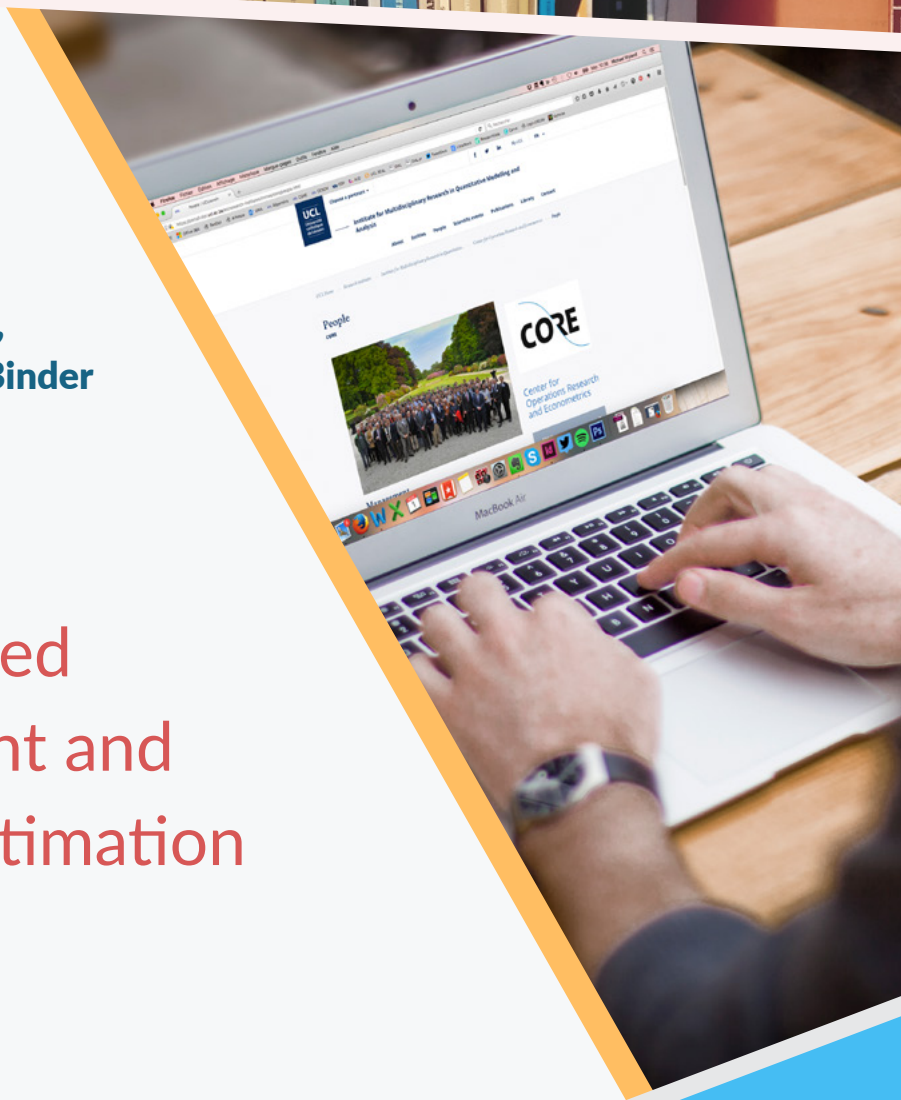
2018/18

DP

Matthias Weber, Jonas Striaukas, Martin Schumacher and Harald Binder



Network constrained covariate coefficient and connection sign estimation

CORE
CENTER FOR OPERATIONS RESEARCH AND ECONOMETRICS

CORE
Voie du Roman Pays 34, L1.03.01
Tel (32 10) 47 43 04
Fax (32 10) 47 43 01
Email: immaq-library@uclouvain.be
https://uclouvain.be/en/research-institutes/immaq/core/discussion-papers.html

Network Constrained Covariate Coefficient and Connection Sign Estimation*

MATTHIAS WEBER [‡] JONAS STRIAUKAS [†]

MARTIN SCHUMACHER [§] HARALD BINDER[§]

June 20, 2018

ABSTRACT

Often, variables are linked to each other via a network. When such a network structure is known, this knowledge can be incorporated into regularized regression settings. In particular, an additional network penalty can be added on top of another penalty term, such as a Lasso penalty. However, when the type of interaction via the network is unknown (that is, whether connections are of an activating or a repressing type), the connection signs have to be estimated simultaneously with the covariate coefficients. This can be done with an algorithm iterating a connection sign estimation step and a covariate coefficient estimation step. We show detailed simulation results of such an algorithm. The algorithm performs well in a variety of settings. We also briefly describe the R-package that we developed for this purpose, which is publicly available.

Keywords: Network regression; network penalty; connection sign estimation; regularized regression.

*We thank seminar participants at the University of Freiburg and the Bank of Lithuania for comments and suggestions. All simulations were conducted on the supercomputer from the High Performance Computing Center of the Lithuanian National Center of Physical and Technology Sciences at Vilnius University. This discussion paper supersedes an older version (Weber et al., 2014).

‡Research Center CEFER, Bank of Lithuania & Faculty of Economics and Business Administration, Vilnius University.

†Université catholique de Louvain, CORE, B-1348 Louvain-la-Neuve, Belgium. The author acknowledges the financial support of the FNRS PDR T.0138.15. Corresponding author. Email: jonas.striaukas@gmail.com.

§Institute of Medical Biometry and Statistics, University of Freiburg.

1 Introduction

Network data have become increasingly important in the last few decades. This concerns a variety of types of data and disciplines, including pathway data in biology that reveals information relevant for medical research, data on social networks from social media websites with up to a couple of billion users, or networks of financial institutions relevant for assessing financial stability and designing regulation.

However, while there have been attempts to provide statistical methods to deal with network data in the recent past, the development of such methods seems to be generally still in early stages, with much work left for the upcoming decades.

In this paper, we introduce a method to incorporate network information into a regularized regression setting and provide simulations showing that the method performs well. The method is an algorithm based on Li and Li (2008) who propose a particular network penalty on top of a Lasso penalty to incorporate network information about the covariates (that is, it is known which covariates are linked on a network). A potential problem with this method is that it assumes that all network connections have positive sign (that is that they are of a activating or enforcing type). However, there may also be network connections with negative signs (for example connections of a repressing type; in a biological application, this could for example be one gene downregulating another one). Often, the signs of the connections are not even known ex ante. In biology, for example, gene expression data is often represented by an undirected graph, without information of which of the connections are of an activating and which of a repressing type. Therefore, the algorithm described in this paper estimates the covariate coefficients and the connection signs simultaneously.

The idea of the algorithm can be summarized as follows. In the algorithm there are two steps. In a coefficient estimation step, the covariate coefficients are estimated by maximizing a penalized log-likelihood given estimates of the connection signs. Then, in a connection sign estimation step, the signs of the connections are estimated, taken the covariate coefficients as given from the last coefficient estimation step. These two steps are then repeated.

One of the goals of this method is to improve prediction performance. In addition to this, the method can help to gain knowledge about the signs of the coefficients. This can in cases where it is important to understand the signs of the connections be a goal in its own.

This paper is organized as follows. Section 2 briefly describes the algorithm to estimate covariate coefficients and connection signs and its implementation in a new accompanying R-package. Section 3 describes the simulations motivated by the availability of biological pathway information and presents the results. Section 4 concludes.

2 Method

We describe the method here concisely. For more detail, see the earlier unpublished version Weber et al. (2014), which is available online.

2.1 Background

We consider a continuous response y and covariate matrix X. The numbers of observations and covariates are n and p, respectively. Thus, $y = (y_1, ..., y_n)^T$ and X is an $n \times p$-matrix with rows $x_i^T = (x_{i1}, ..., x_{ip})$. $x_{(j)}$ denotes the j-th column. We assume y to be centered and X standardized, which means $\sum_{i=1}^n y_i = 0$, $\sum_{i=1}^n x_{ij} = 0$ and $\frac{1}{n}\sum_{i=1}^n x_{ij}^2 = 1$ for $j = 1, ..., p$. In the classical linear model $y_i = x_i^T\beta + \epsilon_i$, with $\epsilon_i \sim N(0, \sigma^2)$., where β is estimated.

The additional information for the regression problem is given by a regulatory network depicted by a weighted graph. The vertices 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

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

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 (see Chung, 1997). 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.

Li and Li (2008) propose to estimate β by minimizing the following penalized residual sum of squares (selecting λ_1 and λ_2 via 10-fold cross-validation):

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

The set of all edges is denoted by $\{u \sim v\}$ (this is the set of all directly linked pairs of predictors). 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 Ideas behind the Algorithm

It is implicitly assumed in Li and Li (2008) that all connections between the connected covariates are positive, that is, that they influence the outcome in the same direction. It is likely, though, that some of the connections have a negative sign (in a biological application, 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 $\lambda_2 \left(\frac{\beta_u}{\sqrt{d_u}} - \frac{\beta_v}{\sqrt{d_v}} \right)^2 w(u,v)$.

It is also possible to look at a different penalty matrix. In addition to the normalized Laplacian (or mutations of it that arise through negative connection signs), we also use the combinatorial Laplacian, which is defined by

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

With positive connection signs, the combinatorial Laplacian leads to the following penalized log-likelihood (noting that maximizing the penalized log-likelihood is equivalent to

minimizing the penalized RSS)

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

We define for all $i, j \in \{1, ..., p\}$

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

Given an initial penalty matrix M with $M_{ij} = 0$ if covariates $i \neq j$ are not connected (e.g., the normalized or combinatorial Laplacian), the optimal solution would be to use the penalty matrix with entries $-\xi_{ij}|(M)_{ij}|$ for $i \neq j$ and $|(M)_{ii}|$ otherwise. However, in many cases the ξ_{ij} are unknown and have to be estimated.[1] Similar to $\{u \sim v\}$, $\left\{u \overset{+}{\sim} v\right\}$ denotes the set of all connections assumed to have positive signs and $\left\{u \overset{-}{\sim} v\right\}$ that with negative signs. Then we have

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

Given estimates of the covariate coefficients (obtained via maximizing a penalized log-likelihood with some set of connection signs), the connection signs can be estimated (anew) as follows. To estimate the connection sign between covariates i and j, all covariate coefficients $\hat{\beta}_k$ are kept fixed, except for the two coefficients whose connection sign is being estimated ($k \neq i, j$), and then a small linear model is fitted for covariates i and j only. A connection sign is estimated to be positive if the coefficient estimates of the connected covariates in this small linear model have the same sign. This is a reasonable estimate of the connection sign as in many applications positively connected covariates influence the output variable in the same direction, while negatively connected covariates influence the output variable in opposite directions. These small linear models should thus reveal

[1] We assume that a penalty matrix M is symmetric and 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. The normalized and combinatorial Laplacians are of such a form.

information about the sign of the connection.

To be more detailed, the connection signs can be estimated as follows. $X^{-(i,j)}$ denotes the input matrix excluding columns i and j, $\hat{\beta}^{-(i,j)}$ denotes the estimate of β excluding $\hat{\beta}_i$ and $\hat{\beta}_j$, while $x_{(i)}$ denotes again the i-th column of the input matrix. Fitting the small linear model for covariates i and j means considering the new response $\tilde{y} = y - X^{-(i,j)}\hat{\beta}^{-(i,j)}$ and minimizing

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

over $(\beta_i, \beta_j)^T$. Let $(\hat{\beta_i}^*, \hat{\beta_j}^*)^T$ denote the minimizer of the above residual sum of squares. If and only if the signs of $\hat{\beta_i}^*$ and $\hat{\beta_j}^*$ are different, the connection between covariates i and j is estimated to have negative sign.

As starting values for the connection sign estimates, the signs of the empirical covariance of the columns of the input matrix can be taken. Because the covariate matrix is standardized, this boils down to the sign of $x_{(i)}^T x_{(j)}$. Thus the starting value of a connection sign is positive if $x_{(i)}^T x_{(j)} \geq 0$ and negative otherwise.

2.3 The 3CoSE Algorithm

The description above contains the ingredients of the algorithm. We propose to call it **3CoSE** (pronounced "three-cose"), standing for **Co**variate **Co**efficient and **Co**nnection **S**ign **E**stimation.[2] Given a log-likelihood $\ell(\beta)$, a penalty matrix M_1, and fixed penalty parameters, the algorithm consists of the following steps:

1. Determine starting values for the connection sign estimates via the empirical covariance, $\hat{\xi}_{ij} = sign(x_{(i)}^T x_{(j)})$.

[2] In Weber et al., 2014, the algorithm has no proper name yet and carries the working title "Network Algorithm".

2. Estimate β by maximizing the penalized log-likelihood

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

with the current estimates of the connection signs, where $(M)_{ij} = -\hat{\xi}_{ij}|(M_1)_{ij}|$ for $i \neq j$ and $(M)_{ii} = |(M_1)_{ii}|$.

3. Update the connection sign estimates by running mini OLS models, 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 (or for a fixed number of repetitions).

This procedure is often still be feasible when straightforward maximization of the penalized log likelihood over all coefficients and connection signs is computationally prohibitive, which is not unusual for big or high-dimensional data. The algorithm usually converges. When it does not converge, it can be run for a certain number of repetitions. 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. No converge is thus not necessarily problematic.

2.4 Implementation and R-Package LassoNet

For these simulations, the accompanying R-package LassoNet was developed, which is publicly available (Striaukas and Weber, 2018). It implements the covariate coefficient estimation step via coordinate descent (see Friedman et al., 2007). This means that only one covariate is updated at a time; all β_k, $k = 1, ...p$ except one, say β_j, are kept fixed at their current value and the maximization of the log-likelihood is conducted only for 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.

With network penalty matrix M, maximizing the log-likelihood is equivalent to min-

imizing this residual sum of squares:

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

One wants to minimize over one β_j 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

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

Setting this equal to zero, we get

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

We have used $\sum_{i=1}^{n} x_{ij}^2 = n$, which is the case as the input matrix is standardized. The case of $\beta_j < 0$ leads to a similar term.[3]

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 defined by

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

Especially for big or high-dimensional data, computational efficiency is very important. Therefore we use covariance updates that can lead to a considerable reduc-

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

tion of compute time (adapting the version proposed in Friedman et al., 2010). It is $y_i - \tilde{y}_i^{(j)} = y_i - \tilde{y}_i + x_{ij}\tilde{\beta}_j = r_i + x_{ij}\tilde{\beta}_j$, 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, it is $\sum_{i=1}^{n} x_{ij}\left(y_i - \tilde{y}_i^{(j)}\right) = \sum_{i=1}^{n} x_{ij}r_i + n\tilde{\beta}_j$ and then one 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 and $\langle\cdot,\cdot\rangle$ is the inner product. This leads to coordinate updates of the form

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

The inner products of y with all columns of the covariate matrix as well as all inner products of two columns of the covariate matrix can then be computed in the beginning and stored. At each coordinate update they can then be accessed.

Note that the coordinate descent algorithm always converges to the global minimum. The proof is contained in the following footnote.[4]

In the connection sign estimation step, we use a trick decomposing some of the elements and storing them. In a high-dimensional setup, there can be a large number of connected covariates. The 3CoSE algorithm relies on a small OLS model to update each single connection sign and can thus be computationally expensive if implemented straightforwardly. However, as the OLS models are similar in the repetitions of the algorithm, compute time can be saved as follows.

Denote by $\tilde{X} := \left(x_{(i)}, x_{(j)}\right)$, the input matrix consisting only of the two columns $x_{(i)}$ and $x_{(j)}$ and by $\tilde{\beta} := (\hat{\beta}_i^{*}, \hat{\beta}_j^{*})^T$ the OLS estimates of the small linear models described in Section 2.2 (the tildes should not be confused with the tildes in the coordinate descent algorithm). Then the solution of the small linear models can be written as

[4]The proof makes 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.

With a network penalty, the penalized RSS can be written 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.

In short, if $\beta^T M\beta$ is convex, the coordinate wise descent algorithm is sure to converge to the global minimum. That is in particular the case for all M which allow $\beta^T M\beta$ to be written as a sum of squared terms.

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

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

$\tilde{\beta}$ only changes with updates of the estimates $\hat{\beta}$, while the rest of the elements can be computed from the data alone. The elements that can be computed from the data alone are

$$\underbrace{B_y^{i,j}}_{2\times 1} := \left(\tilde{X}^\top\tilde{X}\right)^{-1}\tilde{X}^\top y \quad \text{and} \quad \underbrace{B_x^{i,j}}_{2\times(p-2)} := \left(\tilde{X}^\top\tilde{X}\right)^{-1}\tilde{X}^\top X^{(i,j)}.$$

By pre-computing these elements for each connected pair of covariates (i, j), each linear regression is simplified to a single multiplication and subtraction.

3 Simulations

The motivation for the simulations is of biomedical nature, where network information about gene expression data is of great interest. Such data, which is usually high-dimensional, can be used for prediction purposes (e.g., to predict event times of patients in medical applications), but the signs of the network connections are also of interest in themselves for biomedical research (as it is important to know which genes influence other genes in what ways). The simulations that we report are similar (but not identical) to the ones reported in Li and Li (2008) and Weber et al. (2014).

3.1 Setting

We consider two similar sets of simulations with four scenarios each. The difference between the two sets is that the first one contains fewer variables, 1100 covariates as compared to 2200 in the second set, while the number of non-zero covariate coefficients is identical in both sets. That is, in the first set there are relatively more informative covariates. We first describe the first set in detail and briefly mention the small modifications for the second set thereafter.

Note that of the four scenarios in each of the two sets of simulations the third and

fourth scenarios should be seen more as robustness checks. In these scenarios all connections between covariates are per definition positive. These scenarios do thus not promise any potential for the 3CoSE algorithm to do better than the penalty introduced by Li and Li (2008), which already assumes positive connections. Nevertheless, is is interesting to see whether the algorithm does significantly worse in case that all connections are indeed positive.

Simulations with 1100 covariates (100 transcription factors). Suppose that information is available about a regulatory network with 100 transcription factors and 1000 genes that are controlled by these transcription factors. Each transcription factor controls ten genes. The resulting network thus consists of 1100 genes and the connections between the transcription factors and the genes that they regulate. The covariate matrix X consists of 1100 columns, where the first column consists of the expression levels of the first transcription factor, the next ten columns are the expression levels of the genes regulated by the transcription factor in the first column, and so on.

The four different scenarios differ mainly in the true covariate coefficient vector β. We assume a linear model, i.e. $y_i = X_i\beta + \epsilon_i$. The number of observations is 100 and $\epsilon_i \sim N(0, \sigma^2)$ with $\sigma^2 = (\sum_j \beta_j^2)/4$. The expression levels of the 100 transcription factors are independently and identically distributed according to a standard normal distribution. The expression level of a gene depends on the level of the transcription factor that regulates it. The expression level of a gene of observation i that depends on the j-th transcription factor follows a $N(\mathbb{1} \cdot 0.7 \cdot TF_{i,j}, 0.51)$ distribution, where $\mathbb{1}$ is the indicator function, which depends on the connection sign and is equal to 1 or -1.

In the first scenario the true coefficient vector is of the following form:

$$\beta_1 = (5, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{3}, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{7}, 5, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{3}, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{7},$$
$$5, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{3}, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{7}, 5, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{3}, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{7}, \underbrace{0, \ldots, 0}_{1056}).$$

In the second scenario the denominators of $\sqrt{10}$ are replaced by 10, keeping all else equal. In these two scenarios, the indicator function $\mathbb{1}$ takes the value -1 for the first three regulated genes of a transcription factor, while it takes the value 1 for the other seven genes.

The coefficient vector in the third scenario is

$$\beta_3 = (5, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{10}, -5, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{10}, 5, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{10}, -5, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{10}, \underbrace{0, \ldots, 0}_{1056}).$$

The indicator function $\mathbb{1}$ in this scenario always equals 1 (this is intuitive as the coefficients of the transcription factors and the regulated genes always have the same sign). The fourth scenario is identical to the third one, except that the denominator values of $\sqrt{10}$ are replaced by 10.

Simulations with 2200 **covariates (**200 **transcription factors).** In addition to the first set of four scenarios described above, we also consider very similar scenarios with additional zero-coefficients. That is, instead of considering 100 transcription factors, we consider 200 transcription factors, again each one regulating ten genes. The four scenarios resemble the four previously discussed scenarios but with additional 1100 zero coefficients in the coefficient vector β. The coefficient vector in the first scenario, for example, is then as follows:

$$\begin{aligned}\beta_1 = (5, &\underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{3}, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{7}, 5, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{3}, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{7}, \\ &5, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{3}, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{7}, 5, \underbrace{\frac{5}{\sqrt{10}}, \ldots, \frac{5}{\sqrt{10}}}_{3}, \underbrace{\frac{-5}{\sqrt{10}}, \ldots, \frac{-5}{\sqrt{10}}}_{7}, \underbrace{0, \ldots, 0}_{2156}).\end{aligned}$$

3.2 Simulation Details and Definitions

For each scenario, we simulate 50 training data sets and 50 test data sets. In each training set, we select the lasso and the network penalty parameters with ten-fold cross validation. The regression coefficients and connection signs are then computed using the

full training data set and selected penalty parameters. Prediction mean squared errors are then calculated on one full test data set. Note that this means that we evaluate the prediction performance out of sample (which is completely different from just reporting a better fit in sample due to additional parameters or steps).

To evaluate the prediction performance, we use the following definitions:

1. Prediction mean squared error (k here indicates the index of the data set, $k = 1, \ldots, T$, and i indicates the observation number, $i = 1, \ldots, N$; in our case $T = 50$ and $N = 100$)

$$PMSE(y_k) = \frac{1}{N} \sum_{i=1}^{N} (y_{k,i} - \hat{y}_{k,i}) = \frac{1}{N} \sum_{i=1}^{N} \left(y_{k,i} - X_{k,i} \hat{\beta}_{k,i|\lambda_1,\lambda_2} \right)$$

$$PMSE(y) = \frac{1}{T} \sum_{k=1}^{T} (PMSE(y_k))$$

2. Standard errors, denoted by s, for the estimated variable of interest, i.e. $PMSE(y)$, are computed as follows

$$s = \frac{1}{T} \sqrt{\sum_{k=1}^{T} \left[PMSE(y_k) - PMSE(y) \right]^2}$$

The grid from which the possible values of λ_1 and λ_2 are chosen in the cross validation is the following:

$$\lambda_1 = \{\underbrace{600, \ldots, 100}_{\text{in steps of 100}}, \underbrace{99, \ldots, 20}_{\text{in steps of 3}}\}$$

$$\lambda_2 = \{\underbrace{0, \ldots, 100}_{\text{in steps of 5}}\}$$

This grid was chosen based on values found in a variety of small model simulations (these models were similar to the scenarios that we consider but with fewer covariates).

In the simulation studies, we compare the following methods. First, the regular 3CoSE algorithm, using the normalized Laplacian as penalty matrix (abbreviated unsurprisingly as 3CoSE in the tables). Second, the 3CoSE algorithm using the combinatorial Laplacian as penalty matrix (abbreviated as 3CoSE-CL). Third, penalized regression with a network

penalty as in (Li and Li, 2008), which is abbreviated as Net (Li Li) in the tables.[5] Fourth, we use the Lasso as a benchmark model, which incorporates information about the levels of the covariates but not about the network structure. As comparison, we also partly show a null model ignoring all covariate information (that is, always forecasting the intercept) and the true model (that is, forecasting with the β used to create the data).

3.3 Simulation Results with 1100 Covariates

Now, we consider the prediction mean square errors (PMSEs), sensitivity and specificity estimates, and the estimates of the fractions of correctly estimated connection signs for the simulations with 1100 covariates (100 transcription factors). Standard errors are always given in parentheses.

The prediction performance of the different methods is given in Table 1. The 3CoSE algorithm with the normalized Laplacian and with the combinatorial Laplacian perform much better than the other methods in the first scenario. In the other three scenarios there are no large differences in PMSEs. This includes the scenarios with all positive connection signs. Thus, the algorithm seems to be able to improve prediction performance considerably in some cases while not leading to worse performance even in the extreme cases with only positive connection signs.

Table 1: Prediction mean squared errors (PMSEs)

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso	True	Null
Scenario 1	**105.94**	**124.45**	134.72	136.49	53.44	1121.22
	(3.24)	(4.81)	(4.82)	(4.34)	(1.05)	(22.11)
Scenario 2	52.58	**51.75**	**51.75**	61.3	28.11	320.79
	(1.45)	(1.38)	(1.38)	(1.97)	(0.56)	(7.17)
Scenario 3	**104.18**	115.33	**100.25**	131.88	52.28	1150.07
	(4.13)	(4.13)	(4.25)	(4.26)	(0.93)	(23.46)
Scenario 4	**50.78**	**50.17**	51.64	56.85	27.2	315.56
	(1.58)	(1.39)	(1.36)	(1.58)	(0.46)	(6.45)

Measures of sensitivity and specificity can be found in Tables 2 and 3). The sensitiv-

[5] We do not apply any double-shrinkage correction, which is mentioned in Li and Li (2008). Unlike in the elastic net (Zou and Hastie, 2005), where one may indeed talk of double shrinkage, the additional network penalty pulls different coefficients toward each other and not toward zero.

ity 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. 3CoSE identifies non-zero coefficients similarly well as Net (Li Li) in most scenarios and considerably better in the first one. The standard algorithm with the normalized Laplacian in general performs better than the one with the combinatorial Laplacian, which still performs quite well. Specificity estimates are similar for 3CoSE and Net (Li Li) in all scenarios, while the values are very close to one for the Lasso, which estimates many variables to be zero (reflected in very good specificity but very poor sensitivity).

Table 2: Sensitivity estimates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	**0.93**	**0.75**	0.67	0.52
	(0.02)	(0.03)	(0.02)	(0.01)
Scenario 2	**0.47**	**0.41**	0.41	0.26
	(0.03)	(0.01)	(0.01)	(0.01)
Scenario 3	**0.94**	0.84	**0.96**	0.52
	(0.02)	(0.03)	(0.02)	(0.01)
Scenario 4	**0.46**	0.42	**0.57**	0.27
	(0.02)	(0.02)	(0.04)	(0.01)

Table 3: Specificity estimates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	0.94	0.95	**0.95**	**0.99**
	(0.0023)	(0.0026)	(0.0025)	(0)
Scenario 2	0.96	0.96	**0.96**	**1**
	(0.003)	(0.0029)	(0.0029)	(0)
Scenario 3	0.94	**0.94**	0.94	**0.99**
	(0.0026)	(0.0025)	(0.0028)	(0)
Scenario 4	0.96	**0.97**	0.96	**1**
	(0.0026)	(0.0025)	(0.0029)	(0)

Furthermore, we compute the fractions of correctly estimated connections signs. We consider the connection sign to be correctly identified if the coefficient estimates of the two connected covariates are non-zero and either both estimates and both true coefficients have the same signs or if both have different signs (this allows us to also talk about correctly identified connections for Net (Li Li) and the Lasso, although these methods

are not intended to estimate network connection signs. As can be seen in Table 4, 3CoSE performs much better than Net (Li Li) in the first scenario, similarly in the second and third and a bit worse in the fourth (remember, however, that scenarios 3 and 4 are designed as favorably to Net (Li Li) as possible as all connection signs are positive). The Lasso does much worse in identifying the connection signs than the other methods across the different scenarios.

Table 4: Fractions of correctly estimated connection signs

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	**0.95**	**0.82**	0.8	0.66
	(0.03)	(0.05)	(0.06)	(0.07)
Scenario 2	**0.62**	**0.58**	0.58	0.48
	(0.07)	(0.07)	(0.07)	(0.07)
Scenario 3	**0.94**	0.84	**0.96**	0.52
	(0.03)	(0.05)	(0.03)	(0.07)
Scenario 4	**0.46**	0.42	**0.57**	0.27
	(0.07)	(0.07)	(0.07)	(0.06)

The 3CoSE algorithm converges in all simulations and scenarios. In the first scenario, only in 10% of cases the algorithm needs more than one iteration to converge. In the second scenario, convergence happens after the first iteration in 36% of cases, while in the third and fourth scenarios the algorithm converges after the first iteration in all cases. Tables 3.3 and 3.3 show the parameters that were selected via cross-validation in the different scenarios for the different methods. Average selected λ_1 parameters for the first three methods (3CoSE, 3CoSE-CL and Net(Li Li)) are between 83 and 135, while for the lasso it is always 400. The selected network penalty parameter λ_2 varies across methods and simulations in a range from 0 to 71.

Table 5: Average selected λ_1 values

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	84.36	90.40	99.06	400
Scenario 2	108.30	118.62	118.62	400
Scenario 3	83.30	85.36	89.70	400
Scenario 4	128.60	134.08	109.86	400

Table 6: Average selected λ_2 values

	3CoSE	3CoSE-CL	Net (Li Li)
Scenario 1	21.30	8.50	3.60
Scenario 2	1.60	0.00	0.00
Scenario 3	27.30	14.70	70.60
Scenario 4	1.70	0.70	19.60

3.4 Simulation Results with 2200 Covariates

The results for simulations with 200 transcription factors (and thus 2200 covariates) are similar to the results for the simulations with 100 transcription factors. These results are reported in Tables 7 to 12.

Table 7: Prediction mean squared errors (PMSEs), 2200 covariates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso	True	Null
Scenario 1	**123.17**	**139.28**	146.92	144.94	51.59	1140.37
	(6.15)	(5.75)	(6.12)	(6.1)	(0.98)	(27.61)
Scenario 2	52.27	**52.03**	**51.89**	57.82	27.32	308.51
	(1.69)	(1.66)	(1.65)	(1.95)	(0.55)	(6.33)
Scenario 3	**118.61**	131.76	**111.98**	136.6	51.65	1152.17
	(4.21)	(4.47)	(3.94)	(4.43)	(0.99)	(23.61)
Scenario 4	**52.15**	**50.93**	53.18	58.26	27.89	315.07
	(2.06)	(1.99)	(2.07)	(2.75)	(0.61)	(6.59)

Table 8: Sensitivity estimates, 2200 covariates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	**0.86**	**0.72**	0.6	0.48
	(0.03)	(0.03)	(0.02)	(0.01)
Scenario 2	**0.4**	**0.37**	0.37	0.25
	(0.02)	(0.01)	(0.01)	(0.01)
Scenario 3	**0.84**	0.69	**0.92**	0.49
	(0.03)	(0.03)	(0.02)	(0.01)
Scenario 4	**0.44**	0.39	**0.54**	0.27
	(0.03)	(0.01)	(0.04)	(0.01)

In short, in terms of PMSEs, 3CoSE performs again extremely well in the first scenario, while prediction errors are similar to those of Net (Li Li) in the other scenarios. The sensitivity and specificity estimates are comparable to the ones in the simulations with 100 transcription factors, meaning that the sensitivity is considerably better in the first

Table 9: Specificity estimates, 2200 covariates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	0.97	0.97	**0.98**	**0.99**
	(0.0013)	(0.0012)	(0.0011)	(0)
Scenario 2	0.98	0.98	**0.98**	**1**
	(0.0014)	(0.0014)	(0.0014)	(0)
Scenario 3	0.97	**0.97**	0.97	**0.99**
	(0.0011)	(0.0013)	(0.0012)	(0)
Scenario 4	0.98	**0.98**	0.97	**1**
	(0.0015)	(0.0015)	(0.0015)	(0)

Table 10: Fractions of correctly estimated connection signs, 2200 covariates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	**0.91**	**0.8**	0.74	0.63
	(0.04)	(0.06)	(0.06)	(0.07)
Scenario 2	**0.59**	**0.57**	0.56	0.48
	(0.07)	(0.07)	(0.07)	(0.07)
Scenario 3	**0.84**	0.69	**0.92**	0.49
	(0.05)	(0.07)	(0.04)	(0.07)
Scenario 4	**0.44**	0.39	**0.54**	0.27
	(0.07)	(0.07)	(0.07)	(0.06)

Table 11: Average selected λ_1 values, 2200 covariates

	3CoSE	3CoSE-CL	Net (Li Li)	Lasso
Scenario 1	111.82	112.24	146.34	400.00
Scenario 2	109.76	114.88	115.48	400.00
Scenario 3	91.80	119.16	86.18	400.00
Scenario 4	103.52	109.60	91.52	400.00

Table 12: Average selected λ_2 values, 2200 covariates

	3CoSE	3CoSE-CL	Net (Li Li)
Scenario 1	19.40	6.40	1.80
Scenario 2	1.20	0.10	0.10
Scenario 3	14.60	8.10	56.70
Scenario 4	0.90	0.00	15.30

scenario for 3CoSE than for Net (Li Li) while differences are relatively small in the other scenarios. Specificity is similar between the methods. The Lasso again estimates many variables to be zero leading to high specificity but low sensitivity.

Again, in all cases the algorithm converged. Convergence took place similarly fast as in the set with 1100 covariates suggesting that the number of covariates is not a driving factor in determining the rate of convergence of the algorithm. In the first two scenarios, for example, convergence took more than one iteration (of the connection sign estimation step) in only 16 % and 32% of the cases, respectively. It is also worth mentioning here that in both sets of simulations, the algorithm converged after at most 4 iterations. Average selected penalty parameters in the second set of simulations are also similar to the ones in the regressions with 1100 covariates, with slightly lower values for λ_2 in the 2200 covariate setup.

4 Summary

This paper shows that incorporating network information into regularized regression via the introduced 3CoSE algorithm can yield to improved prediction performance. This algorithm can furthermore contribute to discovering signs of network connections when these are unknown. This can be helpful in a variety of fields where networks play an important role, including biology (from which we have borrowed the motivation of the conducted simulation studies), economics, finance, and computer science. We make the accompanying R-package LassoNet (Striaukas and Weber, 2018) publicly available, in the hope that this method will be more widely applied in different fields. The availability of the R-package may also facilitate refining the method to adapt it to different settings.

References

Chung, F. (1997): "Spectral Graph Theory," *CBMS Regional Conferences Series (Vol. 92), American Mathematical Society.*

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

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

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

Striaukas, J. and M. Weber (2018): "LassoNet: Implementation of the 3CoSE Algorithm," R-package, available online at https://github.com/jstriaukas/LassoNet.

Tseng, P. (1988): "Coordinate ascent for maximizing nondifferentiable concave functions," Tech. rep., Massachusetts Institute of Technology, Laboratory for Information and Decision Systems.

Weber, M., M. Schumacher, and H. Binder (2014): "Regularized Regression Incorporating Network Information: Simultaneous Estimation of Covariate Coefficients and Connection Signs," Tinbergen Institute Discussion Paper 14-089/1.

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