT1	C13S523	200	Y	VNN1	C6S5380	187	Y
	FLT1	C13S522	198	Y	VNN3	C6S5441	149	Y
	FLT1	C13S431	178	Y	LPL	C8S442	70	Y
	KDR	C4S1878	124	Y	VNN3	C6S5449	64	Y
	ARNT	C1S6533	93	Y	GCKR	C2S354	43	Y
	KDR	C4S1884	71	Y	PDGFD	C11S5292	32	Y

Table 3.5: Feature selection on Q1 and Q2 in the SNP level for the combined model.

Method	Q1				Q2			
	Gene	SNP	count	causal	Gene	SNP	count	causal
Lasso	FLT1	C13S523	200	Y	VNN1	C6S5380	200	Y
	AGE	AGE	200	Y	VNN3	C6S5441	195	Y
	SMOKING	SMOKING	200	Y	C10orf65	C10S4927	134	N
	FLT1	C13S522	199	Y	VNN3	C6S5427	122	N
	KDR	C4S1878	192	Y	ZNF26	C12S7056	121	N
	FLT1	C13S431	183	Y	MFAP1	C15S1016	119	N
	KDR	C4S1884	131	Y	LPL	C8S442	118	Y
	TRIM16L	C17S1071	127	N	PGBD3	C10S2632	118	N
	EGFR	C7S1339	124	N				
gflasso	FLT1	C13S523	200	Y	VNN1	C6S5380	187	Y
	AGE	AGE	200	Y	PHYHIPL	C10S2973	121	N
	SMOKING	SMOKING	200	Y	SEX	SEX	118	Y
	FLT1	C13S522	199	Y	CHST10	C2S2053	110	N
	KDR	C4S1878	192	Y	ZHX2	C8S3863	102	N
	FLT1	C13S431	183	Y	GCKR	C2S354	101	Y
	KDR	C4S1884	131	Y	POLR2J2	C7S2544	101	N
	TRIM16L	C17S1071	127	N	HLA-E	C6S1760	101	N
	EGFR	C7S1339	125	N	TSPAN13	C7S436	100	N
goscar	FLT1	C13S523	200	Y	VNN1	C6S5380	184	Y
	AGE	AGE	200	Y	SEX	SEX	121	Y
	SMOKING	SMOKING	200	Y	PHYHIPL	C10S2973	120	N
	FLT1	C13S522	199	Y	CHST10	C2S2053	109	N
	KDR	C4S1878	192	Y	ZHX2	C8S3863	104	N
	FLT1	C13S431	183	Y	POLR2J2	C7S2544	102	N
	KDR	C4S1884	131	Y	TSPAN13	C7S436	100	N
	TRIM16L	C17S1071	127	N				
	EGFR	C7S1339	125	N				
ncTLF	FLT1	C13S523	200	Y	VNN1	C6S5380	188	Y
	AGE	AGE	200	Y	VNN3	C6S5441	155	Y
	SMOKING	SMOKING	200	Y	LPL	C8S442	69	Y
	FLT1	C13S522	194	Y	VNN3	C6S5449	62	Y
	FLT1	C13S431	153	Y	PGBD3	C10S2617	36	N
	KDR	C4S1878	148	Y	GCKR	C2S354	35	Y
	KDR	C4S1884	66	Y	PDGFD	C11S5292	34	Y
	ARNT	C1S6533	48	Y				
	EGFR	C7S1339	42	N				
ncTFGS	FLT1	C13S523	200	Y	VNN1	C6S5380	188	Y
	AGE	AGE	200	Y	VNN3	C6S5441	148	Y
	SMOKING	SMOKING	200	Y	LPL	C8S442	65	Y
	FLT1	C13S522	195	Y	VNN3	C6S5449	59	Y
	FLT1	C13S431	154	Y	GCKR	C2S354	38	Y
	KDR	C4S1878	151	Y	PDGFD	C11S5292	31	Y
	KDR	C4S1884	65	Y	PGBD3	C10S2617	30	N
	ARNT	C1S6533	46	Y				
	EGFR	C7S1339	43	N				

Table 3.6: Feature selection on Q1 and Q2 in the gene level for the genotypes only model.

Method	Q1			Q2		
	Gene	count	causal	Gene	count	causal
Lasso	FLT1	200	Y	VNN1	200	Y
	KIF17	200	N	VNN3	199	Y
	KDR	196	Y	LY75	199	N
	APOBEC3F	196	N	GDF15	193	N
	BRWD1	194	N	PGBD3	188	N
	EGFR	192	N	EGFR	183	N
	HIF3A	190	Y			
gflasso	FLT1	200	Y	VNN1	188	Y
	KIF17	200	N	PGBD3	156	N
	KDR	196	Y	TSPAN13	130	N
	APOBEC3F	196	N	HLA-E	129	N
	BRWD1	194	N	PHYHIPL	121	N
	EGFR	191	N			
	HIF3A	188	Y			
goscar	FLT1	200	Y	VNN1	184	Y
	KIF17	200	N	PGBD3	158	N
	KDR	196	Y	TSPAN13	138	N
	APOBEC3F	196	N	GDF15	124	N
	BRWD1	194	N	HLA-E	123	N
	EGFR	190	N			
	HIF3A	188	Y			
ncTLF	FLT1	200	Y	VNN1	189	Y
	KDR	139	Y	VNN3	171	Y
	ARNT	112	Y	LPL	87	Y
	KIF17	107	N	LY75	63	N
	APOBEC3F	102	N	PDGFD	60	Y
ncTFGS	FLT1	200	Y	VNN1	187	Y
	KDR	140	Y	VNN3	165	Y
	ARNT	112	Y	LPL	82	Y
	KIF17	103	N	LY75	62	N
	APOBEC3F	101	N	PGBD3	56	N

Table 3.7: Feature selection on Q1 and Q2 in the gene level for the combined model.

Method	Q1			Q2		
	Gene	count	causal	Gene	count	causal
Lasso	FLT1	200	Y	VNN1	200	Y
	KDR	197	Y	VNN3	199	Y
	HIF3A	179	Y	LY75	199	N
	ARNT	142	Y	GDF15	195	N
	ELAVL4	137	Y	PGBD3	187	N
				EGFR	182	N
gflasso	FLT1	200	Y	VNN1	187	Y
	KDR	197	Y	PGBD3	148	N
	HIF3A	178	Y	TSPAN13	127	N
	ARNT	138	Y	HLA-E	125	N
	ELAVL4	135	Y	PHYHIPL	121	N
goscar	FLT1	200	Y	VNN1	184	Y
	KDR	197	Y	PGBD3	146	N
	HIF3A	178	Y	TSPAN13	133	N
	ARNT	138	Y	HLA-E	123	N
	ELAVL4	135	Y	PHYHIPL	120	N
ncTLF	FLT1	200	Y	VNN1	188	Y
	KDR	163	Y	VNN3	168	Y
	ARNT	65	Y	LPL	85	Y
	HIF3A	50	Y	PDGFD	61	Y
	ELAVL4	42	Y	PGBD3	56	N
ncTFGS	FLT1	200	Y	VNN1	188	Y
	KDR	165	Y	VNN3	161	Y
	ARNT	63	Y	LPL	79	Y
	HIF3A	51	Y	LY75	57	N
	ELAVL4	40	Y	PDGFD	55	Y

3.5 Programs

```
[MATLAB]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Following code produces summary of
% 1.TP and FP in snp/gene level,
% 2.ME and PE,
% 3.bhat
% for ncTLF of first Q2 replicate
%
% Input
% 1.netwk.txt
% : pool of G (=# of genes) pairwise
%   matrices of size
% 2.b.txt
% : true coefficient vector of size
%   p x 1.
% 3.Gene.txt
% : a vector of size p x 1
%   where each component indicates
%   the gene that the snp belongs to,
%   so max(Gene) = G
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Rep=200;
%cutoff in TP and FP
epsilon=0.0001;

%data
netwk0 = dlmread('netwk.txt', ' ');
netwk=netwk0';

b=dlmread('b.txt', ' ');
tY = dlmread('Q2.txt', ' ');
X0 = dlmread('X.txt', ' ');
[n,p]=size(X0);
X=X0-ones(n,1)*mean(X0);

Gene=dlmread('Gene.txt', ' ');
indices = [Gene ones(size(Gene))];

d=1;
q1=find(abs(b)>0);
q0=setxor(q1,1:p);

Y0=tY((1+n*(d-1)):(n*d),1);
 Y=Y0-mean(Y0);
Ytu0=tY((1+n*(d)):(n*(d+1)),1);
 Ytu=Ytu0-mean(Y0);
Yte0=tY((1+n*(d+1)):(n*(d+2)),1);
 Yte=Yte0-mean(Y0);

 gd=5;
 lamf=linspace(0.001,10,gd);
 lams=linspace(0.001,10,gd);
 ltau=linspace(0.01,max(abs(blasso))*0.5,gd);

%% ncTLF
 tunM_TLF=zeros(gd,gd*gd);
 opts=[];
 for s=1:gd
  for t=1:gd
   for u=1:gd
    opts.x0=blasso;
    tau=ltau(u);opts.tau=tau;
    x1= ncTLF(X,Y,netwk,lamf(s),lams(t),opts);
    tunM_TLF(s,(t-1)*gd+u)=norm(Ytu-X*x1)^2;
    end
  end
 end

 tmp=tunM_TLF>0;
 [row,col]=find(tunM_TLF==min(tunM_TLF(tmp)));
 %in case of dup
 row=row(1);col=col(1);
  lam1=mean(lamf(row));
  if   rem(col,gd)==0, lam2=mean(lams(floor(col/gd)));
       tau=mean(ltau(gd));
  else lam2=mean(lams(floor(col/gd)+1));
       tau=mean(ltau(col-floor(col/gd)*gd));
  end
  opts.x0=blasso;
  opts.tau=tau;
  x1= ncTLF(X,Y,netwk,lam1,lam2,opts);
  bTLF=x1;

%record result
  cgene=find(accumarray(indices, abs(b),
             [numel(unique(Gene)) 1], @sum));
  ncgene=setxor(cgene,1:max(Gene));

%TPFP in snp level
TPFPsnp=[sum(abs(bTLF(q1))>epsilon)
         sum(abs(bTLF(q0))>epsilon)];

%TPFP in gene level
TPFPgene=[
length(intersect(cgene,
      find(accumarray(indices,abs(bTLF),
      [numel(unique(Gene)) 1],@sum))))
length(intersect(ncgene,
```

```
      find(accumarray(indices,abs(bTLF),
      [numel(unique(Gene)) 1],@sum))))
];

%ME
 ME=dot(X*(bTLF-b),X*(bTLF-b));
%PE
 PE=dot(Yte-X*bTLF,Yte-X*bTLF);

%bhat
 bhat=bTLF;

%save
dlmwrite(['TPFPsnp' num2str(d) '.txt'], TPFPsnp);
dlmwrite(['TPFPgene' num2str(d) '.txt'], TPFPgene);
dlmwrite(['ME' num2str(d) '.txt'], ME);
dlmwrite(['PE' num2str(d) '.txt'], PE);
dlmwrite(['bhat' num2str(d) '.txt'], bhat);
```

Chapter 4

A network-based penalized regression method with application to genomic data

4.1 Introduction

With large amounts of high-dimensional data accumulating from high-throughput genomic studies, penalized regression methods equipped with simultaneous variable selection and parameter estimation have been increasingly used in practice. Most popular generic methods exploiting sparsity of high-dimensional data include the Lasso [3], SCAD [39], elastic net (Enet) [9] and LARS [4], among others. In addition to sparsity, other structures may be present in a given high-dimensional problem. For example, in genomics, various types of gene networks describe gene-gene interactions and their coordinated functioning: protein-protein interaction (PPI) networks as available from the Biomolecular Interaction Network Database(BIND) [11] and the Human Protein Reference Database (HPRD) [12], biological pathways in the Kyoto Encyclopedia of Genes and Genomes (KEGG) [13], and gene functional annotations in the Gene Ontology (Ashburner *et al.* [14]). Regardless of the network type, often it is reasonable to assume that two neighboring genes in a network are more likely to participate together in the same biological process than two genes far away in the network. Hence, incorporating prior biological knowledge by exploiting the network structure in a statistical method is expected to improve its performance. A natural way to utilize gene network information is to smooth parameters of neighboring genes over a network: given any two neighboring genes in a network, denoted as $j \sim j'$, and their parameters β_j and $\beta_{j'}$, it may be reasonable to assume that $\beta_j/w_j \approx \beta_{j'}/w_{j'}$ with some known or chosen weights w_j and $w_{j'}$ [40]. More generally, since the effect directions could be different, e.g. regulation of gene expression could be either stimulatory or inhibitory, we can assume $|\beta_j|/w_j \approx |\beta_{j'}|/w_{j'}$ [41, 17]. Although the aforementioned assumptions are reasonable, they may be too strong in some cases: in general, it is valid to assume two neighboring genes in a network to be co-functioning, but their effect sizes may or may not be equal. Hence, rather than smoothing the (weighted) parameters over a network, we only assume that two neighboring genes are more likely to participate together in the same biological process than two non-neighboring genes. This latter prior knowledge was recently used by [42] in a modified forward stepwise variable selection scheme. In this paper, we propose a novel penalty to incorporate this prior in a general framework of penalized regression. Simply speaking, we propose

a penalty to encourage $I(|\beta_j| \neq 0) = I(|\beta_{j'}| \neq 0)$ for $j \sim j'$. Since the indicator function, like the L_0-loss function, is not even continuous, it is not computationally feasible to use it directly in an objective function to be minimized. Our major contributions include proposing a novel penalty as its approximation (or surrogate) and a corresponding non-convex minimization method.

This paper is organized as follows. Section 2 first briefly reviews some existing methods, then describes two implementations of our new method in detail. In section 3 simulation results are presented to investigate the finite sample performance of the methods, demonstrating the advantages of the proposed methods over several existing methods. Section 4 illustrates the application of the methods to predict metastases of breast cancer patients with their gene expression profiles and a PPI network. We end with a short summary and discussion outlining a few future topics.

4.2 Methods

4.2.1 Review: penalized regression

In a linear regression model,

$$Y_i = \beta_0 + \sum_{j=1}^{p} X_{ij}\beta_j + \epsilon_i, \qquad E(\epsilon_i) = 0, Var(\epsilon_i) = \sigma^2, i = 1, ..., n \tag{4.1}$$

for $i = 1, ..., n$, we often have a "large p, small n" problem, as arising in high-throughput genomic studies. With a large p, e.g. $p > n$, the ordinary least squares estimate (OLSE) does not perform well due to its over-fitting. As one remedy, penalized regression is proposed: a penalty $p(\beta)$ is added to the objective function

$$S(\beta) = \frac{1}{2}\|Y - X\beta\|^2 + p(\beta). \tag{4.2}$$

The penalty $p(\beta)$ not only regularizes parameter estimation as desired, but also can realize effective variable selection. The Lasso [3] with an L_1-penalty is well-known: $p(\beta) = \lambda \sum_{j=1}^{p} |\beta_j|$, where λ is a tuning parameter to be determined. With a large λ, the Lasso yields a sparse (i.e. few non-zero components of) estimate of β, effectively realizing variable selection. However, the Lasso and many other generic penalized

methods ignore network structures in the predictors, hence may not be efficient. To take advantage of given information embedded in a predictor network, [40], [41] and [17] introduced network-based penalized regression methods. We implicitly assume that a network is given, and as before two directly connected nodes/genes (i.e. with an edge connecting them) are represented as $j \sim j'$. The first is a graph constrained estimation (Grace) method [40] with penalty

$$p(\beta) = \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j \sim j'} \big(\frac{\beta_j}{\sqrt{d_j}} - \frac{\beta_{j'}}{\sqrt{d_{j'}}}\big)^2,$$

where d_j is the degree of node j, i.e. the number of edges connected to j. The first term is an L_1-penalty for variable selection, while the second aims to smooth (weighted) β_j's over the network. As discussed before, since in some applications two neighboring genes might have β_j's with opposite signs, it is more desirable to shrink (weighted) $|\beta_j|$'s towards each other in a network: we'd like to encourage $|\beta_j|/\sqrt{d_j} = |\beta_{j'}|\sqrt{d_{j'}}$ for $j \sim j'$. For this purpose, an adaptive version (aGrace) was proposed [41]:

$$p(\beta) = \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j \sim j'} \big(\frac{\text{sign}(\tilde{\beta}_j)\beta_j}{\sqrt{d_j}} - \frac{\text{sign}(\tilde{\beta}_{j'})\beta_{j'}}{\sqrt{d_{j'}}}\big)^2,$$

where $\tilde{\beta}_j$ is an initial estimate based on OLSE for $p < n$, or an elastic net (Enet) estimate [9] for $p \geq n$. The main idea is to use $\text{sign}(\tilde{\beta}_j)$ to estimate $\text{sign}(\beta_j)$, which however may not work well for high-dimensional data: since we do not even know which β_j's are 0 for variable selection, it is more difficult to estimate their signs. As an alternative, Pan *et al.* (2010) proposed a direct approach with a class of penalties

$$p(\beta) = \lambda \sum_{j \sim j'} \left[\left(\frac{|\beta_j|}{\sqrt{d_j}} \right)^{\gamma} + \left(\frac{|\beta_{j'}|}{\sqrt{d_{j'}}} \right)^{\gamma} \right]^{1/\gamma},$$

with a $\gamma > 1$ to be specified. This class of penalties are essentially a weighted L_γ-norm with some attractive properties: for $j \sim j'$, in addition to the *grouping* effect of shrinking weighted $|\beta_j|$ and $|\beta_{j'}|$ towards each other, it also realizes *group* variable selection that encourages both β_j and $\beta_{j'}$ to be zero simultaneously [7], [43].

[17] demonstrated better performance of the method for variable selection than Lasso, Enet and Grace, though the parameter estimates may be severely biased. [44] proposed a 2-step procedure similar to that of [41] for bias reduction; with a new convex programming method, they also showed that the penalty with $\gamma = \infty$ performed better than that with smaller γ=2 or 8. The penalty with $\gamma = \infty$ is linear:

$$p(\beta) = \lambda \sum_{j \sim j'} \max \left(\frac{|\beta_j|}{\sqrt{d_j}}, \frac{|\beta_{j'}|}{\sqrt{d_{j'}}} \right),$$

closely related to a penalty proposed by [16], though a separate L_1-penalty is added in the latter for variable selection. Hence, in the following we consider only $\gamma = \infty$, and simply denote the method with an L_∞-norm penalty as L_∞, while the two-step procedure as aL_∞. Finally we note that in the above methods, we can replace $\sqrt{d_j}$ with a more general weight w_j, which for example can be simply 1.

Although these methods appear to be useful, their assumption on the smoothness of (weighted) β_j's or $|\beta_j|$'s over a network may be questionable in some applications. Therefore, next we propose a new network-based penalty with a much less stringent assumption.

4.2.2 New methods

Our new methods are based on the below "ideal" penalty:

$$p(\beta) = \lambda_1 \sum_{j=1}^{p} I\left(|\beta_j| \neq 0\right) + \lambda_2 \sum_{j \sim j'} \left| I\left(\frac{|\beta_j|}{w_j} \neq 0 \right) - I\left(\frac{|\beta_{j'}|}{w_{j'}} \neq 0 \right) \right|, \tag{4.3}$$

where the first penalty is the L_0-loss for sparsest variable selection and unbiased parameter estimation [10], while the second one encourages simultaneous selection (or elimination) of two neighboring nodes in a network. Since the indicator function $I()$ is not continuous, it is not computationally tractable. As a computational surrogate of $I(|z| \neq 0)$, [10] proposed a truncated Lasso penalty (TLP), $J_\tau(|z|) = \min(\frac{|z|}{\tau}, 1)$, which tends to $I(|z| \neq 0)$ as $\tau \to 0^+$; the tuning parameter τ determines the degree of approximation. Thus, applying the TLP to (4.3) leads to a new penalty with a TLP for variable selection and a TLP-based penalty for grouping of

indicators, shortened as $TTLP_I$:

$$p(\beta) = \lambda_1 \sum_{j=1}^{p} J_\tau(|\beta_j|) + \lambda_2 \sum_{j \sim j'} \left| J_\tau\left(\frac{|\beta_j|}{w_j}\right) - J_\tau\left(\frac{|\beta_{j'}|}{w_{j'}}\right) \right|, \tag{4.4}$$

where a common τ is used in both terms for variable selection and grouping. Note that, although the weights w_j can be omitted in (4.3), they may play an important role in (4.4) (and other penalties shown earlier), as to be shown later.

For any given $(\lambda_1, \lambda_2, \tau)$, we present a computational algorithm to minimize $S(\beta)$ with the new penalty. First, we decompose the non-convex function $J_\tau(|z|)$ in (4.4) into a difference of two convex functions: $J_\tau(|z|) = \frac{1}{\tau}(|z| - \max(|z| - \tau, 0))$. Additionally, to deal with the absolute value function in the second term of (4.4), we construct another difference of convex (DC) decomposition: $|u - v| = 2\max(u, v) - (u + v)$, where u and v are both convex. Therefore, after applying these two DC decompositions to (4.4), we have

$$\frac{\lambda_1}{\tau}(\sum_{j=1}^{p} |\beta_j| - \max(|\beta_j| - \tau, 0)) + \frac{\lambda_2}{\tau} \sum_{j' \sim j} 2\max(u, v) - (u + v),$$

where $u = \frac{|\beta_j|}{w_j} + \max(\frac{|\beta_{j'}|}{w_{j'}} - \tau, 0)$ and $v = \frac{|\beta_{j'}|}{w_{j'}} + \max(\frac{|\beta_j|}{w_j} - \tau, 0)$. Then, (4.4) can be rewritten as a difference of two convex functions p_1 and p_2,

$$\begin{aligned}
&p(\beta) = p_1(\beta) - p_2(\beta), \\
&p_1(\beta) = \frac{1}{\tau}\left(\lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j' \sim j} 2\max(u, v)\right), \\
&p_2(\beta) = \frac{1}{\tau}\left(\lambda_1 \sum_{j=1}^{p} \max(|\beta_j| - \tau, 0) + \lambda_2 \sum_{j' \sim j} (u + v)\right).
\end{aligned}$$

Linearizing p_2 at a current estimate $\hat{\beta}^{(m-1)}$ and ignoring terms independent of β,

we obtain a convex approximation of $S(\beta)$:

$$S^{(m)}(\beta) = \frac{1}{2}\|Y - X\beta\|^2 + \frac{\lambda_1}{\tau}\sum_{j=1}^{p}|\beta_j|I(|\hat{\beta}_j^{(m-1)}| \leq \tau) + \frac{\lambda_2}{\tau}\sum_{j\sim j'} 2\max(u,v)$$
$$-\frac{\lambda_2}{\tau}\sum_{j\sim j'}\left(\frac{\beta_j}{w_j}Sign(\hat{\beta_j}^{(m-1)})[1 + I(\frac{|\hat{\beta}_j^{(m-1)}|}{w_j} > \tau)] + \frac{\beta_{j'}}{w_{j'}}Sign(\hat{\beta_{j'}}^{(m-1)})[1 + I(\frac{|\hat{\beta_{j'}}^{(m-1)}|}{w_{j'}} > \tau)]\right),$$

which is minimized to obtain an updated estimate $\hat{\beta}^{(m)}$. Since $S^{(m)}(\beta)$ is convex, we use MATLAB package `CVX` [45] to minimize it. The algorithm to compute the final estimate $\hat{\beta}$ is as follows.

[A1] Start with an initial estimate $\hat{\beta}^{(0)}$ and $m = 1$.

[A2] At iteration m, compute $\hat{\beta}^{(m)}$ that minimizes $S^{(m)}(\beta)$.

[A3] Stop if $S(\hat{\beta}^{(m-1)})$-$S(\hat{\beta}^{(m)}) \leq \epsilon$ with a small tolerance ϵ (e.g. 10^{-4} used throughout); otherwise, return to [A2].

We used the Lasso estimate $\hat{\beta}_{lasso}$ as the initial value $\hat{\beta}^{(0)}$ in step [A1]. The three tuning parameters $(\delta_1, \delta_2, \tau)$ with $\delta_1 \equiv \lambda_1/\tau$ and $\delta_2 \equiv \lambda_2/\tau$ were searched over a set of 4, 4 and 5 equally spaced grid points respectively within the following ranges: let t denote the maximum absolute value of the components of the lasso estimate $\hat{\beta}_{lasso}$, and g denote the total number of the edges in the network, we used intervals $[t, \frac{pt}{4}]$ for δ_1, $[t, tg]$ for δ_2, and $[10^{-6}, \frac{t}{2}]$ for τ.

The TLP has been shown to perform well for accurate variable selection and almost unbiased parameter estimation for sparse models [10]. An intuition behind the TLP is that, if a parameter β_j is large with $\beta_j > \tau$, then no penalty is imposed on β_j, which is in contrast to universal penalization of Lasso on all β_j's that leads to Lasso's biased parameter estimation and over selection of too large models. However, with a non-sparse true model, universal penalization imposed by Lasso may be beneficial to parameter estimation and outcome prediction due to its better bias-variance trade-off. In our current context, since the true model may not be too sparse, it might be interesting to contrast the performance of the TLP and Lasso. Furthermore, to save computing time, we have used a common τ for both variable selection and grouping in (4.4), which might not be optimal. Thus, rather than using

the TLP for variable selection in (4.4), we can simply use the Lasso, leading to a modification with a Lasso penalty for variable selection and a TLP-based penalty for grouping indicators, called $LTLP_I$:

$$p(\beta) = \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j \sim j'} \left| J_\tau \left(\frac{|\beta_j|}{w_j} \right) - J_\tau \left(\frac{|\beta_{j'}|}{w_{j'}} \right) \right|. \tag{4.5}$$

The computational algorithm is similar to that for $TTLP_I$. In particular, the intermediate objective function is

$$S^{(m)}(\beta) = \frac{1}{2}\|Y - X\beta\|^2 + \frac{\lambda_1}{\tau} \sum_{j=1}^{p} |\beta_j| + \frac{\lambda_2}{\tau} \sum_{j \sim j'} 2\max(u, v)$$
$$-\frac{\lambda_2}{\tau} \sum_{j \sim j'} \left(\frac{\beta_j}{w_j} Sign(\hat{\beta_j}^{(m-1)})[1 + I(\frac{|\hat{\beta}_j^{(m-1)}|}{w_j} > \tau)] + \frac{\beta_{j'}}{w_{j'}} Sign(\hat{\beta_{j'}}^{(m-1)})[1 + I(\frac{|\hat{\beta_{j'}}^{(m-1)}|}{w_{j'}} > \tau)] \right),$$

The tuning parameters $(\lambda_1, \lambda_2, \tau)$ were tuned in the same way as in $TTLP_I$, except that the searching range of λ_1 was set as interval $[\hat{\lambda}_{lasso}/1.5, 1.5\hat{\lambda}_{lasso}]$, where $\hat{\lambda}_{lasso}$ was the chosen tuning parameter for the Lasso.

4.3 Simulations

4.3.1 Simulation set-ups

Our simulation set-ups are similar to those in [40] and [17]. Briefly, the responses Y were generated from linear model (4.1) with iid error $\epsilon_i \sim N(0, \sum_j \beta_j^2/2)$. A gene regulatory network consisted of 10 independent subnetworks, each including one transcription factor (TF) and its 10 target genes (and thus $p = 110$); each TF was connected to each of its 10 target genes while there was no edge between any other two genes. All predictors were marginally distributed as $N(0, 1)$; conditional on the TF's expression level X_{TF}, a target gene's expression level X_{tg} was distributed as $N(0.5X_{TF}, 0.75)$; any two X_{tg}s were conditionally independent given X_{TF}. The expression levels of any two genes from two different subnetworks were independent. Two types of the true regression coefficient vector β were considered in two sets of simulations I and II respectively: in simulation I, the (weighted) magnitudes of the

β_j's were close to each other, while in simulation II they were completely random. Specifically, in set-up 1 of the simulation I, we have

$$\beta = (5, \underbrace{\frac{5}{\sqrt{10}}, ..., \frac{5}{\sqrt{10}}}_{10}, -3, \underbrace{-\frac{3}{\sqrt{10}}, ..., -\frac{3}{\sqrt{10}}}_{10}, 0, ..., 0)',$$

the first 11 were for the TF and its 10 targets in subnetwork 1, followed by subnetworks 2 to 10. Note that there were $p_1 = 22$ informative predictors with $\beta_j \neq 0$, and there was a strong relationship among β_j's: $\beta_j/\sqrt{d_j} = \beta_{j'}/\sqrt{d_{j'}}$ for any $j \sim j'$. The set-up 2 was similar to set-up 1 except that the signs of the first three target genes' β_j's in the first two subnetworks were flipped; that is, for $j \sim j'$, $\beta_j/\sqrt{d_j} = \beta_{j'}/\sqrt{d_{j'}}$ might not hold, though $|\beta_j|/\sqrt{d_j} = |\beta_{j'}|/\sqrt{d_{j'}}$ always held. Similarly set-up 3 was another type of perturbation to set-up 1: the first 5 genes' β_j's were set to 0 in the first two subnetworks; $|\beta_j|/\sqrt{d_j} = |\beta_{j'}|/\sqrt{d_{j'}}$ held for only some, but not all, gene pairs $j \sim j'$.

In simulation II, we had

$$\beta = (1.5, \underbrace{\beta_2, ..., \beta_{11}}_{10}, 0.5, \underbrace{\beta_{13}, ..., \beta_{22}}_{10}, 0, ..., 0)',$$

where we randomly drew $\beta_2, ..., \beta_{11} \sim$ Unif(0, 3), and $\beta_{13}, ..., \beta_{22} \sim$ Unif(-3, 3); then we flipped the signs of $\beta_7, ..., \beta_{11}$. Specifically the generated regression coefficient vector β was:

$$\beta = (1.5, \underbrace{2.98, 2.01, 1.35, 1.03, 2.7, -0.98, -2.39, -1.33, -0.37, -1.24}_{10},$$
$$0.5, \underbrace{1.85, 2.91, 2.48, 1.45, 2.25, 0.34, -1.12, -1.03, -0.2, 0.7}_{10}, 0..., 0)'.$$

We generated 100 replicates for each set-up, where each replicate consisted of a training set, a tuning set, both of size n=50, and a test set of size m=200. The training set was used to fit the model to obtain parameter estimates $\hat{\beta}$ for any given tuning parameter values. The tuning set was used to select the tuning parameters as the ones with the smallest predictive residual sum of squares (PRSS) for $\hat{\beta}$ on the tuning data. To evaluate the performance, the model error (ME) and the prediction

error (PE) were calculated: ME=$(\beta - \hat{\beta})' E(X'X)(\beta - \hat{\beta})/n$ where $E(X'X)$ is the population covariance matrix of X since $E(X) = 0$, and PE=$\sum_{i=1}^{m}(Y_i - \hat{Y}_i)^2/m$ (based on the test data) . We also calculated the mean and median numbers of the true positives (TPs) and false positives (FPs) for variable selection, where $|\hat{\beta}_j| > 10^{-3}$ was considered as non-zero (or a positive). We considered two types of weights, $w_j = 1$ and $w_j = \sqrt{d_j}$.

We also note that, following [41], unlike in [40] and [17], we did not rescale the estimates of Grace and Enet; we found that the un-scaled versions here performed either better than or almost the same as the rescaled ones.

4.3.2 Simulation results

Simulation results for simulation I are summarized in Table 4.1. The weights $w_j = 1$ were mis-specified in the sense of having $|\beta_j|/w_j \neq |\beta_{j'}|/w_{j'}$ for any two non-null neighboring genes $j \sim j'$ with non-zero β_j and $\beta_{j'}$, whereas the weights $w_j = \sqrt{d_j}$ were correctly specified in the above sense for set-ups 1 and 2, but only partially correctly specified for set-up 3, for the methods depending on the assumptions on the magnitudes of the regression coefficients, i.e. Grace, aGrace, L_∞ and aL_∞. For set-up 1, in which the true regression coefficients of all the directly connected genes in a network had the same signs, with w_j=1, Grace yielded the lowest mean ME and PE, followed closely by $LTLP_I$ and aL_∞. All the network-based methods except L_∞ performed better than the generic Lasso and Enet for parameter estimation and outcome prediction. For variable selection, however, Grace performed poorly with a too large mean number of FPs; L_∞, aL_∞, $TTLP_I$ and $LTLP_I$ had a comparable large number of TPs but a much smaller number of FPs. With weights $w_j = \sqrt{d_j}$, the aL_∞ had the smallest mean ME and PE, closely followed by Grace, then by $LTLP_I$. For variable selection, perhaps due to the group selection, L_∞ and aL_∞ gave the most sparse models with the highest number of TPs, then followed by $TLPI_I$ and $LTLP_I$.

In set-up 2 some neighboring genes had true regression coefficients with opposite signs. As expected, Grace was no longer the winner, and aGrace slightly improved over Grace with a smaller mean ME and PE. With the mis-specified weights $w_j = 1$, $LTLP_I$ gave the smallest mean ME and PE, then followed by aGrace and aL_∞. For

variable selection, $TTLP_I$ was the winner, giving a similarly large number of TPs but fewer FPs than $LTLP_I$; other methods all yielded much smaller numbers of TPs. On the other hand, with the correctly specified weights $w_j = \sqrt{d_j}$, aL_∞ was the winner with the smallest ME and PE, followed by $LTLP_I$ and then aGrace. For variable selection, aL_∞ and L_∞ seemed to be the winners, though $TTLP_I$ was also quite competitive; again $LTLP_I$ gave less sparse models than $TTLP_I$, both performing better than Grace and aGrace.

For set-up 3, with the mis-specified weights $w_j = 1$, $LTLP_I$ performed best with the smallest mean ME and PE, while all other network-based methods gave larger MEs than the generic Lasso and Enet, though $TTLP_I$ and Grace gave smaller mean PEs than those of Lasso and Enet. On the other hand, with the weights $w_j = \sqrt{d_j}$, Grace had the smallest ME, closely followed by $LTLP_I$ and aL_∞.

With random regression coefficients in simulation II (Table 4.2), our new methods showed more substantial advantages over other methods, in terms of both parameter estimation (and outcome prediction) and variable selection.

In summary, in cases that the regression coefficients of neighboring nodes had the same signs, i.e. effect directions, with correctly specified weights Grace performed well in parameter estimation and outcome prediction, otherwise it did not perform well; in both cases, however, it gave too large models. Its modification aGrace could slightly improve over Grace in the case that neighboring nodes had different association directions with the outcome. As expected, L_∞ and aL_∞ were not sensitive to different association directions of neighboring nodes, but they did not perform well if the weights were mis-specified; otherwise they performed best in variable selection, possibly due to their mechanisms of group variable selection. As discussed by [44], due to the over-shrinkage and thus large biases of its parameter estimates (Table 4.3), L_∞ did not perform well in parameter estimation and prediction. On the other hand, our proposed methods, especially $LTLP_I$, seemed to perform reasonable well across all the scenarios; $TTLP_I$ seemed to have some edge over $LTLP_I$ with a comparable number of TPs but fewer FPs, though the former lost its edge to the latter in parameter estimation and outcome prediction, perhaps due to the larger variability of the former's parameter estimates in the not-so-sparse true models considered here (Table 4.3); for more sparse true models, we did observe

that $TTLP_I$ performed better than $LTLP_I$ for both parameter estimation and variable selection (results not shown). Overall, our methods gave less biased estimates than other network-based methods (Table 4.3). We conclude that our proposed new methods were more robust to mis-specified weights or mis-specified relationships among the true regression coefficients than other network-based methods.

4.4 Example

We applied the methods to a breast cancer dataset [46] with the expression levels for 286 patients, 106 of whom developed metastasis within a 5-year follow-up after surgery. In the analysis, we considered three tumor suppressor genes, $BRCA1$, $BRCA2, TP53$, and their direct neighbors in a protein-protein interaction (PPI) network [47], and their corresponding PPI as our prior gene network, leading to 294 genes (nodes) with 326 edges. Among the 294 genes were 40 cancer genes [26] [43] and 7 (*ABL1, JAK2, TP53, PTEN, p14ARF, PTCH, RB*) cancer genes with high mutation frequencies (i.e. larger than 0.10). The outcome was the event time to metastasis, which might be right censored. Since all the methods were developed for linear regression, we approximated a Cox proportional hazards model by a linear model: we first fitted a null proportional hazards model with no predictors, then used its deviance residuals as the outcome for a linear model [48]. The full linear model included all the 294 genes as its candidate predictors.

We standardized the data in the following way: across the samples, the outcomes were centered to have mean 0, and each gene's expression levels were standardized to have mean 0 and standard deviation 1. Since the sample size was relatively small, we ran each method 20 times. In each of 20 runs, the data were randomly split into the training, tuning, and test sets with 95, 95, 96 observations respectively. We compared the methods' performance in PE, selection of cancer (CA) genes and of high mutation cancer genes (CA-HMF), and model size, all averaged over 20 runs. We set a common set of candidate values for the tuning parameters over 20 runs for each method; the candidate tuning parameter values minimizing the averaged PRSS for the tuning data over the 20 runs were selected and used to fit a final model to the whole dataset. As before, we explored the use of two weights, $w_j = 1$ and $w_j = \sqrt{d_j}$; since for this dataset, it is known that some important cancer hub genes,

like TP53, had only moderate to small effect sizes, and it is desirable to select those hub genes [26], we present the results only for using weight $w_j = \sqrt{d_j}$ that favored the selection of hub genes, though similar conclusions were reached with the other weight.

As shown in Table 4.4, averaged over the 20 runs, the Lasso detected only 0.4 CA genes out of a total of 11.2 selected genes; in comparison, the Enet, Grace and aGrace detected about 1.35–2.4 CA genes from about 30 selected genes. Impressively, the $TTLP_I$ and $LTLP_I$ detected 3.3 and 4.0 CA genes from only 10.6 and 12.5 selected genes respectively. For the final models, the Lasso selected 16 genes with no CA genes, while Enet, Grace and aGrace selected 102, 83 and 71 genes with 3, 2 and 4 CA genes respectively. As a comparison, both $TTLP_I$ and $LTLP_I$ detected 3 CA genes out of the 10 and 16 selected genes respectively. On the other hand, the PEs of most methods were very close, while it might not be very meaningful to compare the PEs since a linear model was used to approximate a proportional hazards model. Hence we focus on gene selection. It is interesting to note that our two new methods selected the three hub genes, BRCA1, BRCA2 and TP53, most frequently over the 20 runs. Furthermore, they were the only two methods selecting all the three hub genes in the final model. Figure 1 shows the selected genes in the final model for $LTLP_I$.

4.5 Discussion

In this study, we have proposed a network-based penalized regression approach with a novel penalty $TTLP_I$ containing two penalty terms for two different goals: the first uses a TLP for variable selection while the second (TLP_I) smooths approximate indicators of the nodes' being selected over a network. We have also considered one of its modifications by replacing the TLP by the Lasso for variable selection for not-too-sparse models. Our main contribution is that, in contrast to previously developed network-based methods aiming to smooth the (weighted) regression coefficients or their absolute values over a network [40, 41, 17], we adopt a less stringent assumption to smooth the indicators of the regression coefficients' being non-zero. Specifically, for any two neighboring nodes $j \sim j'$ in a network, rather than assuming and thus encouraging $\beta_j/w_j \approx \beta_{j'}/w_{j'}$ or $|\beta_j|/w_j \approx |\beta_{j'}|/w_{j'}$, our method assumes and aims

to smooth $I(|\beta_j|/w_j \neq 0) \approx I(|\beta_{j'}|/w_{j'} \neq 0)$. As shown in our simulation studies, if the former assumption holds, then some existing methods, such as Grace and aL_∞, which fully incorporate this former assumption, may be more efficient; however, even in this situation, the proposed methods seem to be robust with good performance. More generally, if this assumption does not hold, or even if this assumption holds but the weights w_j are mis-specified, then the proposed methods perform much better. In particular, in our real data application, we have demonstrated the effectiveness of the proposed methods in selecting biologically important hub genes with only small to moderate effect sizes. In summary, we regard our proposed methods as a useful tool complementary to existing methods.

We note that, although our methods encourage simultaneous selection (or elimination) of any two nodes connected in a network, it is related but significantly different from group variable selection. A main difference is that group variable selection only indirectly encourages simultaneous elimination, but not simultaneous selection. Furthermore, existing penalties for group variable selection, e.g. the L_γ-norm for $\gamma > 1$, have strong shrinkage effects on parameter estimation, often leading to severely biased parameter estimation, as demonstrated by the L_∞ method compared here. In addition, although we focus on network-based regression, our proposed penalty can be also applied to more general grouping problems [23, 15]; for example, with no given network, we can construct a complete graph with an edge connecting each pair of nodes, or we can form a linear chain graph as used in the fused Lasso [5], before applying our methods. More studies are needed.

Computationally, we have developed a DC method to relax a non-convex minimization problem into iterative convex programs. Currently we use the existing MATLAB package `CVX` for convex programming; a more efficient implementation for high-dimensional data is desired. In particular, to save computing time, we used a common tuning parameter τ for both variable selection and network smoothing; using two different τ_1 and τ_2 might perform better. In addition, due to the presence of multiple tuning parameters, we only searched a limited number of grid points (4 to 5) for each tuning parameter, which might not be optimal. Developing more efficient computational algorithms for and further investigation on properties of our proposed methods are worthwhile for future study.

Table 4.1: Simulation I: Mean (sd) of ME and PE, mean [median] (sd) of the numbers of TP, FP and the TFs from 100 simulated datasets for each set-up. The true numbers of TP are 22 for set-ups 1 and 2, and 12 for set-up 3. The true number of TFs is 2 for all set-ups.

Set-up	w	Method	ME(sd)	PE(sd)	TP	FP	TF
1	1	Lasso	44.2(13.2)	66.2(13.1)	13.5[14](3.2)	16.8[13](19.2)	2.0[2](0.2)
		Enet	34.2(13.1)	65.0(13.5)	16.5[17](3.7)	22.2[18](16.6)	2.0[2](0.2)
		Grace	15.1(3.9)	42.4(6.4)	21.9[22](0.9)	61.2[66.5](21.0)	2.0[2](0.0)
		aGrace	31.9(13.2)	61.3(14.9)	16.9[17](4.1)	24.3[18.5](19.7)	2.0[2](0.2)
		L_∞	41.2(12.4)	71.1(17.6)	21.1[22](2.0)	13.7[12](9.9)	1.8[2](0.4)
		aL_∞	19.3(8.8)	45.4(12.5)	21.1[22](2.1)	4.6[3](9.5)	1.8[2](0.4)
		$TTLP_I$	24.8(20.6)	50.5(11.4)	20.4[22](4.6)	12.0[0.0](21.5)	2.0[2](0.0)
		$LTLP_I$	17.6(9.5)	46.7(9.8)	21.3[22](2.2)	17.0[11](14.2)	2.0[2](0.0)
	$\sqrt{d}$	Grace	4.7(3.6)	39.7(5.8)	22.0[22](0.1)	59.5[63](21.2)	2.0[2](0.0)
		aGrace	23.9(16.4)	55.6(14.4)	17.6[18](4.1)	29.4[23.5](22.3)	2.0[2](0.2)
		L_∞	14.2(8.0)	50.4(11.2)	22.0[22](0.0)	9.7[8](6.8)	2.0[2](0.0)
		aL_∞	4.3(4.1)	38.8(6.0)	22.0[22](0.0)	4.1[2](5.4)	2.0[2](0.0)
		$TTLP_I$	12.4(12.0)	45.4(9.1)	21.5[22](2.7)	20.2[1](28.3)	2.0[2](0.0)
		$LTLP_I$	9.6(8.5)	43.4(8.5)	21.7[22](1.4)	23.4[22](17.0)	2.0[2](0.0)
2	1	Lasso	34.6(8.8)	67.9(11.4)	10.2[9.5](3.0)	13.4[9.0](15.4)	1.8[2](0.4)
		Enet	34.8(8.5)	68.2(11.4)	13.2[13.0](4.3)	24.4[18](22.1)	1.9[2](0.3)
		Grace	37.6(7.2)	63.4(10.1)	17.7[19.5](4.9)	42.5[38.5](27.1)	2.0[2](0.1)
		aGrace	33.7(8.3)	63.8(11.5)	15.0[15](5.6)	32.3[27](25.5)	1.9[2](0.3)
		L_∞	54.2(6.8)	77.2(12.1)	13.0[14](3.9)	12.1[11](6.9)	0.6[1](0.6)
		aL_∞	48.9(10.1)	71.5(12.8)	12.7[13](3.7)	8.1[8](5.4)	0.6[1](0.6)
		$TTLP_I$	33.9(14.0)	60.0(13.3)	20.3[22](3.7)	16.2[3](24.5)	2.0[2](0.2)
		$LTLP_I$	31.8(9.2)	58.3(9.8)	20.5[22](3.2)	30.5[29](21.1)	2.0[2](0.1)
	$\sqrt{d}$	Grace	27.1(5.7)	59.8(9.0)	18.5[19](3.4)	45.1[43.5](25.1)	2.0[2](0)
		aGrace	25.3(10.9)	58.4(11.6)	17.5[19](5.0)	41.9[39.5](24.1)	1.9[2](0.2)
		L_∞	34.5(10.2)	65.1(12.2)	20.9[22](2.6)	15.2[13](11.0)	1.8[2](0.4)
		aL_∞	20.7(9.9)	53.5(11.6)	20.7[22](3.1)	8.3[5](10.7)	1.8[2](0.4)
		$TTLP_I$	28.5(11.0)	59.5(11.3)	21.0[22](3.3)	26.7[15](28.6)	2.0[2](0.2)
		$LTLP_I$	23.2(8.1)	55.3(9.3)	21.4[22](2.2)	37.2[33](21.4)	2.0[2](0.1)
3	1	Lasso	18.4(6.3)	43.1(8.6)	8.4[8.5](1.7)	14.7[12.5](11.4)	2.0[2](0.2)
		Enet	18.2(6.5)	43.5(8.7)	9.0[9.0](1.9)	17.5[17](11.6)	2.0[2](0.2)
		Grace	22.5(6.0)	40.7(6.7)	10.9[12](1.8)	52.7[60](32.6)	2.0[2](0.1)
		aGrace	20.0(7.0)	44.0(8.5)	9.0[9](1.9)	19.2[16](16.6)	2.0[2](0.2)
		L_∞	41.4(7.1)	57.4(12.4)	10.4[10](1.7)	20.5[20](6.8)	1.4[1](0.6)
		aL_∞	32.2(6.5)	46.1(10.1)	10.3[10](1.7)	12.9[12](4.5)	1.4[1](0.6)
		$TTLP_I$	21.4(7.8)	40.5(7.5)	9.7[12](3.3)	14.6[10](19.2)	2.0[2](0.0)
		$LTLP_I$	17.6(7.2)	39.5(7.1)	11.1[12](1.3)	28.7[22.5](16.8)	2.0[2](0.0)
	$\sqrt{d}$	Grace	12.8(3.5)	37.1(6.1)	11.6[12](1.2)	56.2[60](28.2)	2.0[2](0)
		aGrace	16.4(6.5)	41.8(8.3)	9.3[9](2.0)	27.1[19](22.6)	2.0[2](0.2)
		L_∞	19.9(5.6)	43.4(8.4)	11.9[12](0.4)	23.3[20](11.3)	2.0[2](0.1)
		aL_∞	13.7(4.0)	37.4(6.6)	11.9[12](0.4)	16.6[13](11.1)	2.0[2](0.1)
		$TTLP_I$	18.4(7.0)	40.0(7.2)	10.0[12](3.2)	21.9[10](25.9)	2.0[2](0)
		$LTLP_I$	13.6(4.1)	38.0(6.6)	11.5[12](1.0)	36.5[32](21.4)	2.0[2](0)

Table 4.2: Simulation II: Mean (sd) of ME and PE, mean [median] (sd) of the numbers of TP, FP and the TFs from 100 simulated datasets.The true number of TP is 22, and the true number of TFs is 2.

w	Method	ME(sd)	PE(sd)	TP	FP	TF
1	Lasso	36.2(9.4)	67.0(11.3)	10.0[10](3.3)	13.6[10](16.3)	0.7[1](0.6)
	Enet	34.9(7.9)	65.8(10.3)	12.7[12](3.8)	22.7[17](19.2)	1.1[1](0.7)
	Grace	32.8(7.5)	64.3(10.1)	14.0[13.5](3.7)	25.7[19.5](19.0)	1.5[2](0.7)
	aGrace	31.6(7.2)	62.5(9.0)	15.2[15](5.2)	31.8[23.5](24.6)	1.4[2](0.7)
	L_∞	34.4(8.3)	66.1(10.8)	10.2[10](3.1)	11.8[10](10.0)	0.1[0](0.4)
	aL_∞	34.0(8.4)	65.9(11.1)	10.1[10](3.1)	11.2[9.5](9.7)	0.1[0](0.4)
	$TTLP_I$	31.2(10.0)	62.1(11.5)	19.2[22](5.4)	17.9[11](22.4)	1.7[2](0.7)
	$LTLP_I$	28.1(8.0)	59.0(9.5)	20.6[22](3.6)	37.0[33.5](21.7)	1.8[2](0.5)
$\sqrt{d}$	Grace	34.9(7.8)	65.4(10.6)	13.6[14](4.2)	24.8[19](19.3)	1.4[2](0.7)
	aGrace	36.2(8.4)	63.1(9.0)	15.2[15](5.6)	32.0[24](24.3)	1.2[1](0.8)
	L_∞	33.9(8.1)	65.1(10.3)	15.3[15](4.6)	13.8[11](11.5)	1.0[1](0.6)
	aL_∞	37.6(9.2)	66.0(12.1)	15.0[15](4.7)	9.7[7.5](11.0)	1.0[1](0.6)
	$TTLP_I$	34.2(10.1)	63.9(10.9)	19.1[22](5.2)	20.1[13](22.7)	1.6[2](0.7)
	$LTLP_I$	31.3(7.4)	61.1(9.6)	20.5[22](3.7)	39.2[44](21.9)	1.8[2](0.5)

Table 4.3: Simulation I: Mean, sd, and MSE of regression coefficient estimates from 100 simulated datasets in each set-up.

				β_1=5			β_2=1.58			β_{11}=1.58	
Set-up	w	Methods	Mean	sd	MSE	Mean	sd	MSE	Mean	sd	MSE
1	1	Lasso	7.10	1.66	9.92	0.98	1.03	2.46	0.93	0.99	2.40
		Enet	5.20	1.69	5.71	1.19	0.91	1.82	1.15	0.94	1.95
		Grace	1.99	0.73	10.11	1.93	0.22	0.21	1.91	0.24	0.22
		aGrace	3.96	2.17	10.46	1.46	0.96	1.86	1.39	1.01	2.05
		L_∞	1.09	0.34	15.49	1.88	0.98	2.00	1.94	1.04	2.28
		aL_∞	2.03	0.20	8.93	2.03	0.28	0.36	2.04	0.24	0.32
		$TTLP_I$	4.35	2.51	12.96	1.68	0.94	1.76	1.69	0.83	1.40
		$LTLP_I$	4.05	1.93	8.32	1.64	0.77	1.18	1.69	0.70	0.98
	$\sqrt{d}$	Grace	4.70	0.42	0.44	1.49	0.19	0.08	1.48	0.20	0.09
		aGrace	5.35	1.16	2.79	1.29	0.72	1.12	1.21	0.78	1.36
		L_∞	3.93	0.54	1.73	1.47	0.60	0.73	1.55	0.63	0.80
		aL_∞	4.95	0.47	0.44	1.54	0.38	0.29	1.59	0.23	0.11
		$TTLP_I$	5.28	1.08	2.42	1.52	0.79	1.25	1.53	0.61	0.73
		$LTLP_I$	5.16	0.90	1.65	1.44	0.63	0.82	1.52	0.58	0.66
				β_1=5			β_2=-1.58			β_{11}=1.58	
2	1	Lasso	3.83	1.39	5.18	-0.04	0.31	2.57	0.92	0.97	2.29
		Enet	2.89	1.05	6.64	-0.03	0.38	2.69	1.08	0.89	1.82
		Grace	1.66	1.12	13.66	0.36	0.44	4.16	1.29	0.72	1.13
		aGrace	2.35	1.24	10.10	-0.26	0.81	3.05	1.34	0.92	1.73
		L_∞	0.15	0.23	23.64	0.07	0.39	3.02	1.59	1.24	3.04
		aL_∞	0.70	0.79	19.75	0.09	0.95	4.59	1.60	1.01	2.01
		$TTLP_I$	3.23	1.78	9.44	-0.61	1.33	4.46	1.70	0.79	1.27
		$LTLP_I$	2.77	1.43	9.05	-0.40	1.27	4.64	1.53	0.71	1.00
	$\sqrt{d}$	Grace	2.90	0.53	4.97	0.24	0.38	3.62	1.07	0.54	0.85
		aGrace	3.88	0.83	2.63	-0.51	0.75	2.27	1.15	0.75	1.31
		L_∞	2.18	0.77	9.13	-0.10	0.65	3.04	1.34	0.86	1.51
		aL_∞	3.94	0.78	2.32	-0.31	1.24	4.65	1.31	0.36	0.34
		$TTLP_I$	3.73	1.30	4.97	-0.33	1.10	3.99	1.36	0.77	1.23
		$LTLP_I$	3.55	0.96	3.95	-0.29	1.08	3.99	1.30	0.66	0.95
				β_1=5			β_2=0			β_{11}=1.58	
3	1	Lasso	5.15	1.22	2.98	0.13	0.35	0.25	1.01	0.93	2.05
		Enet	4.47	1.20	3.14	0.20	0.41	0.37	1.08	0.90	1.87
		Grace	2.45	1.76	12.62	0.81	0.56	1.28	1.27	0.62	0.86
		aGrace	3.94	1.61	6.28	0.25	0.53	0.62	1.20	0.98	2.04
		L_∞	0.43	0.29	21.05	0.66	0.66	1.30	1.81	1.11	2.51
		aL_∞	1.26	0.48	14.44	1.05	0.69	2.07	1.56	0.57	0.64
		$TTLP_I$	4.91	2.13	9.02	0.33	0.71	1.10	1.20	0.98	2.06
		$LTLP_I$	3.94	1.50	5.62	0.40	0.71	1.16	1.31	0.71	1.07
	$\sqrt{d}$	Grace	3.45	0.86	3.90	0.71	0.38	0.79	1.11	0.46	0.66
		aGrace	4.37	1.05	2.58	0.22	0.55	0.66	1.11	0.80	1.50
		L_∞	2.46	0.52	6.98	0.59	0.50	0.85	1.42	0.77	1.21
		aL_∞	3.55	0.54	2.68	0.74	0.80	1.81	1.21	0.32	0.34
		$TTLP_I$	4.82	1.86	6.88	0.40	0.72	1.20	1.13	0.90	1.83
		$LTLP_I$	4.14	1.07	3.02	0.43	0.68	1.10	1.23	0.65	0.97

Table 4.4: Results for the breast cancer data with $w = \sqrt{d}$: PE (se), mean [median] (se) of # of selected cancer (CA) genes, cancer genes with high mutation frequencies larger than 0.10 (CA-HMF), and selected genes (Genes) over 20 runs. Frequencies of selecting BRCA1, BRCA2 and TP53 in 20 runs and their inclusion (yes/no) in the final model, and the genes selected more than 10 times out of 20 runs are also included.

Method	PE	# CA	# CA-HMF	# Genes
Lasso	0.372(0.004)	0.40[0](0.13)	0.15[0](0.08)	11.20[11.5](1.96)
Final	-	0	0	16
Enet	0.363(0.005)	1.35[1](0.31)	0.45[0](0.15)	32.05[32](5.09)
Final	-	3	1	102
Grace	0.364(0.005)	2.00[2](0.27)	0.60[0.5](0.15)	33.90[32.5](4.91)
Final	-	2	1	83
aGrace	0.364(0.005)	2.40[2](0.28)	0.35[0](0.13)	28.15[26.5](4.85)
Final	-	4	1	71
L_∞	0.370(0.005)	0.10[0](0.10)	0.20[0](0.12)	14.45[14](1.61)
Final	-	0	0	12
aL_∞	0.373(0.004)	0.10[0](0.10)	0.20[0](0.12)	13.40[14.5](1.92)
Final	-	0	0	11
$TTLP_I$	0.431(0.017)	3.30[3](0.31)	0.10[0](0.07)	10.60[11](1.57)
Final	-	3	0	10
$LTLP_I$	0.371(0.038)	4.00[4](0.54)	0.15[0](0.08)	12.50[13](1.61)
Final	-	3	0	16
	# BRCA1, BRCA2, TP53 & (y/n) in Final Model			
Lasso	1(n), 0(n), 1(n)			
Enet	2(n), 0(n), 2(n)			
Grace	8(n), 4(n), 5(n)			
aGrace	12(y), 5(n), 12(y)			
L_∞	0(n), 0(n), 0(n)			
aL_∞	0(n), 0(n), 0(n)			
$TTLP_I$	20(y), 13(y), 20(y)			
$LTLP_I$	16(y), 16(y), 16(y)			
	Genes selected ≥ 10 times			
Lasso	MAPK9			
Enet	CD74 ERCC2 HIF1A MAPK9 TOP1			
Grace	CD74 HIF1A MAPK9 TOP1			
aGrace	BRCA1 HIF1A MAPK9 TP53			
L_∞	MAPK9			
aL_∞	MAPK9			
$TTLP_I$	BRCA1 BRCA2 TP53			
$LTLP_I$	BARD1 BRCA1 BRCA2 MAPK9 TP53			

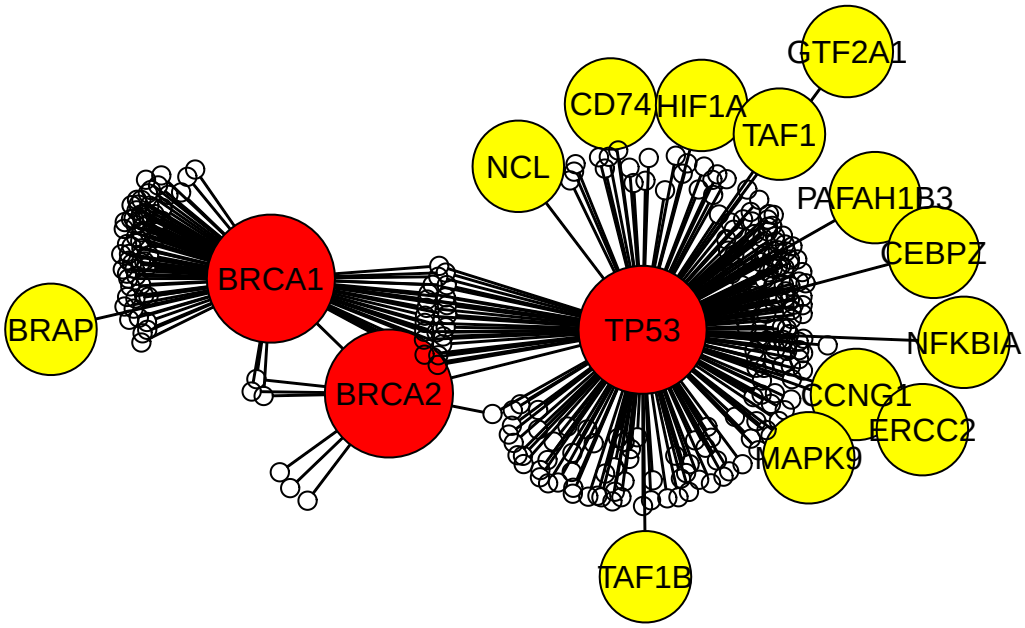


Figure 4.1: The final model by $LTLP_I$: 16 selected genes including 3 tumor suppressor genes (BRCA1, BRCA2 and TP53) (in red and larger circles) and the other 13 genes (in yellow and smaller circles).

4.6 Programs

[R code]

```
#######################################
# sk produces Rep datasets of
# training (@ sk$Tr), tuning (@ sk$Tu),
# and testing (@ sk$Te)

# Input
# nTF : number of transcription factor
#  b  : true coefficient vector of
#       size 1 x p
#######################################

sk<-function(nTF,b)
{
Ntr<-Ntu<-50
Nte<-200
nsim<-100
nTarget<-10

datTr=matrix(0,nsim*Ntr,length(b)+1)
datTu=matrix(0,nsim*Ntu,length(b)+1)
datTe=matrix(0,nsim*Nte,length(b)+1)

Ve<-sum(b*b)/2

for(sim in 1:nsim){

Ytr<-rep(0, Ntr)
Xtr<-matrix(0, nrow=nTF*(nTarget+1),
     ncol=Ntr)
Ytu<-rep(0, Ntu)
Xtu<-matrix(0, nrow=nTF*(nTarget+1),
     ncol=Ntu)
Yte<-rep(0, Nte)
Xte<-matrix(0, nrow=nTF*(nTarget+1),
     ncol=Nte)

set.seed(sim)

j<-1
for(i in 1:nTF){
 Xtr[j,]<-rnorm(Ntr, 0, 1)
 j<-j+1
 for(k in 1:nTarget){
 Xtr[j,]<-rnorm(Ntr,
          0.5*Xtr[(i-1)*(nTarget+1)+1,],
          sqrt(0.75))
 j<-j+1
  }
 }
Ytr<-rnorm(Ntr, 0, sqrt(Ve)) + b%*%Xtr

j<-1
for(i in 1:nTF){
 Xtu[j,]<-rnorm(Ntu, 0, 1)
 j<-j+1
 for(k in 1:nTarget){
 Xtu[j,]<-rnorm(Ntu,
          0.5*Xtu[(i-1)*(nTarget+1)+1,],
          sqrt(0.75))
 j<-j+1
    }
  }
Ytu<-rnorm(Ntu, 0, sqrt(Ve)) +  b%*%Xtu

j<-1
for(i in 1:nTF){
 Xte[j,]<-rnorm(Nte, 0, 1)
 j<-j+1
 for(k in 1:nTarget){
 Xte[j,]<-rnorm(Nte,
          0.5*Xte[(i-1)*(nTarget+1)+1,],
          sqrt(0.75))
 j<-j+1
 }
  }
Yte<-rnorm(Nte, 0, sqrt(Ve)) +  b%*%Xte

#normalize training
Y<-Ytr - mean(Ytr)
X<-Xtr
Xmu<-apply(X, 1, mean)
Xsd<-sqrt(apply(X, 1, var))
for(i in 1:length(Xmu))
  X[i,]<-( X[i,] - Xmu[i])/Xsd[i]
X<-t(X)

#normalize tuning:
Ytu0<-Ytu - mean(Ytr)
Xtu0<-Xtu
for(i in 1:length(Xmu))
  Xtu0[i,]<-( Xtu0[i,] - Xmu[i])/Xsd[i]
Xtu0<-t(Xtu0)

#normalize testing:
Yte0<-Yte - mean(Ytr)
Xte0<-Xte
for(i in 1:length(Xmu))
```

```
  Xte0[i,]<-( Xte0[i,] - Xmu[i])/Xsd[i]
Xte0<-t(Xte0)

Tr=cbind(t(Y),X)
Tu=cbind(t(Ytu0),Xtu0)
Te=cbind(t(Yte0),Xte0)
datTr[(1+Ntr*(sim-1)):(Ntr*sim),]=Tr
datTu[(1+Ntu*(sim-1)):(Ntu*sim),]=Tu
datTe[(1+Nte*(sim-1)):(Nte*sim),]=Te
 }

return(list(Tr=datTr,Tu=datTu,Te=datTe))

}

##################################
# Data generate example
# of simulation I
##################################

a=c(5,rep(5/sqrt(10),10))
b=c(-3,rep(-3/sqrt(10),10))

#Set-up 1
 bC1=c(a,b,rep(0,110-22))
 dat=sk(nTF=10,b=bC1)
 Tr=dat$Tr; Tu=dat$Tu; Te=dat$Te;

#Set-up 2
 r=c(1,-1,-1,-1,1,1,1,1,1,1,1)
 bC2=c(a*r,b*r,rep(0,110-22))
 dat=sk(nTF=10,b=bC2)
 Tr=dat$Tr; Tu=dat$Tu; Te=dat$Te;

#Set-up 3
 r=c(1,0,0,0,0,0,1,1,1,1,1)
 bC3=c(a*r,b*r,rep(0,110-22))
 dat=sk(nTF=10,b=bC3)
 Tr=dat$Tr; Tu=dat$Tu; Te=dat$Te;


[MATLAB]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% TIP produces the summary of
% 1. ME and PE
% 2. TP and FP
% 3. bhat, sd(bhat), and MSE(bhat)
% for TTLP_{I} and LTLP with w=sqrt(d)

% Input :
% b : true coefficient vector
% Tr : Training set
% Tu : Tuning set
% Te : Testing set
% rho : correlation btw TF and target gene
% netwk : netwk matrix of
%         size # of edge x 2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function[MEPE,TPFP,bsdmse]=
         TIP(b,Tr,Tu,Te,Rep,rho,netwk)

format shortG
nnetwk=10;
epsilon=1e-6;

%%data
 [tn,p]=size(Tr(:,2:end));
 n=tn/Rep;

 tm=size(Te(:,2:end),1);
 m=tm/Rep;

%w=sqrt(d)
a=netwk(:);
wt=ones(p,1);
for j=1:p
 wt(j)= sqrt(sum(a==j));
end

netp=p/nnetwk;

q1= abs(b)>0;
q0= abs(b)==0;

%%covX=E[X'X]
 covX1=ones(netp,netp)*n*rho^2;
 covX1(1:netp+1:end)=n;
  a=cell(1,nnetwk);
 [a{:}]=deal(sparse(covX1));
 samcovX=blkdiag(a{:});
 covX=full(samcovX);

tX=Tr(:,2:end);
tY=Tr(:,1);

tXtu=Tu(:,2:end);
tYtu=Tu(:,1);

tXte=Te(:,2:end);
tYte=Te(:,1);

%% Summary Table
```

```
ME=zeros(Rep,2);
PE=zeros(Rep,2);
TP=zeros(Rep,2);
FP=zeros(Rep,2);

%% bias square
bs_TLP=zeros(Rep,p);
bs_LTLP=zeros(Rep,p);

for d =1:Rep

 disp(d)

 Y=tY((1+n*(d-1)):(n*d),1);
 X=tX((1+n*(d-1)):(n*d),:);
 Ytu=tYtu((1+n*(d-1)):(n*d),1);
 Xtu=tXtu((1+n*(d-1)):(n*d),:);
 Yte=tYte((1+m*(d-1)):(m*d),1);
 Xte=tXte((1+m*(d-1)):(m*d),:);

% grid search range
gd1=4;gd2=4;gd3=5;
mx=max(abs(blasso));
K1=linspace(mx,mx/4*p,gd1);
edge=size(netwk,1);
K2=linspace(mx,mx*edge,gd2);
ltau=linspace(epsilon,mx/2,gd3);

%% TTLPI
tunM=zeros(gd1,gd2*gd3);
for j=1:gd1
for k=1:gd2
 for s=1:gd3
 btr=
 cvxtip(Y,X,K1(j),K2(k),ltau(s),ltau(s),
        blasso,netwk,wt);
 tunM(j,(k-1)*gd3+s)=norm(Ytu-Xtu*btr)^2;
 end
 end
end
tmp=tunM>0;
[row,col]=find(tunM==min(tunM(tmp)));
row=row(1);col=col(1);
del1=mean(K1(row));
if rem(col,gd3)==0,
   del2=mean(K2(floor(col/gd3)));
   tau=mean(ltau(gd3));
else del2=mean(K2(floor(col/gd3)+1));
     tau=mean(ltau(col-floor(col/gd3)*gd3));
end
bTLP=cvxtip(Y,X,del1,del2,tau,tau,
            blasso,netwk,wt);

%% LTLP
LtunM=zeros(gd1,gd2*gd3);
K1=linspace(
            lamlasso(ind)/1.5,
            lamlasso(ind)*1.5,gd1
           );
for j=1:gd1
 for k=1:gd2
  for s=1:gd3
  btr=cvxtip(Y,X,K1(j),K2(k),100,
             ltau(s),blasso,netwk,wt);
  LtunM(j,(k-1)*gd3+s)=norm(Ytu-Xtu*btr)^2;
  end
 end
end
tmp=LtunM>0;
[row,col]=find(LtunM==min(LtunM(tmp)));
row=row(end);col=col(end);
del1=mean(K1(row));
if rem(col,gd3)==0,
   del2=mean(K2(floor(col/gd3)));
   tau=mean(ltau(gd3));
else del2=mean(K2(floor(col/gd3)+1));
   tau=mean(ltau(col-floor(col/gd3)*gd3));
end
bLTLP=cvxtip(Y,X,del1,del2,100,
             tau,blasso,netwk,wt);

%bias^2
 bs_TLP(d,:)=(bTLP-b).*(bTLP-b);
 bs_LTLP(d,:)=(bLTLP-b).*(bLTLP-b);

%ME
 ME(d,1)=(b-bTLP)'*covX*(b-bTLP)
         /length(Y);
 ME(d,2)=(b-bLTLP)'*covX*(b-bLTLP)
         /length(Y);
 %PE
 PE(d,1)=dot(Yte-Xte*bTLP,Yte-Xte*bTLP)
         /length(Yte);
 PE(d,2)=dot(Yte-Xte*bLTLP,Yte-Xte*bLTLP)
         /length(Yte);
%TP
   TP(d,1) = sum(abs(bTLP(q1))>0.001);
   TP(d,2) = sum(abs(bLTLP(q1))>0.001);
%FP
   FP(d,1) = sum(abs(bTLP(q0))>0.001);
 FP(d,2) = sum(abs(bLTLP(q0))>0.001);

end
```

```
%% Summary
MEPE=[mean(ME)' sqrt(var(ME))'
      mean(PE)' sqrt(var(PE))'];
TPFP=[mean(TP)' sqrt(var(TP))'
      mean(FP)' sqrt(var(FP))'];
bsdmse=
[mean(b_TLP)' sqrt(var(b_TLP))'
     (mean(bs_TLP)+var(b_TLP))'
 mean(b_LTLP)' sqrt(var(b_LTLP))'
     (mean(bs_LTLP)+var(b_LTLP))'];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% cvxtip produces final bhat in iterated
% DC algorithm given tuning parameters
% (del1,del2,tau).
% Note that in our method, tau1=tau2

% Input
% b0 : initial b estimate
% wt : weight vector of size p x 1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function[b1]=
cvxtip(Y,X,del1,del2,tau1,tau2,b0,netwk,wt)

b1=cvxmainalgo(Y,X,del1,del2,
               tau1,tau2,b0,netwk,wt);
gb1=cvxggg(Y,X,b1,del1,del2,
           tau1,tau2,netwk,wt);
b2=cvxmainalgo(Y,X,del1,del2,tau1,
               tau2,b1,netwk,wt);
gb2=cvxggg(Y,X,b2,del1,del2,
           tau1,tau2,netwk,wt);

while (gb1-gb2>0.0001)
 gb1=gb2;
 b1=b2;
 b2=cvxmainalgo(Y,X,del1,del2,tau1,
                tau2,b1,netwk,wt);
 gb2=cvxggg(Y,X,b2,del1,del2,tau1,
                tau2,netwk,wt);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% cvxmainalgo produces bhat in 1 iteration
% of DC algorithm.

% Input :
% 1.(del1,del2,tau1,tau2) : Tuning param-
% eter set where tau1 is a threshold in
% variable selection, and tau2 is a
% threshold in grouping.

% 2. b0 : current estimate
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function[x1] = cvxmainalgo(Y,X,del1,del2,
                tau1,tau2,b0,netwk,wt)

p=size(X,2);

%1st constraint
lt=abs(b0)<=tau1;

%2nd constraint
lgt=abs(b0)./wt> tau2;
S2=sign(b0).*(1+lgt);

cvx_begin quiet
variable x(p);
minimize
(

 0.5*square_pos(norm(Y-X*x))+
 del1*lt'*abs(x)+
 2*del2*
 sum(
   max(
       max(abs(x(netwk(:,1)))./wt(netwk(:,1)),
       abs(x(netwk(:,2)))./wt(netwk(:,2))),
       abs(x(netwk(:,1)))./wt(netwk(:,1))+
       abs(x(netwk(:,2)))./wt(netwk(:,2))-tau2
      )
    ) -
 del2*
  (
   S2(netwk(:,1))'*(x(netwk(:,1))./wt(netwk(:,1)))+
   S2(netwk(:,2))'*(x(netwk(:,2))./wt(netwk(:,2)))
  )

);
cvx_end
x1=x;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sbeta produces S(m)(x) at current estimate x
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[Sbeta] = cvxggg(Y,X,x,del1,del2,tau1,
                  tau2,netwk,wt)

 c1=sum(min(abs(x)/tau1,1));
 c2=sum(
 min((abs(x(netwk(:,1)))./wt(netwk(:,1))) /tau2,1)
 -
```

```
min((abs(x(netwk(:,2))./wt(netwk(:,2)))/tau2,1)  del1*tau1*c1+del2*tau2*c2);
       );                                  end

Sbeta=(0.5*square_pos(norm(Y-X*x))+
```

Chapter 5

Conclusions and Discussion

We have proposed several new penalized regression approaches to genetic and genomic data analysis. They are all based on the TLP, which approximates the L_0-penalty. In Chapter 2 we apply TLP for variable selection and grouping in hypothesis testing to investigate whether there is any association between rare variants and a quantitative trait. The testing is addressed using a new 1 degree of freedom F-test and the SSU(w) test statistic. While there have been a few negative reports on the application of the penalized regression methods to hypothesis testing in GWAS, they mostly focused on the simple Lasso. We explore newer methods in hypothesis testing. The comparisons on the power using the TLP and other penalized methods demonstrate that TLP based tests may or may not improve over the SSU or SSUw test. Though the parameter estimation bias is reduced by TLP approach, the overall conservativeness to detect true causal variants might be a reason, where the conservativeness might be related to the non-optimal model selection criteria when choosing tuning parameters.

In Chapter 3, we extend the TLP approach to a high dimensional setting ($p > n$) to deal with the genomic dataset of GAW17. Allowing grouping within a gene, our proposed method is likely to detect the true positives as competitively as others, but it yields fewer false positives at both the SNP and gene levels. The TLP grouping unlike other methods leads to the effective deletion of noise variables by combining weakly associated ones to a trait with each other.

Chapter 4 further develops a subnetwork selection method for outcome prediction and gene selection using gene-network information via our DC programming that converts the non-convex problem into convex sub-problems. While pre-existing methods primarily attempt to group regression coefficients toward each other, we relax this assumption to simply group their non-zero indicators. Simulation studies, along with the application to real gene expression breast cancer dataset, reveal the competitiveness of the proposed method in both gene selection and prediction over pre-existing methods.

Finally, we address possible directions for further investigation. Taking the full benefit of penalized regression in genetic hypothesis testing needs a examination of other choices of model selection criteria since the performance of the methods largely depends on the chosen criteria. On the other hand, for the use of our proposed method in genomic-wide data of Chapter 3, variable selection across genes might

be interesting, too. Additionally, our proposed method TLP_I or $LTLP_I$ in Chapter 4 might run slowly for larger p in our implemented algorithm in MATLAB; thus, this might limit the number of tuning parameters that can be explored in a timely manner. Converting the program to C would allow us to explore more candidate tuning parameters and possibly select more precise final model as [19] recently did for functions similar to those used in other chapters.

References

[1] Basu S, Pan W, Shen X, and Oetting W. Multi-locus assocation testing with penalized regression. *Genetic Epidemiology*, 35(8):755–765, 2011.

[2] Hoerl A and Kennard R. Ridge regression. *In Encyclopedia of Statistical Sciences*, 8:129–136, 1988.

[3] Tibshirani R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

[4] Efron B, Hastie T, Johnstone I, and Tibshirani R. Least angle regression (with discussion). *The Annals of Statistics*, 32(2), 2004.

[5] Tibshirani R, Saunders M, Rosset S, Zhu J, and Knight K. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society.*, 67(1):91–108, 2005.

[6] Rinaldo A. Properties and refinements of the fused lasso. *The Annals of Statistics*, 37:2922–2952, 2009.

[7] Yuan M and Lin Y. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Ser.B*, 68:49–67, 2006.

[8] Meinshausen N. Relaxed lasso. *Computational Statistics and Data Analysis*, 51(1):374–393, 2007.

[9] Zou H and Hastie T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Ser.B*, 67:301–320, 2005.

[10] Shen X, Pan W, and Zhu Y. Likelihood-based selection and sharp parameter estimation. *Journal of the American Statistical Association*, 107, 2012.

[11] Almasy LA, Dyer TD, Peralta JM, Kent JW Jr, Charlesworth JC, Curran JE, and Blangero J. The biomolecular interaction network database and related tools 2005 update. *Nucleic Acids Res Database Issue*, 33(D418CD424), 2005.

[12] Peri *et al.* Human protein reference database as a discovery resource for proteomics. *Nucleic Acids Res Database Issue*, 32(D497CD501), 2004.

[13] Kanehisa M and Goto S. Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, 28:27–30, 2000.

[14] Ashburner *et al.* Gene ontology: tool for the unification of biology the gene ontology consortium. *Nature Genetics*, 25:25–29, 2000.

[15] Shen X and Huang H-C. Grouping pursuit through a regularization solution surface. *Journal of the American Statistical Association*, 105(490):727–739, 2010.

[16] Bondell HD and Reich BJ. Simultaneous regression shrinkage, feature selection and supervised clustering of predictors with oscar. *Biometrics*, 64:115–123, 2008.

[17] Pan W, Xie B, and Shen X. Incorporating predictor network in penalized regression with application to microarray data. *Biometrics*, 66(2):474–484, 2010.

[18] Kim S and Xing E. P. Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS genetics*, 5(8)(e1000587), 2009.

[19] Yang S, Yuan L, Shen X, Wonka P, and Ye J. FGSG: Feature Grouping and Selection Over an Undirected Graph. *Manuscript*, 2012.

[20] Maher B. Personal genomes: the case of the missing heritability. *Nature*, 456:18–21, 2008.

[21] Pan W. Asymptotic tests of association with multiple snps in linkage disequilibrium. *Genetic Epidemiology*, 33:497–507, 2009.

[22] Chapman JM and Whittaker J. Anaylsis of multiple snps in a candidate gene or region. *Genetic Epidemiology*, 32:560–566, 2008.

[23] Pan W. Network-based multiple locus linkage analysis of expression traits. *Bioinformatics*, 25(11):1390–1396, 2009.

[24] Basu S and Pan W. Comparison of statistical tests for disease association with rare variants. *Genetic Epidemiology*, 35(7):606–619, 2011.

[25] Almasy LA, Dyer TD, Peralta JM, Kent JW Jr, Charlesworth JC, Curran JE, and Blangero J. Genetic analysis workshop 17 mini-exome simulation. *BMC Proc*, 5(suppl 9)(S2), 2011.

[26] Conneely KN and Boehnke M. So many correlated tests, so little time! rapid adjustment of p values for multiple correlated tests. *Am J Hum Genet*, 81:1158–1168, 2007.

[27] Han F and Pan W. A data-adaptive sum test for disease association with multiple common or rare variants. *Human Heredity*, 70:42–54, 2010.

[28] Akaike and Hirotugu. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[29] Croiseau P and Cordell HJ. Analysis of north american rheumatoid arthritis consortium data using a penalized logistic regression approach. *BMC Proceedings*, 3 (suppl 7)(S61), 2009.

[30] Martinez JG, Carroll RJ, Muller S, Sampson JN, and Chatterjee N. A note on the effect on power of score tests via dimension reduction by penalized regression under the null. *The International Journal of Biostatistics*, 6(1):12, 2010.

[31] Kotowski IK, Pertsemlidis A, Luke A, Cooper RS, Vega GL, Cohen JC, and Hobbs HH. A spectrum of pcsk9 alleles contributes to plasma levels of low density lipoprotein cholesterol. *Am J HumGenet*, 78, 2006.

[32] Cohen J. C. *et al.* Multiple rare alleles contribute to low plasma levels of hdl cholesterol. *Science*, 305:869–872, 2004.

[33] Johnson *et al.* Counting potentially functional variants in brca1, brca2 and atm predicts breast cancer susceptibility. *Human Molecular Genetics*, 16:1051–1057, 2007.

[34] Tibshirani R Friedman J, Hastie T. Regularization paths for generalized linear models via coordinate descent. *J Stat Softw*, 33:1–22, 2010.

[35] Zhou H, Sehl ME, Sinsheimer JS, and Lange K. Association screening of common and rare genetic variants by penalized regression. *Bioinformatics*, 26:2375–2382, 2010.

[36] Guo W and Lin S. Generalized linear modeling with regularization for detecting common disease rare haplotype association. *Genet Epidemiol*, 33:308–316, 2009.

[37] Kooperberg C, LeBlanc M, and Obenchain V. Statistical estimation of correlated genome associations to a quantitative trait network. *Genet Epidemiol*, 34, 2010.

[38] Szymczak S, Biernacka JM, Cordell HJ, Gonzalez-Recio O, Konig IR, Zhang H, and Sun YV. Machine learning in genome-wide association studies. *Genet Epidemiol*, 33(1):S51–S57, 2009.

[39] Fan J and Li R. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, 2001.

[40] Li C and Li H. Network-constrained regularization and variable selection for analysis of genomic data. *Bioinformatics*, 24:1175–1182, 2008.

[41] Li C and Li H. Variable selection and regression analysis for graph-structured covariates with an application to genomics. *Annals of Applied Statistics*, 4:1498–1516, 2010.

[42] Percival D, Roeder K, Rosenfeld R, and Wasserman L. Structured, sparse regression with application to hiv drug resistance. *Annals of applied statistics*, 5(2A):628–644, 2011.

[43] Zhu Y, Shen X, and Pan W. Support vector machines with disease-gene-centric network penalty for high dimensional microarray data. *Stat Interface*, 2(3):257–269, 2009.

[44] Luo C, Pan W, and Shen X. A two-step penalized regression method with networked predictors. *Statistics in Biosciences*, 4(1):27–46, 2012.

[45] Grant M and Boyd S. Cvx: Matlab software for disciplined convex programming, version 1.21. *http://cvxr.com/cvx*, 2011.

[46] Wang Y, Klijn J, Zhang Y, Sieuwerts AM, Look MP, Yang F, Talantov D, Timmermans M, Meijer van Gelder ME, Yu J, Jatkoe T, Berns EMJJ, Atkins D, and Foekens JA. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365:671–679, 2005.

[47] Chuang HY, Lee EJ, YT, Lee DH, and Ideker T. Network-based classification of breast cancer metastasis. *Molecular Systems Biology*, 3(doi:10.1038/msb4100180), 2007.

[48] Segal M. R. Microarray gene expression data with linked survival phenotypes: diffuse large-b-cell lymphoma revisited. *Biostatistics*, 7:268–285, 2006.