TI 2014-089/I
Tinbergen Institute Discussion Paper

Regularized Regression Incorporating Network Information: Simultaneous Estimation of Covariate Coefficients and Connection Signs

Matthias Weber[1]
Martin Schumacher[2]
Harald Binder[3]

[1] *Faculty of Economics and Business, University of Amsterdam, and Tinbergen Institute, the Netherlands;*
[2] *University Medical Center, Freiburg, Germany;*
[3] *University Medical Center, Mainz, Germany.*

Tinbergen Institute is the graduate school and research institute in economics of Erasmus University Rotterdam, the University of Amsterdam and VU University Amsterdam.

More TI discussion papers can be downloaded at http://www.tinbergen.nl

Tinbergen Institute has two locations:

Tinbergen Institute Amsterdam
Gustav Mahlerplein 117
1082 MS Amsterdam
The Netherlands
Tel.: +31(0)20 525 1600

Tinbergen Institute Rotterdam
Burg. Oudlaan 50
3062 PA Rotterdam
The Netherlands
Tel.: +31(0)10 408 8900
Fax: +31(0)10 408 9031

Duisenberg school of finance is a collaboration of the Dutch financial sector and universities, with the ambition to support innovative research and offer top quality academic education in core areas of finance.

DSF research papers can be downloaded at: http://www.dsf.nl/

Duisenberg school of finance
Gustav Mahlerplein 117
1082 MS Amsterdam
The Netherlands
Tel.: +31(0)20 525 8579

Regularized Regression Incorporating Network Information: Simultaneous Estimation of Covariate Coefficients and Connection Signs*

Matthias Weber[†] Martin Schumacher[‡] Harald Binder[§]

June 28, 2014

Abstract

We develop an algorithm that incorporates network information into regression settings. It simultaneously estimates the covariate coefficients and the signs of the network connections (i.e. whether the connections are of an activating or of a repressing type). For the coefficient estimation steps an additional penalty is set on top of the lasso penalty, similarly to Li and Li (2008). We develop a fast implementation for the new method based on coordinate descent. Furthermore, we show how the new methods can be applied to time-to-event data. The new method yields good results in simulation studies concerning sensitivity and specificity of non-zero covariate coefficients, estimation of network connection signs, and prediction performance. We also apply the new method to two microarray time-to-event data sets from patients with ovarian cancer and diffuse large B-cell lymphoma. The new method performs very well in both cases. The main application of this new method is of biomedical nature, but it may also be useful in other fields where network data is available.

Keywords: high-dimensional data; gene expression data; pathway information; penalized regression

*Thanks for comments and suggestions go to Jochen Knaus, Christine Porzelius, and participants of seminars at the University of Freiburg.

[†]University of Amsterdam (CREED), and Tinbergen Institute

[‡]Institute of Medical Biometry and Medical Informatics, University Medical Center Freiburg

[§]Institute of Medical Biostatistics, Epidemiology and Informatics, University Medical Center Johannes Gutenberg University Mainz

1 Introduction

The last decades have seen a surge of availability of data containing network information. Statistical techniques that deal with this network data have started to evolve, but are still in their early development. One of the main concerns of such models is to incorporate network information into regression settings to lead to better estimations and better prediction performance. In many cases it is also important to have statistical tools that give information about the nature of the network. In this paper, we develop an algorithm that incorporates network information into regression settings. It simultaneously estimates covariate coefficients and the signs of the network connections (i.e. whether the connections are of an activating or of a repressing type). The main motivation and application of this method is of biomedical nature, but it might also be useful to deal with other data concerning network information, such as for example web data or data on social and economic networks.

The evolution of biotechnological knowledge has made it possible to extract large amounts of data, e.g. gene expression data, from patients suffering from all kinds of diseases. The nature of these data poses new challenges to the regression techniques applied. One of these challenges is that the data are high-dimensional, which means that the number of covariates is much larger than the number of observations – the number of covariates easily reaches a few thousands, whereas the number of observations is much lower. In the last decades, regression techniques have been developed to deal with this kind of data, such as for example the lasso (Tibshirani, 1996), the SCAD (Fan and Li, 2001) or the elastic net (Zou and Hastie, 2005). By now, not only the gene expression data itself is available, but also a lot of biological knowledge about regulatory relationships between genes or gene products, for example knowledge about pathways. This external knowledge about the regulatory relationships is usually given through networks that are represented by graphs. It has been shown already that good results can be achieved when this external knowledge is incorporated into statistical testing (see for example Wei and Pan, 2008). Although many regression techniques have been developed that can deal with high-dimensional data hardly any of them is able to incorporate knowledge about pathways and regulatory networks (exceptions are e.g. Li and Li, 2008, and Binder and Schumacher, 2009).

Li and Li (2008) develop a penalty that makes use of external network information. One problem of their approach is that they assume that all connections within the pathways are connections where genes and gene products are enforced or triggered by one another. This does not always have to be the case. There can also be connections where genes are suppressed by other genes, for example transcription factors might

exist that up-regulate some genes and down-regulate others. We extend their approach, so that it is possible to consider both, activating (positive) and repressing (negative) connections.

Unfortunately, there is only very little knowledge about the signs of the connections in the databases so far. Thus, also a procedure to estimate the connection signs is needed. Having such a procedure is not only a tool to improve prediction performance, it is also a goal in its own. If the procedure performs well, it can bring about additional biomedical knowledge – knowledge about the nature of the connections between genes in a network.

The algorithm we develop estimates covariate coefficients and connection signs simultaneously. Computational efficiency is often crucial when dealing with high-dimensional data. We develop computationally efficient implementations for these procedures. Furthermore, we adapt the methods to the Cox proportional hazards model to be able to handle time-to-event data appropriately.

This paper is organized as follows. Section 2 contains the methods we use in this paper and their implementations. Section 3 contains simulation results. In Section 4 we apply the new method to time-to-event microarray data from patients with ovarian cancer and from patients with large B-cell lymphoma. Section 5 concludes.

2 Methods

Later on, we will consider time-to-event data. Until then, we consider a data set (y,X) with a continuous response y and covariate matrix X (also called input matrix). The number of observations is n and the number of covariates is p. Thus, $y = (y_1,...,y_n)^T$ is an n-vector and X is an $n \times p$-matrix with rows $x_i^T = (x_{i1},...,x_{ip})$. $x_{(j)}$ denotes the j-th column of the input matrix. For simplicity we assume y to be centered and X to be standardized, which means that $\sum_{i=1}^n y_i = 0$, $\sum_{i=1}^n x_{ij} = 0$ and $\frac{1}{n}\sum_{i=1}^n x_{ij}^2 = 1$ for $j = 1,...,p$. The vector of coefficients to be estimated is $\beta = (\beta_1,...,\beta_p)$, linking X and y.[1]

[1] In the classical linear model $y_i = x_i^T\beta + \varepsilon_i$, with $\varepsilon_i \sim N(0,\sigma^2)$. β is usually estimated by the maximum likelihood estimator $\hat{\beta}$; maximizing the log-likelihood is equivalent to minimizing the residual sum of squares.

2.1 Incorporating network information with only positive connection signs (Li and Li, 2008)

In problems involving gene expression data, graphs of networks depicting biological information are often given. To make use of this additional information when fitting a regression model, Li and Li (2008) introduce an additional penalty, similar in spirit to the fused lasso (Tibshirani et al., 2005), which is set on top of the lasso penalty. Additional to the aim of improving prediction performance, another aim of this method is to get well interpretable models and to identify pathways involved in diseases and other biological processes. The additional information for the regression problem is given by a regulatory network depicted by a weighted graph. In the graph, the vertices represent genes (or gene products), which are the covariates, and the edges indicate some regulatory relationship between the covariates. The regulatory network is incorporated into the network penalty via the normalized Laplace matrix of the associated graph. This is a $p \times p$ -matrix defined as in Chung (1997) by

$$(L)_{uv} = \begin{cases} 1 - \frac{w(u,v)}{d_u} & \text{if } u = v \text{ and } d_u \neq 0 \\ \frac{-w(u,v)}{\sqrt{d_u d_v}} & \text{if } u \text{ and } v \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

where u and v stand for the u-th and v-th covariate and $w(u,v)$ denotes the weight of the edge that links the u-th and v-th covariate. Often the information is given via a connection matrix which consists only of zeros and ones indicating only which genes are connected, thus $w(u,v)$ is usually zero or one. d_u is the degree of vertex u defined as the sum of $w(u,v)$ over all vertices v that are linked to vertex u. When the normalized Laplacian is used to describe biological network information, there cannot be any loops, i.e. $w(u,u)$ is always zero. The matrix L is symmetric and positive semi-definite (see Chung, 1997).

The network-constrained model-fitting procedure Li and Li (2008) propose is to choose the $\hat{\beta}$ that minimizes the penalized residual sum of squares

$$RSS(\lambda_1, \lambda_2, \beta) = (y - X\beta)^T (y - X\beta) + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \beta^T L \beta. \tag{1}$$

Thus the first part of the penalty is just the lasso penalty. λ_1 and λ_2 are chosen via 10-fold cross-validation (or any other model complexity selection procedure).

The set of all edges is denoted by $\{u \sim v\}$, indicating that the predictors u and v are linked on the network directly. For the normalized Laplacian L of a graph without

loops as defined above and $\beta \in \mathbb{R}^p$ it is then $\beta^T L \beta = \sum_{u \sim v} \left(\frac{\beta_u}{\sqrt{d_u}} - \frac{\beta_v}{\sqrt{d_v}} \right)^2 w(u,v)$.[2]

2.2 A new algorithm to estimate covariate coefficients and network connection signs simultaneously

Li and Li (2008) implicitly assume that all connections between the connected covariates are positive, i.e. that they influence the outcome in the same direction. Thus, all the true coefficients of the connected covariates should have the same sign. It is likely, though, that some of the connections have a negative sign, for example if a transcription factor suppresses another gene. In this case it would be suitable to add a penalty of the form $\lambda_2 \left(\frac{\beta_u}{\sqrt{d_u}} + \frac{\beta_v}{\sqrt{d_v}} \right)^2 w(u,v)$ rather than $\lambda_2 \left(\frac{\beta_u}{\sqrt{d_u}} - \frac{\beta_v}{\sqrt{d_v}} \right)^2 w(u,v)$. Usually, the signs of the connections are not known ex ante. Therefore, we develop an algorithm that estimates the covariate coefficients and the connection signs simultaneously (straightforward maximization of the penalized log likelihood over all coefficients and connection signs is computationally unfeasible for high-dimensional data). On top of improving upon the estimation of the regression coefficients, gaining knowledge on the signs of the coefficients can already be seen as a goal on its own.

In the algorithm we develop there will be two steps repeated after one another. In the coefficient (or parameter) estimation step, the covariate coefficients are estimated by maximizing the penalized log-likelihood given estimates of the connection signs.[3] Then, in a connection sign estimation step, the signs of the connections are estimated, taken the covariate coefficients as given from the last parameter estimation step. These two steps are then repeated (the algorithm is thus similar in spirit to an EM-algorithm; see McLachlan and Krishan, 1997).

The implementation Li and Li (2008) use for their problem is computationally very expensive.[4] We develop a much more efficient implementation via coordinate descent.

[2] Minimizing the penalized residual sum of squares (1) is then equivalent to maximizing the penalized log-likelihood

$$l(\lambda_1, \lambda_2, \beta) = \ell(\beta) - \lambda_1 \sum_{j=1}^{p} |\beta_j| - \lambda_2 \sum_{u \sim v} \left(\frac{\beta_u}{\sqrt{d_u}} - \frac{\beta_v}{\sqrt{d_v}} \right)^2 w(u,v)$$

for λ_1 and λ_2 adjusted by a factor of $\frac{1}{2\sigma^2}$, which we omit ($\ell(\beta)$ denotes the log-likelihood of the classical linear model).

[3] Here, we use the additional penalty on top of the L_1 penalty to obtain sparse solutions (similarly to Li and Li, 2008). The algorithm we develop also works well in settings were non-sparse solutions are wanted – the additional penalty can be used together with the L_2 instead of the L_1 penalty. This can be seen in Weber (2009), which is available on request.

[4] They transform their problem into a lasso problem of an augmented data set following Zou and Hastie (2005). With their implementation the problem for fixed λ_1 and λ_2 with n observations and p

2.2.1 Definitions, notation and assumptions

On top of using the normalized Laplacian as penalty matrix (or mutations of it that arise through negative connection signs), we also use the combinatorial Laplacian, which is defined by

$$(L_{comb})_{uv} = \begin{cases} d_u - w(u,u) & \text{if } u = v \text{ and } d_u \neq 0, \\ -w(u,v) & \text{if } u \text{ and } v \text{ are adjacent,} \\ 0 & \text{otherwise} \end{cases}$$

and which leads (with positive connection signs) to a penalized log-likelihood of the form

$$l(\lambda_1, \lambda_2, \beta) = \ell(\beta) - \lambda_1 \sum_{j=1}^{p} |\beta_j| - \lambda_2 \sum_{u \sim v} (\beta_u - \beta_v)^2 w(u,v).$$

This is thus a very natural penalty and indeed Li and Li (2008) write that they would have liked to use this penalty as well, but that they were not able to do so.

If there is a connection between covariate i and covariate j we define

$$\xi_{ij} = \begin{cases} \text{-1} & \text{if the connection has negative sign,} \\ 1 & \text{if the connection has positive sign} \end{cases}$$

and extend the definition immediately to all $i, j \in \{1, ..., p\}$ for convenience by

$$\xi_{ij} = \begin{cases} \text{-1} & \text{if there is a negative connection between covariates } i \text{ and } j, \\ 1 & \text{otherwise.} \end{cases}$$

Given a penalty matrix M with $M_{ij} = 0$ if covariates i and j are not connected, we would actually like to use the penalty matrix $M_{correct}$, which is defined by

$$(M_{correct})_{ij} = -\xi_{ij}|(M)_{ij}| \text{ for } i \neq j \text{ and } (M_{correct})_{ii} = |(M)_{ii}|, \ i, j \in \{1, ..., p\},$$

which is the penalty matrix arising from M when all connection signs are known. In most cases, the ξ_{ij} are unknown and thus $M_{correct}$ as well. In the connection sign estimation step described later, we estimate the ξ_{ij}.

It is (only) in theory possible that one gene suppresses another gene and simultaneously influences the outcome directly in positive direction, while the suppressed gene also influences the outcome directly in positive direction. Then the true coefficients of these covariates would both be positive, whereas their connection is negative. In practice,

covariates is equivalent to solving a lasso problem with $n + p$ observations and p covariates, which is computationally very expensive if p is large.

a case like the above is extremely unrealistic. We always assume that the influence a gene has on the outcome is either only direct influence or influence only through one or more connections or, if both together, that the influences are of a kind that does not give room for the unrealistic problem described above.[5]

Analogous to the meaning of $\{u \sim v\}$ in Li and Li (2008), $\left\{u \overset{+}{\sim} v\right\}$ denotes the set of all connections that we assume to have positive signs, where u and v are the covariates linked by each connection. By $\left\{u \overset{-}{\sim} v\right\}$ we denote the set of connections that we assume to have negative signs, such that we have, with the definition of ξ_{ij} from above and for example the additional penalty term arising from the normalized Laplacian,[6]

$$\sum_{u \overset{+}{\sim} v} \left(\frac{\beta_u}{\sqrt{d_u}} - \frac{\beta_v}{\sqrt{d_v}}\right)^2 w(u,v) + \sum_{u \overset{-}{\sim} v} \left(\frac{\beta_u}{\sqrt{d_u}} + \frac{\beta_v}{\sqrt{d_v}}\right)^2 w(u,v) = \sum_{u \sim v} \left(\frac{\beta_u}{\sqrt{d_u}} - \hat{\xi}_{uv}\frac{\beta_v}{\sqrt{d_v}}\right)^2 w(u,v).$$

$\beta^T L \beta = \sum_{u \overset{+}{\sim} v} \left(\frac{\beta_u}{\sqrt{d_u}} - \frac{\beta_v}{\sqrt{d_v}}\right)^2 w(u,v) + \sum_{u \overset{-}{\sim} v} \left(\frac{\beta_u}{\sqrt{d_u}} + \frac{\beta_v}{\sqrt{d_v}}\right)^2 w(u,v)$. L_{comb} denotes the combinatorial Laplacian or the version arising from the combinatorial Laplacian by changing signs (sometimes also called combinatorial Laplacian).

We assume without loss of generality that any penalty matrix M is symmetric. Furthermore, we assume that M is such that we can write $\beta^T M \beta = \sum_{u \sim v} (a(u,v)\beta_u + b(u,v)\beta_v)^2$, with $a(u,v)$ and $b(u,v)$ real numbers.[7]

2.2.2 The Network Algorithm

Estimation of the covariate coefficients

Given a penalty matrix M and penalty parameters λ_1 and λ_2 we estimate β by maximiz-

[5] One can also use the following alternative definition (this will make the biological interpretability of results more difficult, though). When the unrealistic scenario described above does not appear, the definitions are equivalent: When two connected covariates influence the outcome in the same direction, we say that the connection is positive, when two connected covariates influence the outcome in different directions, we say that the connection is negative.

[6] Usually the network information is given via an unweighted graph which means that the weights $w(u,v)$ are zero or one. We do not want to exclude the possibility of using weights and carry along the weights for the normalized Laplacian, but for the combinatorial Laplacian we always assume that the weights only take values zero (not connected) or one (connected) and thus we get $\beta^T L_{comb} \beta = \sum_{u \overset{+}{\sim} v} (\beta_u - \beta_v)^2 + \sum_{u \overset{-}{\sim} v} (\beta_u + \beta_v)^2$.

[7] With the penalty matrix we want to incorporate network information that is given via a graph and penalize some squared terms, thus this assumption is not a great restriction. This also implies that $(M)_{ij} = 0$ for $i \neq j$ if there is no connection between covariates i and j. Obviously, the normalized Laplacian and the combinatorial Laplacian are of this form.

ing the penalized log-likelihood

$$l(\lambda_1, \lambda_2, \beta) = \ell(\beta) - \lambda_1 \sum_{j=1}^{p} |\beta_j| - \lambda_2 \beta^T M \beta$$

or by minimizing the corresponding residual sum of squares. Solving this extreme value problem in a computationally efficient way (which is important, especially if the number of covariates is large) is not trivial. We develop a fast implementation, which we describe in Section 2.2.3.

Estimation of the connection signs

We estimate the connection signs by directly by considering the influence of the two connected covariates when all other coefficients are fixed. This means that to get an estimation of the sign of the connection between covariates i and j, we keep $\hat{\beta}_k$ fixed for $k \neq i, j$ and then fit a small linear model.

We denote by $X^{-(i,j)}$ the input matrix excluding the colums i and j and by $\hat{\beta}^{-(i,j)}$ the estimated β excluding $\hat{\beta}_i$ and $\hat{\beta}_j$, while $x_{(i)}$ denotes again the i-th column of the input matrix. We consider the new response

$$\tilde{y} = y - X^{-(i,j)} \hat{\beta}^{-(i,j)}$$

and minimize

$$\sum_{k=1}^{n} \left(\tilde{y}_k - X_{ki}\beta_i - X_{kj}\beta_j\right)^2 = \left(\tilde{y} - \left(x_{(i)}, x_{(j)}\right) \begin{pmatrix} \beta_i \\ \beta_j \end{pmatrix}\right)^T \left(\tilde{y} - \left(x_{(i)}, x_{(j)}\right) \begin{pmatrix} \beta_i \\ \beta_j \end{pmatrix}\right) \qquad (2)$$

over $(\beta_i, \beta_j)^T$. We denote the minimizer of the above residual sum of squares by $(\hat{\beta_i}^*, \hat{\beta_j}^*)^T$. If the signs of $\hat{\beta_i}^*$ and $\hat{\beta_j}^*$ are different we estimate the connection between covariates i and j to have negative sign, otherwise we assume it to have positive sign. Thus: If covariates i and j are connected, update $\hat{\xi}_{ij}$ by $\hat{\xi}_{ij} \leftarrow sign(\hat{\beta}_i^*) \cdot sign(\hat{\beta}_j^*)$. In the end of Section 2.2.3 we show a decomposition through which the large number of linear models can be computed very efficiently.

As starting values we propose to use the signs of the empirical covariance of the columns of the input matrix. Because the covariate matrix is standardized, we can simply take the sign of $x_{(i)}^T x_{(j)}$ as starting value for the sign of the connection between covariates i and j, i.e. we assume the connection to be positive, if $x_{(i)}^T x_{(j)} > 0$, and negative, if $x_{(i)}^T x_{(j)} < 0$.

Algorithm

Now we have all the ingredients for the algorithm. We have a procedure to initially estimate the signs of the connections, we have a procedure to minimize the penalized residual sum of squares and we have a procedure to update the estimates of the connection signs. Given a log-likelihood $\ell(\beta)$, a penalty matrix M_1 and fixed penalty parameters, we get the following Network Algorithm:

1. Find starting values for the estimates of the connection signs by employing the empirical covariance, i.e. $\hat{\xi}_{ij} = sign(x_{(i)}^T x_{(j)})$.

2. Estimate β by maximizing the penalized log-likelihood

$$l(\lambda_1, \lambda_2, \beta) = \ell(\beta) - \lambda_1 \sum_{j=1}^{p} |\beta_j| - \lambda_2 \beta^T M \beta,$$

   where $(M)_{ij} = -\hat{\xi}_{ij}|(M_1)_{ij}|$ for $i \neq j$ and $(M)_{ii} = |(M_1)_{ii}|,\ i, j \in \{1, ..., p\}$,

   with the current estimates of the connection signs.

3. Update the estimates of the connection signs by running mini OLS models as described in Section 2.2.2, with the current estimate of β from step 2, so that $\hat{\xi}_{ij} \leftarrow sign(\hat{\beta}_i^*) \cdot sign(\hat{\beta}_j^*)$.

4. Iterate steps 2 and 3 until convergence (which means that the $\hat{\xi}_{ij}$ do not change anymore once for all connections) or for a fixed number of repetitions.

We cannot guarantee that this algorithm converges. However, even when it does not converge it can be run for a certain number of repetitions and then be stopped as it is very unlikely that a large number of connections have still changing signs and that their coefficients are important. It is much more likely that the few connections with still changing signs have coefficients that are estimated to be zero. The failure to converge that can occur in some circumstances is thus not necessarily problematic.

2.2.3 Implementation and convergence results

As noted above, the implementation of our algorithm is not trivial. Here, we first show a computationally efficient way to find the minimum of the penalized residual sum of squares (in the coefficient estimation step) via a coordinate descent algorithm (and covariate updates). In order to make this comprehensible we first review coordinate

descent and and covariance updates very briefly for the lasso. Afterwards, we prove that the coordinate descent algorithm in the coefficient estimation step always converges to the global minimum. In the end, we show how we improved the efficiency of the computation in the connection sign estimation step by decomposing the terms and storing and accessing certain parts.

Coordinate descent and covariate updates for the lasso

Given a log-likelihood $\ell(\beta)$ and a penalty parameter $\lambda \geq 0$, the solution to a lasso problem (Tibshirani, 1996) maximizes the penalized log likelihood $l(\lambda,\beta) = \ell(\beta) - \lambda \sum_{j=1}^{p} |\beta_j|$ over β. In the classical linear model, this is equivalent to minimizing the following residual sum of squares:

$$RSS(\lambda,\beta) = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| = (y - X\beta)^T (y - X\beta) + \lambda \sum_{j=1}^{p} |\beta_j|. \tag{3}$$

We take a look at an implementation of this minimization procedure via coordinate descent. This method minimizes a function, e.g. a residual sum of squares, over β "step by step", i.e. all β_k, $k = 1,...p$ except one, say β_j, are kept fixed (at their current value) and the function is minimized only over one coordinate, i.e. over one coefficient, β_j, which is then updated. This is carried out for all β_j, $j = 1,...,p$ and then the whole cycle is repeated until convergence. The coordinate updates can be written down explicitly as

$$\tilde{\beta}_j \leftarrow \frac{S\left(\sum_{i=1}^{n} 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right), \lambda\right)}{2n},$$

where $S(\cdot,\cdot)$ is the soft thresholding operator defined by

$$S(x,\kappa) = sign(x)(|x| - \kappa)_+ = \begin{cases} x - \kappa & \text{if } x > 0 \text{ and } |x| > \kappa \\ x + \kappa & \text{if } x < 0 \text{ and } |x| > \kappa \\ 0 & \text{if } |x| \leq \kappa \end{cases} \tag{4}$$

and $\tilde{y}_i^{(j)} := \sum_{k \neq j} x_{ik}\tilde{\beta}_k$ (the tilde indicates that the values are fixed at their current value).[8] This algorithm (updating each coefficient β_j in this way continuously) converges to the lasso solution.

[8]This can be seen as follows. The function to be minimized is the penalized residual sum of squares (3), which can be rewritten a little bit ($k = 1,...,p$, $j \in \{1,...,p\}$): $RSS(\lambda,\beta) = \sum_{i=1}^{n} \left(y_i - \sum_{k\neq j} x_{ik}\beta_k - x_{ij}\beta_j\right)^2 + \lambda \sum_{k\neq j} |\beta_k| + \lambda|\beta_j|$. Now, only β_j is considered and minimized over it, while all others are kept fixed at their current values $\tilde{\beta}_k$. A possible starting value is for example zero for all coefficients. For $\beta_j > 0$, the derivative with respect to β_j becomes $\frac{\partial}{\partial \beta_j} RSS(\lambda,\beta) = \sum_{i=1}^{n} \left(-2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) + 2x_{ij}^2\beta_j\right) + \lambda$. Setting this equal to zero, having in mind that $\sum_{i=1}^{n} x_{ij}^2 = n$ (stan-

When dealing with high-dimensional data, computational efficiency is very important. Friedman et al. (2010) introduce covariance updates to get a considerable speed-up of the procedure in most cases. Denoting by $x_{(j)}$ the j-th column of the input matrix and by $\langle \cdot, \cdot \rangle$ the standard inner product, the coordinate updates from above can be written as

$$\tilde{\beta}_j \leftarrow \frac{S\left(2\left(n\tilde{\beta}_j + \langle x_{(j)}, y\rangle - \sum_{k=1}^{p} \langle x_{(j)}, x_{(k)}\rangle \tilde{\beta}_k\right), \lambda\right)}{2n}.$$

Using these updates, the inner products of y with the columns of the input matrix and the inner products of two columns of this matrix can be computed in the beginning and then stored. Each time a coordinate is updated, the values of these inner products are accessed. As often many of these updates have to be performed this can result in a considerable reduction of compute time.

Coordinate descent with a network penalty

When a network penalty is present, the following residual sum of squares has to be minimized:

$$\begin{aligned} RSS(\lambda_1, \lambda_2, \beta) &= \sum_{i=1}^{n}\left(y_i - \sum_{h=1}^{p} x_{ih}\beta_h\right)^2 + \lambda_1 \sum_{h=1}^{p}|\beta_h| + \lambda_2 \beta^T M \beta \\ &= \sum_{i=1}^{n}\left(y_i - \sum_{k\neq j} x_{ik}\beta_k - x_{ij}\beta_j\right)^2 + \lambda_1 \sum_{h=1}^{p}|\beta_h| + \lambda_2 \sum_{h=1}^{p} M_{hh}\beta_h^2 + \lambda_2 \sum_{u\sim v} 2M_{uv}\beta_u\beta_v. \end{aligned}$$

Now we want to look at β_j and minimize over it, while keeping all other coefficients fixed at their current values $\tilde{\beta}_k$. For $\beta_j > 0$ and $\tilde{y}_i^{(j)} := \sum_{k\neq j} x_{ik}\tilde{\beta}_k$, the derivative with respect to β_j becomes

$$\frac{\partial}{\partial \beta_j} RSS(\lambda_1, \lambda_2, \beta) = \sum_{i=1}^{n}\left(-2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) + 2x_{ij}^2\beta_j\right) + \lambda_1 + 2\lambda_2 M_{jj}\beta_j + 2\lambda_2 \sum_{j\neq v} M_{jv}\tilde{\beta}_v.$$

dardized input), yields $\beta_j = \frac{\sum_{i=1}^{n} 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) - \lambda}{2n}$. For $\beta_j < 0$ differentiating and setting the derivative equal to zero leads to $\beta_j = \frac{\sum_{i=1}^{n} 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) + \lambda}{2n}$. It is important to note that setting the derivative equal to zero leads to the minimum only if $|\sum_{i=1}^{n} 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right)| \geq \lambda$, otherwise the minimum is at $\beta_j = 0$ (see Friedman et al., 2007). Thus we arrive at the solution given in the main text.

Setting this equal to zero, having in mind that $\sum_{i=1}^n x_{ij}^2 = n$ (standardized input), we get

$$\beta_j = \frac{\sum_{i=1}^n 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) - 2\lambda_2 \sum_{j\neq v} M_{jv}\tilde{\beta}_v - \lambda_1}{2n + 2\lambda_2 M_{jj}}.^9$$

This finally leads to coordinate updates of the form

$$\tilde{\beta}_j \leftarrow \frac{S\left(\sum_{i=1}^n 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) - 2\lambda_2 \sum_{j\neq v} M_{jv}\tilde{\beta}_v, \lambda_1\right)}{2n + 2\lambda_2 M_{jj}},$$

where $S(\cdot,\cdot)$ is the soft thresholding operator as defined in expression (4).

Covariance updates with a network penalty

As already mentioned, for methods concerning high-dimensional data, computational efficiency is very important. Therefore we use covariance updates. We have

$$y_i - \tilde{y}_i^{(j)} = y_i - \tilde{y}_i + x_{ij}\tilde{\beta}_j = r_i + x_{ij}\tilde{\beta}_j,$$

where $\tilde{y}_i = \sum_{j=1}^p x_{ij}\tilde{\beta}$ and r_i is the current residual. Because the input matrix is standardized, we have

$$\sum_{i=1}^n x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) = \sum_{i=1}^n x_{ij} r_i + n\tilde{\beta}_j.$$

Then we can write

$$\sum_{i=1}^n x_{ij} r_i = \langle x_{(j)}, y\rangle - \sum_{k=1}^p \langle x_{(j)}, x_{(k)}\rangle \tilde{\beta}_k,$$

where $x_{(j)}$ is the j-th column of the input matrix ($\langle\cdot,\cdot\rangle$ is again the standard inner product). Finally we can write the coordinate updates as

$$\tilde{\beta}_j \leftarrow \frac{S\left(2\left(n\tilde{\beta}_j + \langle x_{(j)}, y\rangle - \sum_{k=1}^p \langle x_{(j)}, x_{(k)}\rangle \tilde{\beta}_k\right) - 2\lambda_2 \sum_{j\neq v} M_{jv}\tilde{\beta}_v, \lambda_1\right)}{2n + 2\lambda_2 M_{jj}}.$$

We can then compute the inner products of y and all columns of the covariate matrix as well as all inner products of two columns of the covariate matrix in the beginning and store them. At each coordinate update they are accessed – this leads to a considerable reduction of compute time.

Convergence results for the coordinate descent algorithm

We have not established yet that the coordinate descent algorithm converges to the

[9] For $\beta_j < 0$, we similarly get $\beta_j = \frac{\sum_{i=1}^n 2x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) - 2\lambda_2 \sum_{j\neq v} M_{jv}\tilde{\beta}_v + \lambda_1}{2n + 2\lambda_2 M_{jj}}$.

global minimum given the penalty structure we use. It indeed converges to the global minimum, which we prove now.

We make use of a theorem from Tseng (1988) stating that the coordinate descent algorithm converges to the global minimum in cases where the function f that shall be minimized is of the form

$$f(\beta_1, ..., \beta_p) = g(\beta_1, ..., \beta_p) + \sum_{j=1}^{p} h_j(\beta_j),$$

with g differentiable and convex and $h_j,\ j = 1, ..., p$ convex.

For fixed parameters λ_1 and λ_2, we can write the penalized residual sum of squares of as

$$RSS(\beta) = g(\beta) + \sum_{j=1}^{p} h_j(\beta_j),$$

with $g(\beta) = (y - X\beta)^T (y - X\beta) + \lambda_2 \beta^T M \beta$ and $h_j(\beta_j) = \lambda_1 |\beta_j|$.

Then g is a sum of differentiable and convex functions and thus again differentiable and convex, while the h_j are obviously convex. Thus, the coordinate descent algorithm converges to the global minimum.[10]

Storing decomposition elements in the connection sign estimation step

Dealing with high-dimensional data the number of connections can be very large (M is a $p \times p$ matrix). Therefore it can be computationally very expensive to solve a whole linear regression problem for each connection many times in the algorithm (in the connection sign estimation step), although each single one of these problems with $p = 2$ is solved very fast. We now show a decomposition that can yield a considerable speed-up of the computations.

Note that with the notation $\tilde{X} := \left(x_{(i)}, x_{(j)}\right)$, the input matrix consisting only of the two columns $x_{(i)}$ and $x_{(j)}$ and $\tilde{\beta} := (\hat{\beta}_i^*, \hat{\beta}_j^*)^T$ (the tildes are not to be confused with the tildes in the coordinate descent algorithm), the solution to the least squares problem of Expression 2 (when satisfying the usual conditions) becomes

$$\tilde{\beta} = \left(\tilde{X}^T \tilde{X}\right)^{-1} \tilde{X}^T \tilde{y}.$$

[10] Here we can nicely see a sufficient condition that the penalty matrix M must satisfy for the coordinate descent algorithm to work. If $\varphi_M(\beta) = \beta^T M \beta$ is convex, the coordinate wise descent algorithm is sure to converge to the global minimum (the differentiability is obvious). That is in particular the case for all M which allow $\beta^T M \beta$ to be written as a sum of squared terms as assumed in the beginning of this section, which are the only penalty matrices we are interested in. Of course, the normalized and the combinatorial Laplacian satisfy this sufficient condition.

As $\tilde{y} = y - X^{-(i,j)}\hat{\beta}^{-(i,j)}$, this is equivalent to

$$\tilde{\beta} = \left(\tilde{X}^T\tilde{X}\right)^{-1}\tilde{X}^T y - \left(\tilde{X}^T\tilde{X}\right)^{-1}\tilde{X}^T X^{-(i,j)}\hat{\beta}^{-(i,j)}.$$

For each connection (between two covariates i and j), $\tilde{X}$, $X^{-(i,j)}$ and y do not change when $\hat{\beta}$ changes. Thus, for each connection (again between two covariates i and j) we compute in the beginning

$$B_y^{ij} := \left(\tilde{X}^T\tilde{X}\right)^{-1}\tilde{X}^T y$$

and

$$B_X^{ij} := \left(\tilde{X}^T\tilde{X}\right)^{-1}\tilde{X}^T X^{-(i,j)},$$

which are 2×1 and $2 \times (p-2)$-matrices, respectively. These matrices are stored and accessed each time one of these small linear models has to be solved. This reduces the problem of solving a linear model each time to a single matrix multiplication and subtraction and leads to a considerable speed-up of the computation.

2.3 Adapting the new method to time-to-event data

When gene expression data is used for prediction, we are often in a time-to-event setting, for example predicting the probability of survival for cancer patients. In a time-to-event setting, observations are usually of the form (t_i, δ_i, x_i), $i = 1, ..., n$, where t_i is the observed time, δ_i is the censoring indicator taking value one, if the observed time is due to an event and value zero, if the observed time is due to censoring, x_i is the covariate vector of observation (or individual) i.

In time-to-event settings, the Cox model (Cox, 1972) is often appropriate. There, the hazard $\lambda(t|x_i)$, i.e. the instantaneous risk of having an event at time t, is modeled as

$$\lambda(t|x_i) = \lambda_0(t)exp(x_i^T\beta) = \lambda_0(t)exp(\eta_i),$$

where $\lambda_0(t)$ is an unspecified baseline hazard. $\hat{\beta}$ is obtained by maximizing the partial log-likelihood

$$\ell(\beta) = \sum_{i=1}^{n} \delta_i \left(\eta_i - log\left(\sum_{k=1}^{n} I(t_k \geq t_i)exp(\eta_k) \right) \right),$$

where $I(\cdot)$ denotes the indicator function.

For the case of real data, when there is a possibility of ties, i.e. a possibility of having two equal observation times, the most common approximations of the partial log-likelihood are the ones of Breslow (1975) and the more accurate one provided by Efron (1977),

which we use here. Let D_i denote the set of all $k \in \{1,...,n\}$ with $t_k = t_i$. We consecutively number all elements of D_i from 1 to $|D_i|$. When $D_i = D_j$ we agree that the numbers in the sets have to be equal, thus any $i \in \{1,...,n\}$ has exactly one associated number which we denote by $\#i$ (note that once we have chosen this number, it is fixed, but we can choose the numbers within each D_i (and simultaneously the numbers in the equal sets) in the beginning arbitrarily without changing the partial log-likelihood). Then the Efron version of the partial log-likelihood can be written as[11]

$$\ell(\beta) = \sum_{i=1}^{n} \delta_i \left(\eta_i - log \left(\left(\sum_{k=1}^{n} I(t_k \geq t_i) exp(\eta_k) \right) - \frac{\#i - 1}{|D_i|} \sum_{l \in D_i} exp(\eta_l) \right) \right). \tag{5}$$

For the estimation of the cumulative baseline hazard the Breslow estimator can be used, which is in the Efron version

$$\hat{\Lambda}_0(t) = \sum_{t_i \leq t} \frac{1}{(\sum_{k=1}^{n} I(t_k \geq t_i) exp(\hat{\eta}_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\hat{\eta}_l)},$$

thus

$$\hat{S}_0(t) = exp\left(-\hat{\Lambda}_0(t)\right) = exp\left(- \sum_{t_i \leq t} \frac{1}{(\sum_{k=1}^{n} I(t_k \geq t_i) exp(\eta_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\eta_l)} \right).$$

Then we have an estimator for the survival function S. $S(t|x)$ is the probability of still being alive at time t for an individual with covariate vector x. For such an individual the estimate of the survival function is

$$\hat{S}(t|x) = \left(\hat{S}_0(t)\right)^{exp(x^T \hat{\beta})}. \tag{6}$$

To implement this method, one possibility to maximize the partial log-likelihood of the Cox model is via iteratively reweighted least squares (Green, 1984; see also Hastie and Tibshirani, 1990). We note that $\frac{\partial l}{\partial \beta} = \left(\frac{\partial \eta}{\partial \beta}\right)^T \frac{\partial l}{\partial \eta} = X^T \frac{\partial l}{\partial \eta}$ and use the approximation

$$\frac{\partial^2 l}{\partial \beta \beta^T} \approx X^T \frac{\partial^2 l}{\partial \eta \eta^T} X,$$

$\frac{\partial l}{\partial \beta} \in \mathbb{R}^{p \times 1}$, $\frac{\partial^2 l}{\partial \beta \beta^T} \in \mathbb{R}^{p \times p}$. Let $u = \frac{\partial l}{\partial \eta}$ and $A = -\frac{\partial^2 l}{\partial \eta \eta^T}$, $u \in \mathbb{R}^{n \times 1}$, $A \in \mathbb{R}^{n \times n}$. Then the iteratively reweighted least squares algorithm that implements the Newton-Raphson

[11] This way to write the Efron version of the partial log-likelihood is not very common, but convenient for our demands later on.

method to maximize the partial log-likelihood involves finding β that solves

$$X^T A X(\beta - \beta_0) = X^T u,$$

where β_0 is the value of β from the previous step and A and u are evaluated at $\eta_0 = X\beta_0$. This is equivalent to finding the β that minimizes

$$(y^* - X^*\beta^*)^T (y^* - X^*\beta^*)$$

with $X^* = QX$, $y^* = \tilde{Q}u$ and $\beta^* = \beta - \beta_0$ for matrices Q and $\tilde{Q}$ with $Q^T Q = A$ and $Q^T \tilde{Q} = I$, if these matrices exist. They do not have to exist, but they do for example if A is positive definite.

The implementation of the Cox model via iteratively reweighted least squares can also be used when the partial log-likelihood is penalized, which results in a corresponding penalty on the residual sum of squares of the adapted data set. This can be seen when one regards the maximization of a penalized (log-)likelihood as a maximization under constraints (see Hastie et al., 2009). This is what we do here.[12]

The Network Algorithm can be applied to time-to-event data by solving the above least squares problem using a Network Algorithm instead of ordinary least squares (the penalization of the log-likelihood leads to a penalization of the residual sum of squares representing the Newton-Raphson steps).[13] Similar to Witten and Tibshirani (2009) we propose to perform only one step to save compute time. As quadratic functions are minimized in one step via the Newton-Raphson algorithm, it seems to be admissible to perform only one Newton Raphson step and empirically this approach works well.

[12]For the calculation of u and A as introduced above, we need the derivatives of the Efron approximation of the log-likelihood (Equation 5). They are

$$\frac{\partial l}{\partial \eta_j} = \delta_j - \sum_{i=1}^{n} \delta_i \frac{I(t_j \geq t_i)exp(\eta_j) - \frac{\#i-1}{|D_i|} exp(\eta_j) I(j \in D_i)}{(\sum_{k=1}^{n} I(t_k \geq t_i)exp(\eta_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\eta_l)},$$

$$\frac{\partial^2 l}{\partial \eta_j^2} = \sum_{i=1}^{n} \delta_i \left(- \frac{I(t_j \geq t_i)exp(\eta_j) - \frac{\#i-1}{|D_i|} exp(\eta_j) I(j \in D_i)}{(\sum_{k=1}^{n} I(t_k \geq t_i)exp(\eta_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\eta_l)} + \frac{(I(t_j \geq t_i)exp(\eta_j) - \frac{\#i-1}{|D_i|} exp(\eta_j) I(j \in D_i))^2}{((\sum_{k=1}^{n} I(t_k \geq t_i)exp(\eta_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\eta_l))^2} \right),$$

and for all $j \neq m$

$$\frac{\partial^2 l}{\partial \eta_j \eta_m} = \sum_{i=1}^{n} \left(\delta_i \frac{\left(I(t_j \geq t_i)exp(\eta_j) - \frac{\#i-1}{|D_i|} exp(\eta_j) I(j \in D_i) \right)}{\left((\sum_{k=1}^{n} I(t_k \geq t_i)exp(\eta_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\eta_l) \right)} \frac{\left(I(t_m \geq t_i)exp(\eta_m) - \frac{\#i-1}{|D_i|} exp(\eta_m) I(m \in D_i) \right)}{\left((\sum_{k=1}^{n} I(t_k \geq t_i)exp(\eta_k)) - \frac{\#i-1}{|D_i|} \sum_{l \in D_i} exp(\eta_l) \right)} \right).$$

[13]To make sure that we can find matrices Q and $\tilde{Q}$, which may be impossible when censoring occurs, we add a small multiple of the identity matrix to the matrix A. Thus, instead of A we use the matrix $A + \tau I$. In the applications shown later we use $\tau = n \cdot 10^{-6}$.

$\beta_0 = 0$ is used as starting value. The cumulative baseline hazard can then be estimated by the Breslow estimator in the Efron version, leading to the estimate of the survival function as in Equation 6.

3 Simulations

The simulations we consider are the same simulations as in Li and Li (2008). Suppose that there is a regulatory network with 200 transcription factors (TFs), each of them controlling ten genes. The resulting network then consists of 2200 genes and the connections between the TFs and the genes that they regulate. The covariate matrix X has 2200 columns, the first column consists of the expression levels of the first TF, the next ten columns of the expression levels of the genes regulated by it, and so on. The four different models (i.e. simulation scenarios) are as follows, differing only in the nature of β.

We assume that $y_i = X_i\beta + \varepsilon_i$, the number of observations being 100 and $\varepsilon_i \sim N(0,\sigma^2)$, where $\sigma^2 = (\sum_j \beta_j^2)/4$. The expression levels of the 200 TFs are independent and follow a standard normal distribution. The expression levels of the TFs and the genes they regulate are jointly distributed as bivariate normal with a correlation of 0.7. A gene regulated by the j-th TF, with expression level TF_j, follows thus a $N(0.7 \cdot TF_j,\ 0.51)$-distribution. In the first model

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

In the second model

$$\beta = \left(5, \underbrace{\frac{-5}{\sqrt{10}},...,\frac{-5}{\sqrt{10}}}_{3}, \underbrace{\frac{5}{\sqrt{10}},...,\frac{5}{\sqrt{10}}}_{7}, -5, \underbrace{\frac{5}{\sqrt{10}},...,\frac{5}{\sqrt{10}}}_{3}, \underbrace{\frac{-5}{\sqrt{10}},...,\frac{-5}{\sqrt{10}}}_{7},\right.$$
$$\left. 5, \underbrace{\frac{-5}{\sqrt{10}},...,\frac{-5}{\sqrt{10}}}_{3}, \underbrace{\frac{5}{\sqrt{10}},...,\frac{5}{\sqrt{10}}}_{7}, -5, \underbrace{\frac{5}{\sqrt{10}},...,\frac{5}{\sqrt{10}}}_{3}, \underbrace{\frac{-5}{\sqrt{10}},...,\frac{-5}{\sqrt{10}}}_{7}, 0,...,0\right).$$

In the third and fourth models β is the same as in the first two models, respectively, but the coefficients of the genes that are regulated by a transcription factor are divided by 10 instead of $\sqrt{10}$.

Some doubts whether these simulations make sense biologically or not seem to be appropriate. It seems very unrealistic to have genes with fairly large true coefficients that are positively correlated to a transcription factor that has a large negative true coefficient. Also, scenarios 1 and 3 are somewhat extreme, because in these scenarios, all connections have positive signs. We use these scenarios nevertheless – in doing so we use a setting that is beyond any doubt of having been created to favor our method. Scenarios one and three are also interesting from the particular perspective of how much our algorithm could "destroy" when indeed all of the connections have positive signs.

For each scenario considered, 50 training and 50 test data sets are simulated.[14] The combination of penalty parameters is chosen on a two dimensional grid. This is done in the training set, via 10-fold cross-validation. Then the predicted mean squared error (PMSE) is calculated on the corresponding test set.

Prediction performance is not the only criterion for a regression model. It is also important to have high interpretability, which means for gene expression data especially to identify genes or networks that really influence the outcome. For Li and Li (2008) this is one of the main achievements of their new penalty, as it performs very well concerning sensitivity and specificity and thus concerning the identification of involved networks and pathways. Therefore, also sensitivity and specificity with respect to identification of non-zero coefficients are considered. Furthermore, we also show how many of the connection signs have been identified correctly. As mentioned before, this can be seen as a goal in itself as it could help bring about new biomedical knowledge.

3.1 Simulation Results

We compare the new method and two natural competitors. We consider two versions of the Network Algorithm, using either the normalized Laplacian as penalty matrix (denoted by NW-Alg) or the combinatorial Laplacian (denoted by NW-Alg$_{CL}$). We compare their performances to the performance of the network penalty as in Li and Li (2008), which we denote by Net (Li Li), and to the performance of the lasso. One would expect the fixed network penalty to perform best in scenarios 1 and 3 as there indeed all connection signs are positive. In models 2 and 4 we expect the Network Algorithms to perform better, as there are some connections with negative signs. Before we show the results in terms of prediction performance, we show the performance in terms of the

[14] This has been done with the parallel computing solution of sfCluster/`snowfall` (Knaus, 2008; Knaus et al., 2009).

identification of non-zero coefficients and the performance in terms of correct estimation of connection signs.

First we take a look at sensitivity and specificity with respect to the identification of non-zero coefficients. The results are shown in Table 1. The sensitivity shows the fraction of non-zero coefficients that have been correctly estimated to be non-zero, the specificity shows the fraction of zero coefficients that have been correctly estimated to be zero. The first column indicates the simulation scenario used. We can see that

Table 1: Sensitivity/Specificity

	NW-Alg	NW-Alg$_{CL}$	Lasso	Net (Li Li)
1	1.00/0.61	1.00/0.56	0.56/0.97	1.00/0.69
2	0.96/0.62	0.94/0.60	0.48/0.93	0.87/0.84
3	0.97/0.65	0.91/0.65	0.41/0.97	0.99/0.73
4	0.84/0.71	0.75/0.72	0.39/0.95	0.87/0.76

the Network Algorithms in general keep the fixed network penalty's property of good identification of non-zero coefficients. The Network Algorithm with the normalized Laplacian performs better than the one with the combinatorial Laplacian which still performs quite well.

Now we deal with the question of how well the signs of the connections are identified by our algorithms. The lasso and the network penalty from Li and Li (2008) are not meant to discover any connection signs, but they are nevertheless shown as a comparison. We take a connection sign between two covariates to be estimated correctly when the estimates of the coefficients of the two connected covariates are non-zero and when either both estimates and both true coefficients have the same signs or if both have different signs. The results are shown in Table 2. The fourth scenario seems

Table 2: Fraction of correctly estimated connection signs

	NW-Alg	NW-Alg$_{CL}$	Lasso	Net (Li Li)
1	0.99	0.99	0.56	1.00
2	0.84	0.79	0.44	0.69
3	0.94	0.90	0.39	0.99
4	0.64	0.58	0.33	0.65

to be somehow difficult for the Network Algorithms, here only around sixty percent

of the connection signs are identified correctly. This could be the case, because the unrealistic structure of positive correlations of covariates that influence the outcome in different directions might be especially harmful to the Network Algorithms when the true coefficients of these covariates are relatively small, which is the case in scenario 4. In the other three scenarios the results are very good. In scenarios 1 and 3 where all connection signs are positive, more than 90% of the signs are identified correctly by both versions of the Network algorithm. In scenario 2 with different connection signs, the Network Algorithm with the normalized Laplacian estimates 84% of the connection signs correctly, the algorithm with the combinatorial Laplacian identifies 79% of the connection signs correctly.

Now we get to the prediction performance. The results are shown in Table 3. The values are the predicted mean squared errors (PMSE) with standard errors (SE) in parentheses below. As a comparison, the PMSE when predicting with the true β (True model) and the prediction error of a null model using no covariate information at all, always predicting the intercept (Intercept) are given. The two best performances in each scenario are shown in bold.

Table 3: Predicted mean squared errors (two best performances in bold, SE in parentheses)

	True model	Intercept	NW-Alg	NW-Alg$_{CL}$	Net (Li Li)	Lasso
1	33.3	743.9	**45.2**	**45.2**	45.7	84.4
	(0.6)	(15.9)	(1.1)	(1.1)	(1.2)	(2.3)
2	33.3	299.5	**66.1**	**76.7**	83.5	90.4
	(0.6)	(6.8)	(2.1)	(2.5)	(2.5)	(2.2)
3	18.3	209.9	**27.3**	29.3	**26.1**	32.5
	(0.3)	(4.6)	(0.7)	(0.7)	(0.6)	(1.1)
4	18.3	127.8	33.5	35.3	**32.3**	**32.9**
	(0.3)	(2.9)	(0.9)	(0.9)	(0.9)	(0.8)

We can see that the newly developed algorithms compete well with the lasso and the fixed network penalty. We also see that the Network Algorithms do not achieve considerably worse results than the fixed network penalty even when all connection signs are positive (scenarios 1 and 3). In scenario 4, the performance of the Network Algorithms is also very similar to the performances of the fixed network penalty and the lasso, although only around 60 percent of the connection signs were identified correctly (see Table 2). In scenario 2, we see that the Network Algorithms achieve results that are by far better than the result of the fixed network penalty which still outperforms the

lasso. The Network Algorithm with the normalized Laplacian performs consistently better than the one using the combinatorial Laplacian. Nevertheless, the algorithm with the combinatorial Laplacian also performs well.

4 Applications to two microarray time-to-event data sets

In this section, we apply the new method to two data sets of microarray survival data (as explained in Section 2.3).

To estimate the prediction error of a model fitted to real data, bootstrap resampling is often adequate (Efron and Tibshirani, 1993). Because the bootstrap estimate is biased upwards, Efron and Tibshirani (1997) propose the bootstrap .632+ estimate, which is a weighted combination of the error in the full data set and the bootstrap error estimate. To evaluate prediction performance in time-to-event settings, bootstrap .632+ prediction error curve estimates are appropriate (Gerds and Schumacher, 2007; Schumacher et al., 2007; Gerds and Schumacher, 2006), which track the .632+ prediction error estimates over time. The quantity that is estimated is the (time dependent) Brier score

$$BS\left(t,\hat{S}\right) = E\left(Y\left(t\right) - \hat{S}\left(t|Z\right)\right)^2,$$

where $Y(t) = I(T > t)$ is the true event status at time t and $\hat{S}(t|Z)$ is the predicted event status at that time.

We use bootstrap resampling without replacement, drawing $0.632 \cdot n$ observations per sample, which is the expected number of unique observations when drawing with replacement. Binder and Schumacher (2008a) show that this results in better estimation of the prediction error (.632+ estimates can be employed here as well). To select the penalty parameters in each of the bootstrap samples and in the full data set, 10-fold cross-validation is used and the parameters are selected that maximize the predictive partial log-likelihood.[15]

As the observations are not uniformly distributed over time, the interpretation of the prediction error curves can sometimes be difficult. Therefore and to have a good summary measure of these curves, we also give the estimated integrated prediction error curves (IPEC, see Gerds and Schumacher, 2007, and Porzelius et al., 2010). The IPEC is the integral of the prediction error curve with respect to the distribution of event times.

[15]The .632+ prediction error curve estimates are computed employing the `R` package `peperr` (Porzelius and Binder, 2009; Porzelius et al., 2009). The number of bootstrap samples is 50.

Thus,

$$IPEC\left(\hat{S}\right) = \int_0^\infty BS\left(t, \hat{S}\right) p(t) dt$$

is estimated, where $p(t)$ is the probability density of the event times. Low values indicate good performance.

We show the results of the same methods as in Section 3.1. As comparison, we also show the prediction error curve of the null model, which is the Kaplan-Meier estimator (Kaplan and Meier, 1958).

4.1 Application to microarray data from patients with ovarian cancer

In this section, we apply the methods to time-to-event data from 133 patients with ovarian cancer. 72 of the patients had an event. Preprocessing of the microarray data using the RMA approach (Irizarry et al., 2003) was performed, resulting in 21801 microarray features. The original analysis of the data by Bild et al. (2006) already shows a connection between pathway activity and survival, with pathway information that is derived from prior experiments (see also Binder and Schumacher, 2009). We restrict our analysis to the 2503 features included in regulatory or cancer pathways in the KEGG database.

Figure 1 shows the prediction error curves. Table 4 shows the values of the estimated IPEC, multiplied by 100 for convenience.

Table 4: IPEC of different methods applied to time-to-event microarray data from patients with ovarian cancer (two best performances in bold)

	Null model	NW-Alg	NW-Alg$_{CL}$	Net (Li Li)	Lasso
IPEC $\cdot$ 100	14.27	**11.85**	**12.17**	12.43	13.65

We see that all three considered methods using network information improve considerably over the null model and the lasso. The Network Algorithm with the normalized Laplacian performs best, but the difference to the Network Algorithm with the combinatorial Laplacian is not very big. The network penalty of Li and Li (2008) performs well compared to the null model and the lasso not incorporating any network information.

Both Network Algorithms (with normalized and combinatorial Laplacian) select almost all of the 2503 covariates (the one with the normalized Laplacian 2502 covariates of

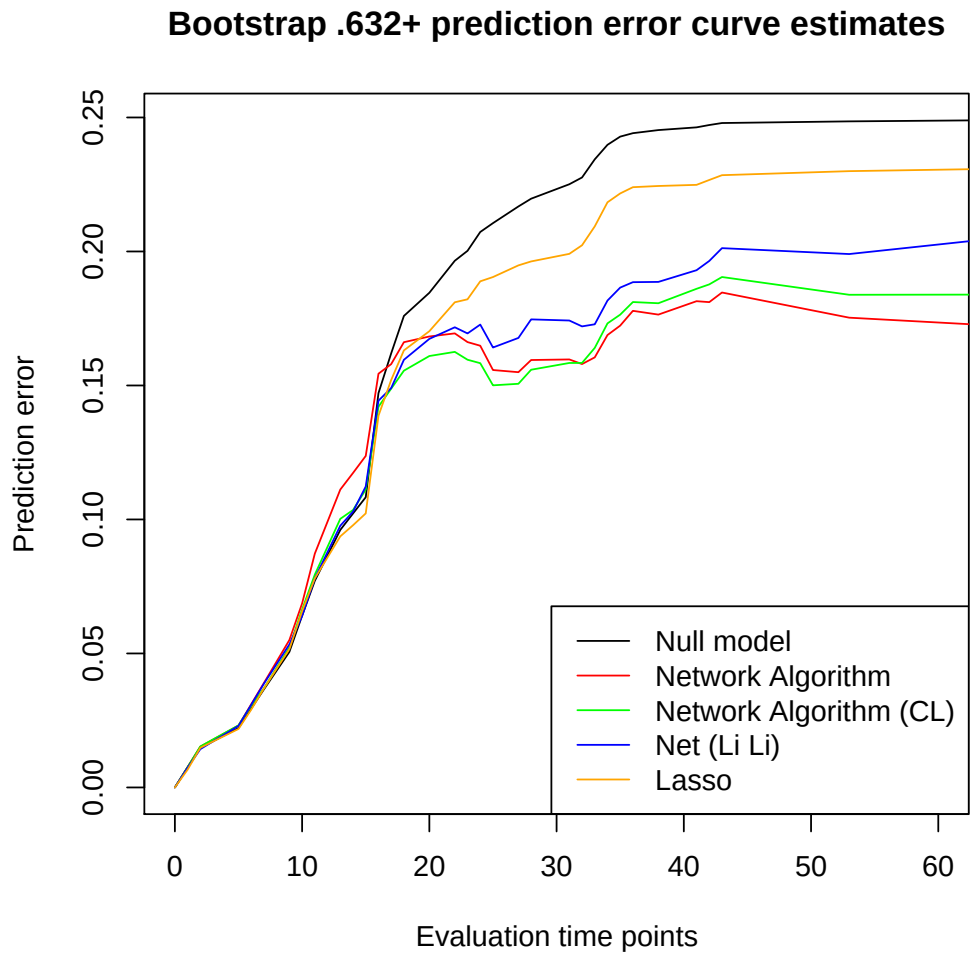

Figure 1: Prediction error curves of different methods applied to time-to-event microarray data from patients with ovarian cancer

which 1331 have positive coefficients, the one with the combinatorial Laplacian 2499 covariates of which 1246 have positive coefficients). The network penalty of Li and Li (2008) selects 1592 covariates of which 870 have positive coefficients. The regulatory network consists of 28945 connections. The Network Algorithm with the normalized Laplacian estimates 14821 of the connection signs to be positive and 14088 to be negative. The Network Algorithm with the combinatorial Laplacian estimates 14482 of the connection signs to be positive and 14411 to be negative.

4.2 Application to microarray data from patients with large B-cell lymphoma

In this section, the methods are applied to time-to-event data from patients with diffuse large B-cell lymphoma (Rosenwald et al., 2002). These data have already been

used to illustrate prediction error curve techniques (Schumacher et al., 2007) and a likelihood-based boosting technique for the Cox proportional hazards model (Binder and Schumacher, 2008b). Details of preprocessing of the data are explained there. There are 240 observations with 7399 microarray features, the number of events is 133. Again, we restrict the analysis to the microarray features represented in regulatory and cancer pathways in the KEGG pathway database. This reduces the number of microarray features to 1281.

Figure 1 shows the prediction error curves and Table 4 the values of the estimated IPEC, again multiplied by 100 for convenience.

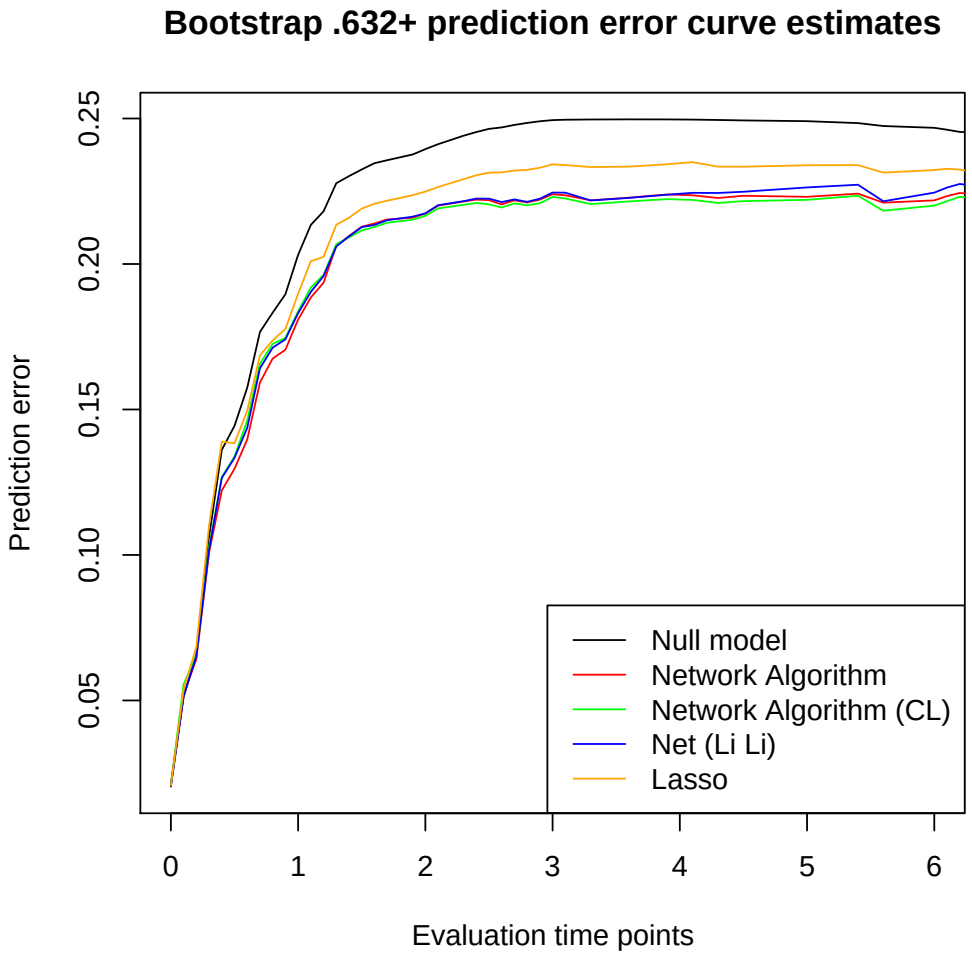

Figure 2: Prediction error curves of different methods applied to time-to-event microarray data from patients with large B-cell lymphoma

Again, we can see that the three methods incorporating network information perform better than the null model and the lasso. The Network Algorithm with the normalized Laplacian performs best (in terms of IPEC), followed closely by the Network Algorithm with the combinatorial Laplacian and the fixed network penalty.

Table 5: IPEC of different methods applied to time-to-event microarray data from patients with large B-cell lymphoma (two best performances in bold)

	Null model	NW-Alg	NW-Alg$_{CL}$	Net (Li Li)	Lasso
IPEC $\cdot$ 100	13.47	**12.04**	**12.17**	12.23	12.86

Of the 1281 covariates, 924 are selected by the Network Algorithm with the normalized Laplacian, 450 of the coefficients have positve sign. The Network Algorithm with the combinatorial Laplacian selects 935 covariats, of which 443 have positive coefficients. The network penalty of Li and Li (2008) selects 721 covariates, of which 378 have positive coefficients. The regulatory network in this example consists of 12144 connections. The Network Algorithm with the normalized Laplacian estimates 3696 of the connection signs to be positive and 3650 to be negative. The Network Algorithm with the combinatorial Laplacian estimates 4235 of the connection signs to be positive and 4284 to be negative.

5 Discussion

In this paper, we introduced a new method to incorporate external network information into regression models. We built upon a network penalty introduced by Li and Li (2008) where an additional penalty term, incorporating network information, is put on top of the lasso penalty. We first generalized their approach, not restricting the method to the assumption of only positive connections between covariates. Thus, in addition to activating (positive) connections, also repressing (negative) connections can be considered. As there is often not much knowledge on the signs of the connections available, we developed an algorithm that estimates both the covariate coefficients and the connection signs. An obvious goal of this algorithm is to lead to improved prediction performance, but it can also lead to better interpretable models and it can be used to gain additional knowledge on the signs of the network connections, which is (especially in the case of pathway information) important in itself.

Furthermore, we not only incorporated the network information via the normalized Laplace matrix, but also investigated the penalty when the additional information is incorporated via the combinatorial Laplace matrix. We developed a fast implementation via coordinate descent that can also be used for the fixed network penalty as proposed by Li and Li (2008). Then, we showed how to handle time-to-event data with these

methods. Therefore, we adapted the methods to the Cox proportional hazards model via iteratively reweighted least squares.

In simulation studies we tested the performance of the new algorithm, using the normalized Laplacian as well as the combinatorial Laplacian as penalty matrices. We compared the new methods to adequate competitors, regarding prediction performance, sensitivity and specificity (when judging whether covariates have an influence on the dependent variable), as well as the performance concerning the estimation of network connection signs. We found that the new algorithms perform well consistently – the prediction performance mainly improved over the competitors while the new algorithm generally keeps the good properties of the network penalty from Li and Li (2008) concerning sensitivity and specificity; furthermore, the majority of connection signs was discovered correctly.

In the end we applied both versions of the new method to time-to-event microarray data from patients with ovarian cancer and to time-to-event microarray data from patients with diffuse large B-cell lymphoma. There, the new methods improve massively over the null model and over the lasso, in terms of prediction performance. Both methods also perform better than the network penalty proposed by Li and Li (2008), which still outperforms the lasso. This affirms our approach and indicates that it is beneficial to allow for negative connection signs.

References

Bild, A. H., Yao, G., Chang, J. T., Wang, Q., Potti, A., Chasse, D., Joshi, M. B., Harpole, D., Lancaster, J. M., Berchuck, A., Olson, J. A., Marks, J. R., Dressman, H. K., West, M., and Nevins, J. R. (2006). Oncogenic pathway signatures in human cancers as a guide to targeted therapies. *Nature*, 439(7074):353–357.

Binder, H. and Schumacher, M. (2008a). Adapting prediction error estimates for biased complexity selection in high-dimensional bootstrap samples. *Statistical Applications in Genetics and Molecular Biology*, 7(2):Article12.

Binder, H. and Schumacher, M. (2008b). Allowing for mandatory covariates in boosting estimation of sparse high-dimensional survival models. *BMC Bioinformatics*, 9:14.

Binder, H. and Schumacher, M. (2009). Incorporating pathway information into boosting estimation of high-dimensional risk prediction models. *BMC Bioinformatics*, 10:18.

Breslow, N. (1975). Analysis of survival data under the proportional hazards model. *International Statistical Review*, 43:45–58.

Chung, F. (1997). Spectral graph theory. *CBMS Regional Conferences Series (Vol. 92), American Mathematical Society*.

Cox, D. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34:187–220.

Efron, B. (1977). The efficiency of Cox's likelihood function for censored data. *Journal of the American Statistical Association*, 72:557–565.

Efron, B. and Tibshirani, R. (1993). *An introduction to the bootstrap*. Chapman and Hall, New York.

Efron, B. and Tibshirani, R. (1997). Improvements on cross-validation: The .632+ bootstrap method. *Journal of the American Statistical Association*, 92(438):548–560.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360.

Friedman, J., Hastie, T., Höfling, H., and Tibshirani, R. (2007). Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1.

Gerds, T. A. and Schumacher, M. (2006). Consistent estimation of the expected Brier score in general survival models with right-censored event times. *Biometrical Journal*, 48:1029–1040.

Gerds, T. A. and Schumacher, M. (2007). Efron-type measures of prediction error for survival analysis. *Biometrics*, 63(4):1283–1287.

Green, P. (1984). Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of the Royal Statistical Society. Series B (Methodological)*, 46:149–192.

Hastie, T. and Tibshirani, R. (1990). *Generalized additive models.* Monographs on Statistics and Applied Probability. 43. London etc.: Chapman and Hall.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning. Data mining, inference, and prediction. 2nd ed.* Springer Series in Statistics. New York.

Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4:249–264.

Kaplan, E. and Meier, P. (1958). Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53:457–481.

Knaus, J. (2008). *snowfall: Easier cluster computing (based on snow)*. R package version 1.53.

Knaus, J., Porzelius, C., Binder, H., and Schwarzer, G. (2009). Easier parallel computing in R with snowfall and sfCluster. *The R Journal*, 1:54–59.

Li, C. and Li, H. (2008). Network-constrained regularization and variable selection for analysis of genomic data. *Bioinformatics*, 24(9):1175–1182.

McLachlan, G. J. and Krishan, T. (1997). *The EM algorithm and extensions*. Wiley, New York.

Porzelius, C. and Binder, H. (2009). *peperr: Parallelised estimation of prediction error*. R package version 1.1-2.

Porzelius, C., Binder, H., and Schumacher, M. (2009). Parallelized prediction error estimation for evaluation of high-dimensional models. *Bioinformatics*, 25:827–829.

Porzelius, C., Schumacher, M., and Binder, H. (2010). A general, prediction error-based criterion for selecting model complexity for high-dimensional survival models. *Statistics in medicine*, 29(7-8):830–838.

Rosenwald, A., Wright, G., Chan, W., Connors, J., Campo, E., Fisher, R., Gascoyne, R., Muller-Hermelink, H., Smeland, E., Staudt, L., and Lymphoma Leukemia Mol Profiling Pr (2002). The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. *The New England Journal of Medicine*, 346(25):1937–1947.

Schumacher, M., Binder, H., and Gerds, T. (2007). Assessment of survival prediction models based on microarray data. *Bioinformatics*, 23:1768–1774.

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

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

Tseng, P. (1988). Coordinate ascent for maximizing nondifferentiable concave functions. Technical report, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems.

Weber, M. (2009). Simultaneous estimation of covariate coefficients and connection signs. *Diplomarbeit, Faculty of Mathematics and Physics, University of Freiburg*.

Wei, P. and Pan, W. (2008). Incorporating gene networks into statistical tests for genomic data via a spatially correlated mixture model. *Bioinformatics*, 24:404–411.

Witten, D. and Tibshirani, R. (2009). Covariance-regularized regression and classification for high-dimensional problems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 71(3).

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B (Methodological)*, 67(2):301–320.