Published in final edited form as:
Biometrics. 2013 September ; 69(3): 582–593. doi:10.1111/biom.12035.

Network-based penalized regression with application to genomic data

Sunkyung Kim[1], **Wei Pan**[1], and **Xiaotong Shen**[2]

Wei Pan: weip@biostat.umn.edu

[1]Division of Biostatistics, University of Minnesota, Minneapolis, Minnesota, 55405, USA

[2]School of Statistics, University of Minnesota, Minneapolis, Minnesota, 55455, USA

Summary

Penalized regression approaches are attractive in dealing with high-dimensional data such as arising in high-throughput genomic studies. New methods have been introduced to utilize the network structure 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, which however may not hold. Here 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, and an algorithm based on difference convex programming. We use simulated data and two breast cancer gene expression datasets to demonstrate the advantages of the proposed method over some existing methods. Our proposed methods can be applied to more general problems for group variable selection.

Keywords

Gene expression; networks analysis; nonconvex minimization; penalty; truncated Lasso penalty

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 (Tibshirani 1996), SCAD (Fan and Li 2001), elastic net (Enet) (Zou and Hastie 2005) and LARS (Efron *et al.* 2004), 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)

Correspondence to: Wei Pan, weip@biostat.umn.edu.

Supplementary Materials: Web Appendices A-C referenced in Sections 2-4 are available with this paper at the Biometrics website on Wiley Online Library.

(Alfarano *et al.* 2005) and the Human Protein Reference Database (HPRD) (Peri *et al.* 2004), biological pathways in the Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa and Goto 2000), and gene functional annotations in the Gene Ontology (Ashburner *et al.* 2000). 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 frequentist or Bayesian method is expected to improve its performance (Li and Zhang 2010, Monni and Li 2010, Tai *et al.* 2010, Huang and Wang 2012). In particular, as demonstrated in cancer marker discovery (Chuang *et al.* 2007), changes in expression of some causal genes governing metastatic potential (e.g. ERBB2 and MYC) may be only subtle and non-significant while some of their neighbors have much stronger alterations; incorporation of gene networks can improve the chance of identifying these cancer-causing genes.

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 $w_{j'}$ (Li and Li 2008). More generally, since the effect directions could be different, e.g. regulation of gene expression could be either stimulatory or inhibitory, we can assume $|\beta_j|/w_j \approx |\beta_{j'}|/w_{j'}$ (Li and Li 2010, Pan *et al.* 2010). 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 Percival *et al.* (2011) 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 (or the L_0-loss) 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 idea 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 research topics.

2. Methods

2.1 Review: penalized regression

In a linear regression model,

$$Y_i=\beta_0+\sum_{j=1}^{p}X_{ij}\beta_j+\varepsilon_i, E(\varepsilon_i)=0, \quad (1)$$

for $i = 1,\ldots, n$, 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

$$S(\beta)=\frac{1}{2}\|Y-X\beta\|^2+P(\beta). \quad (2)$$

The penalty $P(\beta)$ not only regularizes parameter estimation as desired, but also can realize effective variable selection. The Lasso (Tibshirani 1996) 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, Li and Li (2008), Li and Li (2010) and Pan *et al.* (2010) 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 (Li and Li 2008) with penalty

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

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 (Li and Li 2010):

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

where $\tilde{\beta}_j$ is an initial estimate based on OLSE for $p < n$, or an elastic net (Enet) estimate (Zou and Hastie 2005) for $p \geq n$. The main idea is to use sign($\tilde{\beta}_j$) to estimate sign(β_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 (Yuan and Lin 2006, Zhao *et al.* 2009). Pan *et al.* (2010) demonstrated better performance of the method for variable selection than Lasso, Enet and Grace, though the parameter estimates may be severely biased. Luo *et al.* (2012) proposed a 2-step procedure similar to that of Li and Li (2010) 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 (Bondell *et al.* 2008), 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.

2.2 New methods

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

$$P(\beta)=\lambda_1\sum_{j=1}^{p}I(|\beta_j|\neq 0)+\lambda_2\sum_{j\sim j'}|I\left(\frac{|\beta_j|}{w_j}\neq 0\right)-I\left(\frac{|\beta_{j'}|}{w_{j'}}\neq 0\right), \quad (3)$$

where the first penalty is the L_0-loss for sparsest variable selection and unbiased parameter estimation (Shen *et al.* 2012), 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 $I(|z| \neq 0)$,

Shen *et al.* (2012) 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 (3) leads to a new penalty with a TLP for variable selection and a TLP-based penalty for grouping of indicators, shortened as *TTLP*$_I$:

$$P(\beta)=\lambda_1\sum_{j=1}^{p}J_\tau(|\beta_j|)+\lambda_2\sum_{j\sim j'}|J_\tau\left(\frac{|\beta_j|}{w_j}\right)-J_\tau\left(\frac{|\beta_{j'}|}{w_{j'}}\right), \quad (4)$$

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 (3), they may play an important role in (4) (and other penalties shown earlier), as to be shown later.

For any given (λ_1, λ_2, τ), we present a difference convex (DC) programming algorithm to minimize $S(\beta)$ with the new penalty. First, we decompose the non-convex function $J_\tau(|z|)$ in (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), we construct another DC decomposition: $|f_1 - f_2| = 2\max(f_1, f_2) - (f_1 + f_2)$ for two convex functions f_1 and f_2. After applying these two DC decompositions to (4), we have

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

where $u_{j,j'}=\frac{|\beta_j|}{w_j}+\max(\frac{|\beta_{j'}|}{w_{j'}}-\tau,0)$ and $v_{j,j'}=\frac{|\beta_{j'}|}{w_{j'}}+\max(\frac{|\beta_j|}{w_j}-\tau,0)$ are defined to simplify notation. Then, (4) can be rewritten as a difference of two convex functions P_1 and P_2,

$$\begin{aligned}P(\beta)&=P_1(\beta)-P_2(\beta),\\ P_1(\beta)&=\tfrac{1}{\tau}\left(\lambda_1\sum_{j=1}^{p}|\beta_j|+\lambda_2\sum_{j'\sim j}2\max(u_{j,j'},v_{j,j'})\right),\\ P_2(\beta)&=\tfrac{1}{\tau}\left(\lambda_1\sum_{j=1}^{p}\max(|\beta_j|-\tau,0)+\lambda_2\sum_{j'\sim j}(u_{j,j'}+v_{j,j'})\right).\end{aligned}$$

Linearizing P_2 at a current estimate $\hat{\beta}^{(m-1)}$ and ignoring terms independent of β, we obtain a convex approximation of $S(\beta)$:

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

which is minimized to obtain an updated estimate $\hat{\beta}^{(m)}$. Since $S^{(m)}(\beta)$ is convex, we use Matlab package CVX (Grant and Boyd 2011) to minimize it. The DC 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 $S(\hat{\beta}^{(m-1)})$-$S(\hat{\beta}^{(m)}) \le \varepsilon$ with a small tolerance ε (e.g. 10^{-4} used throughout); otherwise, return to [A2].

We have the following convergence result with its proof given in Web Appendix B.

Theorem 1—The sequence $S(\hat{\beta}^{(m)})$ decreases strictly in m unless $\hat{\beta}^{(m)} = \hat{\beta}^{(m-1)}$. In addition, the DC algorithm terminates in finite steps; that is, there exists $m^* < \infty$ such that $\hat{\beta}^{(m)} = \hat{\beta}^{(m*-1)}$ for all $m \ge m^*$. Finally, $\hat{\beta}^{(m*)}$ is a local minimizer of (2) with penalty (4).

We used the Lasso estimate $\hat{\beta}_{lasso}$ as the initial value $\hat{\beta}^{(0)}$ in step [A1]. The three tuning parameters (δ_1, δ_2, τ) 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 $\left[t, \frac{pt}{4}\right]$ for δ_1, $[t, tg]$ for δ_2, and $\left[10^{-6}, \frac{t}{2}\right]$ for τ.

The TLP has been shown to perform well for accurate variable selection and almost unbiased parameter estimation for sparse models (Shen *et al.* 2012). 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), which might not be optimal. Thus, rather than using the TLP for variable selection in (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 $LTLP_I$:

$$P(\beta)=\lambda_1\sum_{j=1}^{p}|\beta_j|+\lambda_2\sum_{j\sim j'}|J_\tau\left(\frac{|\beta_j|}{w_j}\right)-J_\tau\left(\frac{|\beta_{j'}|}{w_{j'}}\right). \quad (5)$$

The computational algorithm and its convergence properties are similar to that for $TTLP_I$. In particular, the intermediate objective function $S^{(m)}(\beta)$ is the same as before except that its second term $\frac{\lambda_1}{\tau}\sum_{j=1}^{p}|\beta_j|I(|\hat{\beta}_j^{(m-1)}|\le\tau)$ is replaced by $\frac{\lambda_1}{\tau}\sum_{j=1}^{p}|\beta_j|$.

The tuning parameters (λ_1, λ_2, τ) 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.

3. Simulations

3.1 Simulation set-ups

Our simulation set-ups are similar to those in Li and Li (2008) and Pan *et al.* (2010). Briefly, the responses Y were generated from linear model (1) with iid error $\varepsilon_i\sim N(0,\sum_j\beta_j^2/2)$. 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 respectively: in simulation I, the (weighted) magnitudes of the non-zero β_j's were close to each other, while in simulation II they were completely random. Specifically, in set-up 1 of simulation I, we had

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

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 setup 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 ~ 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 the first set-up of simulation II, we had

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

where we randomly drew β_2, …, β_{11} ~ Unif(0, 3), and β_{13}, …, β_{22} ~ Unif(-3, 3); then we flipped the signs of β_7, …, β_{11}. Specifically the generated regression coefficient vector β was:

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

In set-up 2, we used a true β similar to that of set-up 1 except that five target genes in each of the first two subnetworks were randomly selected to have their corresponding $\beta_j = 0$, mimicking a setting with $I(\beta_j \neq 0) = I(\beta_{j'} \neq 0)$ for some, but not all, gene pairs $j \sim j'$. A sparser true model was used in set-up 3, in which only the TF and its five randomly selected genes in the first subnetwork had non-zero β_j's while all others were zero.

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 the response on the tuning data. To evaluate the performance, the model error (ME) and the prediction error (PE) were calculated: $\mathrm{ME}=(\beta-\hat{\beta})'E(X'X)(\beta-\hat{\beta})$ with $E(X'X)$ as the population covariance matrix of X (since $E(X) = 0$), and $\mathrm{PE}=\sum_{i=1}^{m}(Y_i-\hat{Y}_i)^2/m$ (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 Li and Li (2010), unlike in Li and Li (2008) and Pan *et al.* (2010), 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.

3.2 Main results

Simulation results for simulation I are summarized in Table 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 β_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 2), our new methods showed more substantial advantages over other methods, in terms of both parameter estimation (and outcome prediction) and variable selection. With a sparse true model in set-up 3, the $TTLP_I$ method performed best.

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 (Luo *et al.* 2012), due to the over-shrinkage and thus large biases of its parameter estimates (Table 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 3); for more sparse true models, we did observe that $TTLP_I$ performed better than $LTLP_I$ for both parameter estimation and variable selection, as shown in set-up 3 in Table 2. Overall, our methods gave less biased estimates than other network-based methods (Table 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.

3.3 Other results

As shown in Web Appendix B, the DC algorithm typically converged in about 3 iterations for simulated data, in agreement with Theorem 1. Since our penalties are not convex, the local minimum the DC algirthm converges to depends on the starting values. In general, better starting values give better results. For example, for set-up 1 of Simulation I as shown in Web Appendix B, since the Enet estimates performed only slightly better than the Lasso estimates with smaller MEs, the $TTLP_I$ and $LTLP_I$ estimates with the Enet estimates as the starting values gave also slightly smaller MEs than those with the Lasso estimates as the starting values; with weight $w = \sqrt{d}$, since the L_∞ method performed much better than the Lasso method, the $TTLP_I$ and $LTLP_I$ estimates with the former as their starting values yielded much smaller MEs than those with the latter. More studies are warranted on this topic.

4. Example

We applied the methods to two breast cancer gene expression datasets, named the Wang data (Wang *et al*. 2005) and van de Vijver data (van de Vijver *et al*. 2002) respectively. The two datasets contained 286 and 295 patients respectively, out of which 106 and 78 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 (Chuang *et al*. 2007), which formed our prior gene network with 294 genes (nodes) and 326 edges. Among the 294 genes 18 were breast cancer (BC) genes from the 60-gene list given in (Chuang *et al*. 2007). Since all the methods were developed for linear regression, we fitted linear models to the binary metastasis status as the response. Each full 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. We first assessed marginal association between the binary metastasis status and each gene' expression level by fitting marginal linear regression models. For TP53 with the van de Vijver data, its regression coefficient estimate was −0.0189, while those for its neighbors ranged from −0.1123 to 0.1488 with the three quartiles as −0.0177, 0.0079 and 0.0377 respectively; similar results were obtained for the three tumor suppressor genes and their neighboring genes on each dataset as shown in Web Appendix C. These results suggest that the association strength of a hub gene like TP53 could be quite different from its direct neighbors. In particular, as pointed out by (Chuang *et al*. 2007), the expression changes of some cancer-causing genes like TP53 might be much weaker than some downstream ones.

Since the sample size was relatively small, we ran each method 20 times on each dataset. In each of 20 runs, each dataset was randomly split into the training, tuning, and test sets with almost equal sample sizes (i.e., with 95, 95, 96 observations for the Wang data, and 95, 95, 105 for the van de Vijver data). We compared the methods' performance in PE, selection of the breast cancer (BC) genes and model size, all averaged over 20 runs. For each method, we used 5-fold cross-validation to select the tuning parameter values by minimizing the PRSS and then used the selected tuning parameters 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 (Chuang *et al.* 2007), 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.

For the Wang data, as shown in Table 4, averaged over the 20 runs, our proposed $TTLP_I$ selected most BC genes at 2.90, followed by the $LTLP_I$ and aGrace at 1.35 and 1.30 respectively, all much higher than those from other methods, though our two methods tended to select slightly larger models with slightly larger PEs than those of the other methods. It is interesting to note that our two new methods and aGrace selected the three hub genes, BRCA1, BRCA2 and TP53, most frequently over the 20 runs. Furthermore, our two new methods were the only ones selecting all three hub genes (and another BC gene) in their final models.

For the van de Vijver data (Table 5), compared to the Wang data, although all the methods tended to select slightly smaller models, we reached the same conclusion. As before, our proposed $TTLP_I$ selected most BC genes, including the three hub genes, both across the 20 runs and in the final model, though $LTLP_I$ and aGrace also performed well. In particular, in agreement with the biological knowledge and with the results from the Wang data, again our two methods $TTLP_I$ and $LTLP_I$ along with aGrace selected the three tumor suppressor genes more often than any other method. Also note that the fitted final models from our two new methods selected almost the same set of the genes with almost equal regression coefficient estimates for each dataset. Figure 1 shows the selected genes in the final models for $TTLP_I$.

Since the Wang data also contained the time to metastasis, possibly right censored, we fitted penalized linear regression models to approximate penalized proportional hazards models. As shown in Web Appendix C, we reached similar conclusions: our proposed methods were more likely to select the three tumor hub genes and other BC genes. We also applied the methods to an RNA-seq gene expression dataset for breast cancer tumor and normal samples as shown in Web Appendix C.

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 (Li and Li 2008, Li and Li 2010, Pan *et al*. 2010), 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, our 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 our 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. We also note that our proposed methods have broad applications beyond microarray gene expression data: since no assumption is imposed on the distribution of the predictors, we can equally apply our methods (as other penalized regression methods) to risk prediction and phenotype modeling with RNA-seq, DNA sequence and other high-throughput genomic data, for example. 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 encourages indirectly, through shrinkage and soft-thresholding, simultaneous elimination, but not necessarily simultaneous selection. Consider a simple case with two orthogonal covariates and $\beta = (\beta_1, \beta_2)'$: with a group Lasso penalty (i.e. the L_2-norm), as shown in (Yuan and Lin 2006), the penalized estimate is $\hat{\beta}=(1 - \sqrt{2}\lambda/\|\tilde{\beta}\|)_+\tilde{\beta}$, where $\tilde{\beta}$ is the OLSE. While the simultaneous elimination effect of the group Lasso is clear with the soft-thresholding, the shrinkage effect is also persistent: we always have $|\hat{\beta}_j| \leq |\tilde{\beta}_j|$ for any component j. Hence, if $\tilde{\beta}_1 = 0$ (or close to 0), no matter how large is $\tilde{\beta}_2 \neq 0$, we will *never* have $\hat{\beta}_1 \neq 0$ (or larger than $\tilde{\beta}_1$); in contrast, with our proposed penalty, it is possible to have $\hat{\beta}_1 \neq 0$ (or larger than

$\tilde{\beta}_1$), in which sense we say that our methods can perform simultaneous selection (while the group Lasso cannot). 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 (Pan 2009, Shen and Huang 2010); 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 (Tibshirani *et al.* 2005), 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 to be solved. 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 and further investigating the properties of our proposed methods are worthwhile for future study.

Matlab code will be posted at http://www.biostat.umn.edu/~weip/prog.html.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

we than the reviewers and editors for helpful and constructive comments. This research was supported by NIH grants R01GM081535, R01HL65462 and R01HL105397.

References

Alfarano, et al. The Biomolecular Interaction Network Database and related tools 2005 update. Nucleic Acids Res Database Issue. 2000; 33:D418–D424.

Ashburner M, et al. Gene ontology: tool for the unification of biology The Gene Ontology Consortium. Nature Genetics. 2000; 25:25–29. [PubMed: 10802651]

Bondell HD, Reich BJ. Simultaneous regression shrinkage, feature selection and supervised clustering of predictors with OSCAR. Biometrics. 2008; 64:115–123. [PubMed: 17608783]

Chuang HY, Lee EJ, YT, Lee DH, Ideker T. Network-based classification of breast cancer metastasis. Molecular Systems Biology. 2007; 3:140. [PubMed: 17940530]

Efron B, Hastie T, Johnstone I, Tibshirani R. Least Angle Regression (with discussion). Annals of Statistics. 2011; 32:407–499.

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

Grant, M.; Boyd, S. CVX: Matlab software for disciplined convex programming, version 1.21. 2011. http://cvxr.com/cvx

Huang Y, Wang P. Network based prediction model for genomic data analysis. Statistics in Biosciences. 2012; 4:47–65.

Kanehisa M, Goto S. KEGG: Kyoto encyclopedia of genes and genomes. Nucleic Acids Res. 2000; 28:27–30. [PubMed: 10592173]

Li C, Li H. Network-constrained regularization and variable selection for analysis of genomic data. Bioinformatics. 2008; 24:1175–1118. [PubMed: 18310618]

Li C, Li H. Variable Selection and Regression Analysis for Graph-Structured Covariates with an Application to Genomics. Annals of Applied Statistics. 2010; 4:1498–1516. [PubMed: 22916087]

Li F, Zhang NZ. Bayesian variable selection in structured high-dimensional covariate spaces with applications in genomics. JASA. 2010; 105:1202–1214.

Luo C, Pan W, Shen X. A Two-Step Penalized Regression Method with Networked Predictors. Statistics in Biosciences. 2012; 4:27–46. [PubMed: 23795219]

Monni, S.; Li, H. Bayesian methods for network-structured genomics data. In: Chen, Ming-Hui; Dey, Dipak K.; Mueller, Peter; Sun, Dongchu; Ye, Keying, editors. Frontiers of Statistical Decision Making and Bayesian Analysis In Honor of James O Berger. New York: Springer; 2010.

Pan W. Network-based multiple locus linkage analysis of expression traits. Bioinformatics. 2009; 25(11):1390–1396. [PubMed: 19336446]

Pan W, Xie B, Shen X. Incorporating Predictor Network in Penalized Regression with Application to Microarray Data. Biometrics. 2010; 66(2):474–484. [PubMed: 19645699]

Percival D, Roeder K, Rosenfeld R, Wasserman L. Structured, sparse regression with application to HIV drug resistance. Annals of Applied Statistics. 2011; 5(2A):628–644. [PubMed: 21892380]

Peri, et al. Human protein reference database as a discovery resource for proteomics. Nucleic Acids Res Database Issue. 2004; 32:D497–D501.

Shen X, Huang HC. Grouping pursuit through a regularization solution surface. Journal of the American Statistical Association. 2010; 105:727–739. [PubMed: 20689721]

Shen X, Pan W, Zhu Y. Likelihood-based selection and sharp parameter estimation. Journal of the American Statistical Association. 2012; 107:223–232. [PubMed: 22736876]

Tai, F.; Pan, W.; Shen, X. Bayesian Variable Selection in Regression with Networked Predictors. In: Cai, T.; Shen, X., editors. Analysis of High Dimensional Data. Higher Education Press; Beijing, China: 2010. p. 147-165.

Tibshirani R. Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society. 1996; 58:267–288.

Tibshirani R, Saunders M, Rosset S, Zhu J, Knight K. Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society. 2005; 67:91–108.

van de Vijver MJ, He YD, van't Veer LJ, Dai H, Hart AA, Voskuil DW, Schreiber GJ, Peterse JL, Roberts C, Marton MJ, Parrish M, Atsma D, Witteveen A, Glas A, Delahaye L, van der Velde T, Bartelink H, Rodenhuis S, Rutgers ET, Friend SH, et al. A gene-expression signature as a predictor of survival in breast cancer. N Engl J Med. 2002; 347:1999–2009. [PubMed: 12490681]

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, Foekens JA. Gene-expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. Lancet. 2005; 365:671–679. [PubMed: 15721472]

Yuan M, Lin Y. Model selection and estimation in regression with grouped variables. J Royal Statist Soc B. 2006; 68:49–67.

Zhao P, Rocha G, Yu B. The composite absolute penalties family for grouped and hierarchical variable selection. Ann Statist. 2009; 37:3468–3497.

Zhu Y, Shen X, Pan W. Support Vector Machines with Disease-gene-centric Network Penalty for High Dimensional Microarray Data. Stat Interface. 2009; 2(3):257–269. [PubMed: 20401316]

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

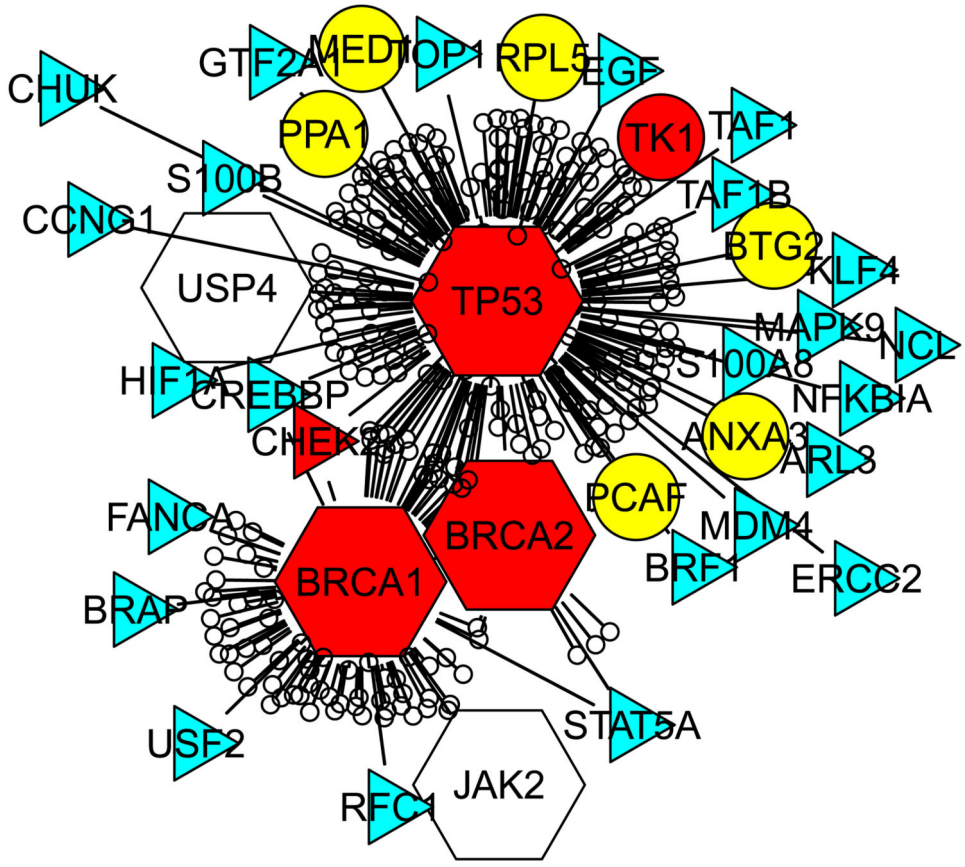

Figure 1.

The genes selected in the final models by *TTLP*$_I$ with the Wang data and the van de Vijver data in the used PPI subnetwork: the 5 genes (in hexagons) appearing in both models, 25 genes (in triangles) only in the model with the Wang data, and 7 genes (in large circiles) only in the model with the van de Vijver data; the 5 BC genes are BRCA1, BRCA2, TP53, TK1 and CHEK2.

Table 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)

Set-up	*w*	Method	ME(sd)	PE(sd)	*TP*	*FP*	*TF*
		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 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 for each set-up. The true numbers of TP are 22, 12 and 6, and the true numbers of TFs are 2, 2 and 1 for the three set-ups respectively.

Set-up	*w*	Method	ME(sd)	PE(sd)	*TP*	*FP*	*TF*
1	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)
2	1	Lasso	15.7(4.7)	33.6(5.9)	6.2[6](1.6)	14.2[11](13.7)	0.5[0.5](0.6)
		Enet	16.1(4.5)	34.3(5.8)	7.0[7](2.1)	20.6[16](17.1)	0.8[1.0](0.7)
		Grace	15.8(4.3)	34.0(5.4)	7.3[7](1.9)	20.6[16](16.1)	1.1[1.0](0.8)
		aGrace	15.5(4.0)	33.7(5.5)	7.2[7](2.2)	21.0[16.5](16.9)	0.9[1.0](0.8)
		L_∞	15.1(4.2)	33.2(5.7)	5.8[6](1.5)	12.9[11](10.5)	0.0[0.0](0.3)
		aL_∞	15.1(4.3)	33.3(5.8)	5.8[6](1.4)	12.4[11](10.5)	0.0[0.0](0.3)
		$TTLP_I$	15.3(5.6)	32.3(6.4)	8.5[7](3.2)	19.2[10](23.9)	1.1[1.0](0.8)
		$LTLP_I$	14.9(4.3)	32.9(5.6)	10.8[12](2.2)	46.8[54](24.5)	1.7[2.0](0.6)
	$\sqrt{d}$	Grace	16.1(4.6)	34.1(5.6)	7.1 [7](2.0)	20.8[17](16.7)	0.9[1.0](0.8)
		aGrace	16.7(5.4)	34.0(5.6)	7.3[7](2.5)	24.1[17](20.2)	0.8[1.0](0.8)

Set-up	*w*	Method	ME(sd)	PE(sd)	*TP*	*FP*	*TF*
		L_∞	15.2(4.1)	33.3(5.6)	7.3[7.5](2.2)	17.0[14.5](9.9)	0.6[1.0](0.6)
		aL_∞	17.8(5.6)	35.7(7.0)	7.2[7](2.1)	14.0[12](8.0)	0.6[1.0](0.6)
		$TTLP_I$	15.7(6.5)	32.6(6.8)	8.2[7](3.1)	20.3[11](24.4)	1.1[1.0](0.8)
		$LTLP_I$	15.7(5.5)	33.5(5.8)	10.6[12](2.3)	47.1[46.5](25.0)	1.6[2.0](0.7)
3	1	Lasso	6.0(2.2)	18.2(2.9)	4.0[4](0.8)	11.9[10](11.4)	0.8[1](0.5)
		Enet	6.4(2.3)	18.6(3.3)	4.2[4](0.8)	13.8[12](11.9)	0.8[1](0.5)
		Grace	6.2(2.3)	18.4(3.0)	4.3[4](0.8)	14.9[12](13.1)	0.9[1](0.5)
		aGrace	6.1(2.1)	18.4(2.9)	4.2[4](0.8)	15.1[12](12.5)	0.8[1](0.5)
		L_∞	6.5(2.1)	18.3(3.0)	3.5[3](0.9)	11.6[10](7.8)	0.1[0](0.3)
		aL_∞	6.6(2.1)	18.5(3.1)	3.5[3](0.9)	10.6[9](7.1)	0.1[0](0.3)
		$TTLP_I$	4.8(2.1)	16.1(2.8)	5.0[6](1.2)	8.5[5](12.7)	0.8[1](0.5)
		$LTLP_I$	6.5(2.3)	18.4(2.9)	5.5[6](0.9)	41.9[38](26.1)	1.2[1](0.6)
	$\sqrt{d}$	Grace	6.2(2.4)	18.5(3.0)	4.2[4](0.8)	14.0[12](12.7)	0.8[1](0.5)
		aGrace	7.4(4.2)	18.8(3.0)	4.5[4](1.0)	24.1[13](25.6)	0.9[1](0.6)
		L_∞	6.1(2.1)	18.1(2.9)	5.2[6](1.3)	15.7[13](9.7)	0.8[1](0.5)
		aL_∞	8.0(3.1)	20.1(3.9)	5.2[6](1.3)	13.3[12](8.4)	0.8[1](0.5)
		$TTLP_I$	4.7(2.2)	16.0(2.7)	4.9[6](1.3)	10.2[5](15.9)	0.7[1](0.5)
		$LTLP_I$	6.1(2.2)	18.2(2.8)	5.6[6](0.8)	45.0[46.5](27.6)	1.3[1](0.6)

Table 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

			β_1=5			β_2=1.58			β_{11}=1.58		
Set-up	***w***	**Methods**	**Mean**	**sd**	**MSE**	**Mean**	**sd**	**MSE**	**Mean**	**sd**	**MSE**
	$\sqrt{d}$	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

Results for the Wang data with w = $\sqrt{d}$:PE (se), mean [median] (se) of the frequencies of selecting breast cancer (# BC) genes and all genes (# Genes) over 20 runs and in the final models. The frequencies of selecting the 3 tumor suppressor genes (BRCA1/BRCA2/TP53) and other genes selected ≥ 5 times are given. The BC genes are underlined.

Method	PE	# BC	# Genes
Lasso	0.235(0.004)	0.30[0.00](0.13)	8.80[8.00](1.91)
Final	-	1	30
Enet	0.227(0.003)	0.20[0.00](0.09)	9.90[1.00](2.60)
Final	-	2	51
Grace	0.227(0.003)	0.70[1.00](0.16)	9.50[2.50](2.38)
Final	-	2	49
aGrace	0.229(0.003)	1.30[1.00](0.25)	10.20[6.00](2.10)
Final	-	2	52
$L_i nf$	0.236(0.005)	0.10[0.00](0.07)	10.35[7.50](1.97)
Final	-	0	3
$aL_i nf$	0.239(0.005)	0.10[0.00](0.07)	10.20[7.50](2.43)
Final	-	0	3
TTLP	0.282(0.015)	2.90[3.00](0.34)	12.00[8.00](2.68)
Final	-	4	30
LTLP	0.256(0.009)	1.35[1.50](0.28)	11.10[8.00](2.07)
Final	-	4	30

	Frequency of the 3 tumor genes being selected (Freq) Frequency of other genes selected ≥ 5 times (Freq)
Lasso	BRCA1 (1), BRCA2 (0), TP53 (1) MAPK9 (10), TOP1 (6)
Enet	BRCA1 (0), BRCA2 (0), TP53 (0) ERCC2 (6), MAPK9 (7), TOP1 (6)
Grace	BRCA1 (7), BRCA2 (2), TP53 (2) BRCA1 (7), ERCC2 (5), MAPK9 (7), TOP1(6)
aGrace	BRCA1 (10), BRCA2 (4), TP53 (9)

Method	PE	# BC	# Genes
	ERCC2 (5), MAPK9 (8)		
L_∞	BRCA1 (0), BRCA2 (0), TP53 (0)		
	CD74 (6), HIF1A (5), MAPK9 (12), RFC1 (5), TOP1 (7)		
aL_∞	BRCA1 (0), BRCA2 (0), TP53 (0)		
	HIF1A (5), MAPK9 (11), TOP1 (7), CEBPZ (5)		
$TTLP_I$	BRCA1 (20), BRCA2 (10), TP53 (20),		
	MAPK9 (8), TOP1 (5)		
$LTLP_I$	BRCA1 (9), BRCA2 (5), TP53 (9)		
	HIF1A (5), MAPK9 (9) TOP1 (7)		

Table 5

Results for the van de Vijver data with $w = \sqrt{d}$: $PE\ (se)$, mean [median] (se) of the frequencies of selecting breast cancer (# BC) genes and all genes (# Genes) over 20 runs and in the final models. The frequencies of selecting the 3 tumor suppressor genes (BRCA1/BRCA2/TP53) and other genes selected ≥ 5 times are given. The final model for $TTLP_I$ with the selected genes (their regression coefficient estimates ×100) are listed. The BC genes are underlined.

Method	PE	# BC	# Genes
Lasso	0.192(0.005)	0.70[1](0.16)	7.35[6](0.97)
Final	-	1	10
Enet	0.189(0.004)	0.75[1](0.18)	8.00[6](1.64)
Final	-	1	9
Grace	0.190(0.004)	0.80[1](0.17)	7.90[5](1.65)
Final	-	1	9
aGrace	0.189(0.003)	1.25[1](0.18)	8.20[7](1.61)
Final	-	1	9
L_∞	0.193(0.006)	0.65[1](0.13)	7.10[6](0.96)
Final	-	1	2
aL_∞	0.193(0.006)	0.65[1](0.13)	7.15[6](0.97)
Final	-	1	2
$TTLP_I$	0.217(0.008)	1.50[1](0.26)	4.90[4](0.44)
Final	-	4	12
$LTLP_I$	0.203(0.005)	1.00[1](0.21)	6.45[4](1.15)
Final	-	4	13

	Frequency of the 3 tumor genes being selected (Freq) Frequency of other genes selected ≥ 5 times (Freq)
Lasso	BRCA1 (1), BRCA2 (0), TP53 (0) E2F1 (6), JAK2 (7), STAT5A (5), TK1 (8), USP4 (6)
Enet	BRCA1 (1), BRCA2 (0), TP53 (0) E2F1 (5), JAK2 (5), TK1 (8), PCAF (7)
Grace	BRCA1 (2), BRCA2 (0), TP53 (1) E2F1 (5), TK1 (7), PCAF(7)

Method	PE	# BC	# Genes
aGrace	BRCA1 (5), BRCA2 (3), TP53 (5)		
	E2F1 (5), TK1 (6), PCAF (6)		
L_∞	BRCA1 (0), BRCA2 (0), TP53 (0)		
	GSK3B (5), JAK2 (7), RPL5 (5), TK1 (10), USP4 (7), BTG2 (5)		
aL_∞	BRCA1 (0), BRCA2 (0), TP53 (0)		
	GSK3B (5), JAK2 (7), RPL5 (5), TK1 (10), USP4(7)		
$TTLP_I$	BRCA1 (8), BRCA2 (2), TP53 (11),		
	E2F1 (5), TK1 (7), USP4 (5)		
$LTLP_I$	BRCA1 (5), BRCA2 (0), TP53 (5)		
	E2F1 (5), JAK2 (6), TK1 (8), USP4(6)		
	Final model: selected genes ($\hat{\beta}_j \times 100$)		
$TTLP_I$	ANXA3 (-5.88), BRCA1 (-4.53), BRCA2 (-1.70), JAK2 (-3.52),		
	PPA1 (-6.83), MED1 (3.91), RPL5 (-1.51), TK1 (1.94)		
	TP53 (-7.02), USP4 (-9.75), BTG2 (-3.13), PCAF (4.86)		