Estimation of Covariance Matrix via the Sparse Cholesky Factor with Lasso

Changgee Chang[a,1], Ruey S. Tsay[*,b,2]

[a] *Department of Statistics, University of Chicago, Chicago, Illinois, USA*
[b] *Booth School of Business, University of Chicago, Chicago, Illinois, USA*

Abstract

In this paper, we discuss a parsimonious approach to estimation of high-dimensional covariance matrices via the modified Cholesky decomposition with lasso. Two different methods are proposed. They are the equi-angular and equi-sparse methods. We use simulation to compare the performance of the proposed methods with others available in the literature, including the sample covariance matrix, the banding method, and the L_1-penalized normal loglikelihood method. We then apply the proposed methods to a portfolio selection problem using 80 series of daily stock returns. To facilitate the use of lasso in high-dimensional time series analysis, we develop the dynamic weighted lasso (DWL) algorithm that extends the LARS-lasso algorithm. In particular, the proposed algorithm can efficiently update the lasso solution as new data become available. It can also add or remove explanatory variables. The entire solution path of the L_1-penalized normal loglikelihood method is also constructed.

Key words: Adding and removing variables, Covariance matrix estimation, Equi-angular, Dynamic weighted lasso, L_1 penalty, Lasso, Updating, Modified Cholesky decomposition

1. Introduction

Estimating a high-dimensional covariance matrix with limited data is a difficult problem because the matrix often contains many parameters. For an $m\times m$ covariance matrix Σ, there are $m(m+1)/2$ parameters, yet the sample size n is often small. In addition, the positive-definiteness of Σ makes the problem even more complicated. The sample covariance matrix is positive-definite and unbiased if $n > m$, but it only works well for the low-dimensional problem. As the dimension m increases, the sample covariance matrix tends to become unstable. As shown by Yin [21], the sample covariance matrix can even fail to be consistent if $m/n \nrightarrow 0$. On the other hand, modern statistical applications often encounter high dimensionality with a limited number of data points. See, for instance, problems in image processing, longitudinal data analysis,

[*] Corresponding author: Ruey S. Tsay, Booth School of Business, University of Chicago, 5807 S. Woodlawn Avenue, Chicago, IL 60637.

Email addresses: `changgee@uchicago.edu` (Changgee Chang), `ruey.tsay@chicagobooth.edu` (Ruey S. Tsay)

[1] Research supported in part by the ROC fund, Center for East Asian Studies, University of Chicago.

[2] Research supported in part by Booth School of Business, University of Chicago.

machine learning, and gene array analysis. Many methods to covariance matrix estimation are available in the literature if m is small relative to n, including the spectral decomposition, Bayesian methods, modeling the matrix-logarithm, nonparametric smoothing, and banding/thresholding techniques. See, for example, Bickel and Levina [1] [2], Boik [3], Chiu et al. [4], Diggle and Verbyla [7], Leonard and Hsu [11], and Yang and Berger [20].

In some applications, one needs the inverse covariance matrix rather than the covariance matrix itself. Consider, for instance, the question of asset allocation in finance. Solutions to the portfolio selection problem of Markowitz [13] and many mean-variance type of financial problems are often written in terms of the inverse covariance matrix. Furthermore, due to the time-varying nature of stock returns, the covariance structure of return series is often estimated using only the most recent data, resulting in a small sample size compared to the number of parameters to be estimated. In many cases, the sparse structure of the inverse covariance matrix has attracted special interest because zero correlations represent conditional independence between the variables, which is a major concern in some scientific fields, e.g., the graphical model. Dempster [6] parsimoniously estimated the entries of the inverse covariance matrix, treating them as the canonical parameters of a multivariate normal density, Wong et al. [18] used a Bayesian approach to estimation of the inverse covariance matrix, and d'Aspremont et al. [5], Meinshausen and Bühlmann [14], and Yuan and Lin [22] identified the sparse elements in the inverse covariance matrix by imposing the lasso type of penalty. But dealing with the elements of the inverse covariance matrix may encounter the difficulty of the positive definiteness, and often entails heavy computation. Some of the aforementioned papers indeed provide computationally relaxed approximations.

Pourahmadi [15] employed the modified Cholesky decomposition, which reparameterizes the inverse covariance matrix. The approach not only can easily guarantee the positive definiteness of a covariance matrix, but also transforms the problem of estimating covariance matrices into one that employs $m-1$ linear regressions. The modified Cholesky decomposition to covariance matrix estimation has been employed in Bickel and Levina [1], Huang et al. [10], Levina et al. [12], Tsay [17] and Wu and Pourahmadi [19], among others. In this paper, we also focus on methods based on the modified Cholesky decomposition (assuming $n > m$) and seek to estimate entries of the Cholesky factor parsimoniously. The parsimony is achieved by a series of lasso regressions proposed by Tibshirani [16] and known as the L_1-penalized least squares method. Specifically, we discuss how to fairly penalize the $m-1$ lasso regressions with a single penalizing parameter and propose two penalizing methods.

The first proposed method is the equi-angular method inspired by the least angle regression (LAR) of Efron et al. [8]. Adopting the framework of Fan and Li [9], Huang et al. [10] proposed an estimation method that L_p-penalizes the loglikelihood of Σ for normally distributed data. The equi-angular method is a L_1-penalized least squares method and, hence, is similar to the L_1-penalized normal loglikelihood method. But the two methods differ in choosing the penalties for the $m-1$ lasso regressions involved. The equi-angular method uses penalties proportional to the residual standard deviations of the lasso regressions. That is, if the size of a residual of a lasso regression is twice larger than that of another lasso regression, we impose twice heavier penalty on the former than the latter. We show that the L_1-penalized normal loglikelihood method is equivalent to choosing the lasso penalties being proportional to the corresponding residual variance. So in the same situation, it assigns four times larger penalty to the former regression than the latter one. We discuss situations under which the proposed method is fair and reasons for calling it the equi-angular method. The other method we propose is the equi-sparse method. We define the degree of sparsity of a lasso regression as the ratio of the current penalty over the minimum lasso penalty needed to make all coefficients zero, and all lasso regressions synchronize their degree of sparsity. By so doing, if the degree of sparsity is 1, we have all lasso regressions fully penalized and obtain the identity matrix for the Cholesky factor, and if the degree of sparsity is 0, the lasso regressions are not penalized at all and we obtain the sample covariance matrix. When the degree of sparsity is between 0 and 1, we have the Cholesky factor between the two extremes. The idea behind the equi-sparse method is that, if a lasso regression is penalized to some degree, then the same should be applied to the other lasso regressions, and this method is expected to work well when all the regressions are similar.

We compare via simulation and empirical analysis the proposed two methods with the L_1-penalized normal loglikelihood method and the banding method of Bickel and Levina [1]. The simulation study shows that the proposed equi-angular method outperforms the other methods in general, and the equi-sparse method does slightly better for covariance matrices whose regression structures are alike. Since the modified Cholesky decomposition is not permutation invariant, we also investigate the sensitivity to permutation of each method by randomly permuting the variables before estimation. The result shows that the equi-angular method still outperforms the others. As a real-world application, the optimal portfolio selection problem in the finance literature is considered. We perform covariance matrix estimation by various methods and use the estimated covariance matrices to select the global minimum variance (GMV) portfolios. The portfolios are updated monthly and their monthly out-of-sample performance is compared.

The paper also develops a new algorithm for solving the lasso problem for serially dependent data. Efron et al. [8] developed the LARS-lasso algorithm in light of the least angle regression and showed that the whole solution path of the lasso is piecewise linear with respect to its penalty. A drawback of the LARS-lasso algorithm is that it only supports fixed weights for the penalties. Although the LARS-lasso algorithm was written only for homogeneous penalties for all coefficients, we may continue to use the LARS-lasso algorithm when weighted penalties are used for different coefficients. This is achieved by re-scaling the variables so that giving the same penalty for all coefficients has the same effect; see, for example, Zou [23]. However, it is impossible to change the weights in the LARS-lasso algorithm. Furthermore, it is unclear how to efficiently update the lasso solution when a new data point becomes available. In such cases, even if the solution is almost unchanged, one needs to perform the LARS-lasso algorithm from the beginning. These deficiencies are major obstacles in applying lasso to the high-dimensional time series analysis. To overcome the difficulties, we propose a new algorithm that extends the LARS-lasso algorithm. We call it the dynamic weighted lasso (DWL) algorithm. The time complexity of the DWL algorithm is exactly the same as that of the LARS-lasso algorithm. Indeed, the two algorithms are twins if the lasso penalties are homogeneous. But the DWL algorithm allows weighted penalties, can change the weights without rebuilding the solution from the beginning, and accepts more flexible initial points. Consequently, the proposed new algorithm can efficiently update the lasso solution. We show that substantial saving in running time is obtained by using the updating algorithm. Moreover, the new algorithm can efficiently add or remove variables in the lasso.

We provide a fitting algorithm for the equi-angular method, which is based on the DWL (or the LARS-lasso) algorithm. The whole solution path of each lasso regression with respect to the common penalizing parameter η is the same as the ordinary lasso solution path with respect to the lasso penalty λ, and there is a one-to-one correspondence between η and λ, which implies the solution of the equi-angular method exists uniquely. Since the mapping between η and λ is analytically tractable, the solution for a particular η is immediately available when the lasso solution path is available. We also discuss ways to find a particular solution without generating the entire solution path.

Finally, we investigate the whole solution path of the L_1-penalized normal loglikelihood method. This not only provides a justification for the algorithm we used, but also has its own independent research interest. The solution of the L_1-penalized normal loglikelihood can have multiple local solutions due to its nonconvexity, and an iterative quadratic approximation technique has been used in the literature; see Fan and Li [9]. However, it turns out that the solution path of the L_1-penalized normal loglikelihood method is a subset of the lasso solution path, and not only the global optimal solution but all local optima are immediately available when the lasso solution path is available. This leads to a much faster algorithm for solving the L_1-penalized normal loglikelihood problem.

The paper is organized as follows. In Section 2, we briefly review the modified Cholesky decomposition and define our covariance matrix estimators. Their fitting algorithms are also provided. The L_1-penalized normal loglikelihood method of Huang et al. [10] is compared and its solution path is investigated. In Section 3, we compare via simulation the proposed methods with the banding method, Huang et al.'s method, and the sample covariance matrix. Section 4 contains the empirical analysis of portfolio selection using daily stock returns. The DWL algorithm is presented and its derivatives are derived in Section 5. Section 6 concludes the paper with some discussions.

2. The Estimation Methods

2.1. The Modified Cholesky Decomposition

For completeness, this subsection briefly reviews the modified Cholesky decomposition (e.g., Pourahmadi [15]). Suppose Σ is an $m \times m$ positive-definite matrix and let $y = (y_1, \ldots, y_m)'$ be a random vector with mean zero and covariance matrix Σ. Let $\phi_{j,1}, \ldots, \phi_{j,j-1}$ be the coefficients of the least-squares predictors for y_j based on $y_1, \ldots, y_{j-1}$ and ε_j be the prediction error. Then we have

$$y_j = \sum_{k=1}^{j-1} \phi_{jk} y_k + \varepsilon_j. \tag{1}$$

Let T be a unit lower triangular matrix with $T_{jk} = -\phi_{jk}$ for $k < j$ and $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_m)'$. Then (1) becomes

$$Ty = \varepsilon. \tag{2}$$

Since ε_{j-1} depends only on y_1 through y_{j-1} and ε_j is uncorrelated with y_1 through y_{j-1}, all ε_j's are uncorrelated and hence $\mathrm{cov}(\varepsilon) = \mathrm{diag}\big(\sigma_1^2, \ldots, \sigma_m^2\big) = D$. Therefore, it follows from (2) that

$$T \Sigma T' = D.$$

This is called the modified Cholesky decomposition of Σ. ϕ_{jk}'s are called the generalized autoregressive parameters, σ_j^2's are the corresponding innovation or the residual variances, and T is the Cholesky factor.

2.2. The Proposed Estimators

The modified Cholesky decomposition reparameterizes Σ or Σ^{-1} using ϕ_{jk}'s and σ_j^2's, and it transforms the covariance matrix estimation problem into a regression coefficient estimation problem. It enables us to consider every effort that has been devoted to estimation of regression coefficients as a potential solution to the covariance matrix estimation problem.

To efficiently estimate covariance matrices with parsimonious Cholesky factor T, we consider the lasso regression, which is known for its usefulness in variable selection and shrinkage. The lasso regression (Tibshirani [16]) is formulated as

$$\text{minimize} \quad \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \|\boldsymbol{\beta}\|_1, \tag{3}$$

where $\mathbf{y}$ is the response vector, $\mathbf{X}$ is a design matrix, $\boldsymbol{\beta}$ is the vector of regression coefficients, $\|\boldsymbol{\beta}\|_1 = \sum_j |\beta_j|$, and λ is the penalizing factor with larger values of λ giving sparser and more parsimonious regression coefficient estimates.

We run $m-1$ lasso regressions to estimate ϕ_{jk}'s and σ_j^2's of the modified Cholesky decomposition. Note that the explanatory variables in a lasso regression are usually normalized for the sake of fair penalizing.

But in this paper, every variable stays in its own scale, and we use the weighted penalties. The j-th lasso regression becomes

$$\text{minimize} \quad \|\mathbf{y}_j - \mathbf{Y}_j \boldsymbol{\phi}_j\|^2 + \lambda_j \sum_{k=1}^{j-1} w_k |\phi_{jk}|, \quad j = 2, \ldots, m, \tag{4}$$

where $\mathbf{y}'_j = \big(y_j^1, \ldots, y_j^n\big)$ is the observation vector for the j-th variable, $\mathbf{Y}_j = [\mathbf{y}_1, \ldots, \mathbf{y}_{j-1}]$ and $\boldsymbol{\phi}_j = (\phi_{j,1}, \ldots, \phi_{j,j-1})'$, and $w_k^2 = \|\mathbf{y}_k\|^2/n$ for $k = 1, \ldots, m$. The solution $\widehat{\boldsymbol{\phi}}_j$ of (4) is the estimator for $\boldsymbol{\phi}_j$ and

$$\widehat{\sigma}_j^2 = \frac{1}{n} \big\|\mathbf{y}_j - \mathbf{Y}_j \widehat{\boldsymbol{\phi}}_j\big\|^2 \tag{5}$$

is the estimator for the residual variance σ_j^2.

Since we use different penalty factors λ_j for the $m-1$ regressions, we encounter the issue of how to fairly balance λ_j in penalizing the $m-1$ regressions. There might be other ways to address the issue, but in this paper we propose a method that extends the spirit of lasso. The lasso gives parsimonious regression coefficient estimates by making all variables corresponding to nonzero coefficients evenly correlated with the residual and by zeroing the other coefficients whose corresponding variables are less correlated with the residual. See Proposition 5.1 below or Efron et al. [8].

To make it more specific, assume that $\widehat{\boldsymbol{\phi}}_j$ is the solution of (4), and let $\mathcal{A}_j$ be the index set of nonzero coefficients in $\widehat{\boldsymbol{\phi}}_j$. Then it follows by Proposition 5.1 that

$$2\big|\mathbf{y}'_k\big(\mathbf{y}_j - \mathbf{Y}_j \widehat{\boldsymbol{\phi}}_j\big)\big| = \lambda_j w_k, \quad k \in \mathcal{A}_j.$$

Observe that the LHS of the above equation is proportional to the size of the residual. So, if the true residual variance σ_j^2 is large, then the j-th lasso regression is relatively less penalized. Conversely, if σ_j^2 is small, then the j-th lasso regression is penalized more heavily. Therefore, it would be fair to make λ_j proportional to the residual standard error $\widehat{\sigma}_j$. That is, we propose

$$\lambda_j(\eta) = \eta \widehat{\sigma}_j, \tag{6}$$

for some common penalty factor $\eta > 0$. By so doing, we can indeed achieve the balance condition

$$2\big|\mathrm{cor}\big(\mathbf{y}_k, \mathbf{y}_j - \mathbf{Y}_j \widehat{\boldsymbol{\phi}}_j\big)\big| = \eta/n, \quad k \in \mathcal{A}_j, 2 \le j \le m,$$

which can be interpreted as the k-th variable and the j-th residual are equally correlated or equally angled for every pair of j and k for which $\widehat{\phi}_{jk}$ is nonzero. We call the resulting covariance matrix $\widehat{\Sigma}_a(\eta)$ the equi-angular covariance matrix.

Alternatively, we consider another method which also uses the lasso regression but balances λ_j differently. We introduce the degree of sparsity ν, which assumes a value in [0,1] and governs the entire sparsity of the $m-1$ lasso regressions. In particular, $\nu = 0$ means that every regression is not penalized at all and the

Cholesky factor becomes that of the sample covariance matrix. On the other hand, $\nu = 1$ means that all regressions are fully penalized and the Cholesky factor becomes the identity matrix. For $0 < \nu < 1$, we linearly interpolate the penalties for the two extreme cases. So, our choice of λ_j given ν becomes

$$\lambda_j(\nu) = 2\nu \max_{1 \le k < j} |\mathbf{y}_k' \mathbf{y}_j| / w_k. \tag{7}$$

We call the resulting covariance matrix $\widehat{\Sigma}_s(\nu)$ the equi-sparse covariance matrix and expect that the estimate fares well if the regressions in the modified Cholesky decomposition are similar.

2.3. Computation

In Section 5, we provide the DWL algorithm that extends the LARS-lasso algorithm of Efron et al. [8]. Readers are referred to the section for details of our estimation algorithm. Here we briefly discuss the computation associated with the two proposed estimation methods. The computation for the equi-sparse covariance matrix is straightforward. We can construct the whole solution path or any particular solution with respect to λ_j using the DWL algorithm, and hence we can do the same thing with respect to ν by (7).

The situation is similar for the equi-angular covariance matrix except that the relation between η and λ_j becomes a little more complicated. If we have obtained the whole solution path with respect to λ_j, we are able to find λ_j satisfying (6) because the estimated residual variance $\widehat{\sigma}_j^2(\lambda_j)$ is a tractable function of λ_j. Suppose that $\widehat{\boldsymbol{\phi}}_j$ is the solution with lasso penalty λ_j and let $\mathcal{A}_j$ and $\mathbf{S}_{\mathcal{A}_j}$ be its nonzero index set and the corresponding sign matrix, respectively. Then, it follows from (5) and (19) that

$$\widehat{\sigma}_j^2(\lambda_j) = \frac{1}{n}\Big(\mathbf{y}_j'\mathbf{y}_j - \mathbf{y}_j'\mathbf{Y}_{\mathcal{A}_j}\left(\mathbf{Y}_{\mathcal{A}_j}'\mathbf{Y}_{\mathcal{A}_j}\right)^{-1}\mathbf{Y}_{\mathcal{A}_j}'\mathbf{y}_j\Big) + \frac{1}{4n}\lambda_j^2 \mathbf{w}_{\mathcal{A}_j}'\mathbf{S}_{\mathcal{A}_j}\left(\mathbf{Y}_{\mathcal{A}_j}'\mathbf{Y}_{\mathcal{A}_j}\right)^{-1}\mathbf{S}_{\mathcal{A}_j}\mathbf{w}_{\mathcal{A}_j}. \tag{8}$$

Hence, the estimated residual variance is piecewise quadratic in λ_j, and its entire curve is analytically available if the whole solution path is available. Now assume further that $\widehat{\boldsymbol{\phi}}_j$ is the solution we seek for η. Then, (6) implies that λ_j should be

$$\lambda_j = \sqrt{\frac{4\eta^2\Big(\mathbf{y}_j'\mathbf{y}_j - \mathbf{y}_j'\mathbf{Y}_{\mathcal{A}_j}\left(\mathbf{Y}_{\mathcal{A}_j}'\mathbf{Y}_{\mathcal{A}_j}\right)^{-1}\mathbf{Y}_{\mathcal{A}_j}'\mathbf{y}_j\Big)}{4n - \eta^2\mathbf{w}_{\mathcal{A}_j}'\mathbf{S}_{\mathcal{A}_j}\left(\mathbf{Y}_{\mathcal{A}_j}'\mathbf{Y}_{\mathcal{A}_j}\right)^{-1}\mathbf{S}_{\mathcal{A}_j}\mathbf{w}_{\mathcal{A}_j}}}. \tag{9}$$

Note that (8) implies that $\lambda_j/\widehat{\sigma}_j(\lambda_j)$ is continuous and strictly monotone regardless of $\mathcal{A}_j$. Therefore, λ_j uniquely exists for each η, which means our equi-angular covariance matrix is unique. Figure 1(a) shows a simple possible curve for $\widehat{\sigma}_j^2(\lambda_j)$ in solid line and the curves of λ_j^2/η^2 for some η's in dashed lines. We can see the one-to-one correspondence between η and λ_j.

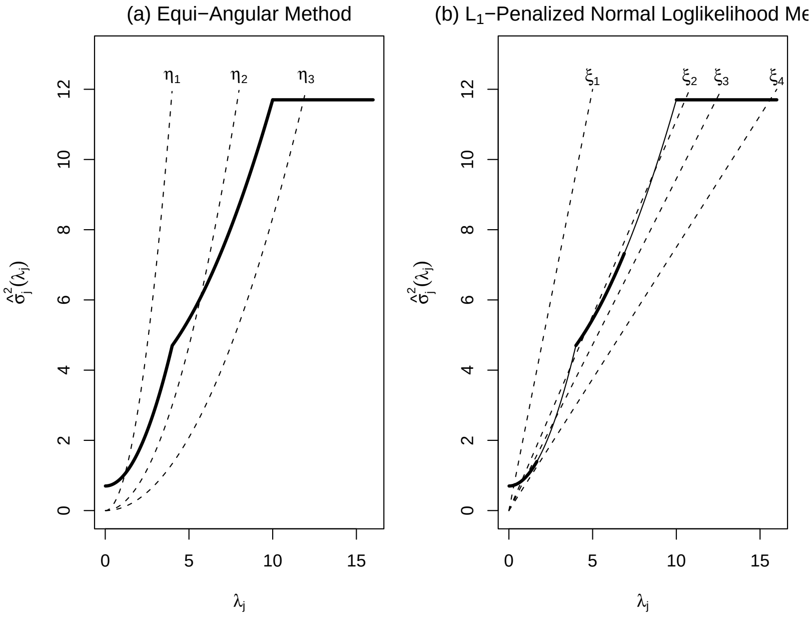
To obtain the solution for a single η without having to generate the whole solution path, one encounters the problem that the correct $\mathcal{A}_j$ is not known beforehand. The definition of λ_j in (6) depends on the solution, and therefore λ_j is not available either before the estimation is actually carried out. In fact, the choice of λ_j in (6) makes (4) no longer a classical lasso regression and we cannot obtain the solution from the original DWL algorithm. But with some simple modification to the DWL algorithm, we can overcome the difficulty at no extra cost. Due to the continuity and monotonicity of $\lambda_j/\widehat{\sigma}_j(\lambda_j)$, it is always clear whether to increase or decrease λ_j to get closer to the solution. If $\lambda_j/\widehat{\sigma}_j(\lambda_j) < \eta$, we increase λ_j, and conversely if $\lambda_j/\widehat{\sigma}_j(\lambda_j) > \eta$, we decrease λ_j. Then we will find the correct $\mathcal{A}_j$ where the inequality changes its direction, and can obtain the exact λ_j by (9).

2.4. Solution Path for L_1-Penalized Normal Loglikelihood Estimator

Following the framework of Fan and Li [9], Huang et al. [10] proposed the L_p-penalized loglikelihood method for normally distributed data. Let $\mathbf{y}^i = (y_1^i, \ldots, y_m^i)'$ be the i-th observation of a normal random vector with mean $\mathbf{0}$ and covariance matrix Σ. Suppose that the sample size is n. Then the loglikelihood of Σ is given by

$$\begin{aligned} -2l\big(\Sigma; \mathbf{y}^1, \ldots, \mathbf{y}^n\big) &= n \log |D| + \sum_i \big(\mathbf{y}^i\big)' T' D^{-1} T \mathbf{y}^i \\ &= n \sum_j \log \sigma_j^2 + \sum_{i,j} \frac{\big(\varepsilon_j^i\big)^2}{\sigma_j^2}, \end{aligned}$$

Figure 1: The solution paths of the equi-angular method and the L_1-penalized normal loglikelihood method. The solid (thinner) curve represents $\sigma_j^2(\lambda_j)$ of the lasso. The thicker curve means the set of possible solutions for each method. (a) The dashed curves are of λ_j^2/η^2. Every lasso solution is a solution for an η, and every η has a corresponding lasso solution. (b) The dashed lines are of λ_j/ξ. Only a part of the lasso solutions can be a solution for a ξ (proposition 2.1), and a single ξ can have multiple local solutions.



where $\varepsilon_1^i = y_1^i$ and $\varepsilon_j^i = y_j^i - \sum_{k<j} \phi_{jk} y_k^i$ for $j = 2, \ldots, m$. These authors obtained the estimates for ϕ_{jk} and σ_j^2 by minimizing

$$-2l\bigl(\Sigma; \mathbf{y}^1, \ldots, \mathbf{y}^n\bigr) + \xi \sum_{k<j} |\phi_{jk}|^p, \tag{10}$$

where $p \geq 1$. Since the penalty term does not involve σ_j^2, the optimal choice of σ_j^2 is

$$\widehat{\sigma}_j^2 = \frac{1}{n} \|\mathbf{y}_j - \mathbf{Y}_j \boldsymbol{\phi}_j\|^2,$$

and the rest of the problem reduces to

$$\text{minimize} \quad n \log \|\mathbf{y}_j - \mathbf{Y}_j \boldsymbol{\phi}_j\|^2 + \xi \sum_{k=1}^{j-1} |\phi_{jk}|^p, \tag{11}$$

for each $j = 2, \ldots, m$. It is easy to see that the (local) solution $\widehat{\boldsymbol{\phi}}_j$ of (11) also minimizes

$$\|\mathbf{y}_j - \mathbf{Y}_j \boldsymbol{\phi}_j\|^2 + \lambda_j \sum_{k=1}^{j-1} |\phi_{jk}|^p, \tag{12}$$

where $\lambda_j = \frac{\xi}{n} \bigl\|\mathbf{y}_j - \mathbf{Y}_j \widehat{\boldsymbol{\phi}}_j\bigr\|^2$. Therefore, the case of $p = 1$ is similar to the proposed equi-angular method except that the penalty parameter becomes

$$\lambda_j(\xi) = \xi \widehat{\sigma}_j^2. \tag{13}$$

The L_1-penalized normal loglikelihood method is, therefore, different from the equi-angular method, unless σ_j^2 are homogeneous. We will assume $w_k^2 = \|\mathbf{y}_k\|^2/n$ for the Huang et al.'s method, otherwise the variables should be normalized. The difference between (6) and (13) cannot be overlooked.

The prior discussion also unveils the whole solution path of the L_1-penalized normal loglikelihood estimators. Because (11) is not convex, its solution suffers from multiple local minima and the solution path with respect to ξ is intrinsically discontinuous. But the fact that the (local) solution of (11) also minimizes (12) says that we can construct the solution path of (11), including all local minima as well as the global minimum, via the solution path of the lasso problem (12). And (13) provides the clue needed to map the penalty parameters of the two different problems.

To state the following proposition, we rewrite the problems (11) and (12) as follows.

$$\text{minimize} \quad n \log \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \xi \sum_k w_k |\beta_k|, \tag{14}$$

$$\text{minimize} \quad \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \sum_k w_k |\beta_k|. \tag{15}$$

Proposition 2.1. *Suppose that $\widehat{\boldsymbol{\beta}}$ is the solution of* (15) *with penalty λ and let $\mathcal{A}$ and $\mathbf{S}_\mathcal{A}$ be its nonzero index set and the corresponding sign matrix, respectively. If $\widehat{\boldsymbol{\beta}}$ is a breakpoint with $\bigl|c_l\bigl(\widehat{\boldsymbol{\beta}}\bigr)\bigr| = \lambda w_l/2$ and*

$\beta_l = 0$ for some l, where c_l is defined in (20)*, include l in $\mathcal{A}$ and set $s_l = sign\big(c_l(\widehat{\boldsymbol{\beta}})\big)$. Then, $\widehat{\boldsymbol{\beta}}$ is a local minimizer of* (14) *with penalty $\xi := \lambda/\widehat{\sigma}^2(\lambda)$ if and only if $\mathcal{A} = \emptyset$ or*

$$\frac{\xi\lambda}{2n}\mathbf{w}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}} \le 1,$$

which is equivalent to λ being the smaller (or double) root of the quadratic equation

$$\lambda = \xi\widehat{\sigma}^2(\lambda) = \frac{\xi}{n}\Big(\mathbf{y}'\mathbf{y} - \mathbf{y}'\mathbf{X}_{\mathcal{A}}\big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{X}_{\mathcal{A}}'\mathbf{y}\Big) + \frac{\xi\lambda^2}{4n}\mathbf{w}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}}. \tag{16}$$

Proof. Note that the square brackets [] below are used to deal with the case where $\widehat{\boldsymbol{\beta}}$ is a breakpoint. Let $g(\boldsymbol{\beta})$ be the objective function of (14) and let $f_{\mathbf{u}}(h) = g\big(\widehat{\boldsymbol{\beta}} + h\mathbf{u}\big)$ where $\mathbf{u}$ is a arbitrary vector. Then, we have

$$f_{\mathbf{u}}'(h) = \frac{-2\mathbf{u}'\mathbf{X}'\big(\mathbf{y} - \mathbf{X}\big(\widehat{\boldsymbol{\beta}} + h\mathbf{u}\big)\big)}{\frac{1}{n}\big\|\mathbf{y} - \mathbf{X}\big(\widehat{\boldsymbol{\beta}} + h\mathbf{u}\big)\big\|^2} + \xi\mathbf{u}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}} + \xi\|\mathbf{u}_{\mathcal{A}^c}'\mathbf{w}_{\mathcal{A}^c}\|_1 - \big[\xi u_l s_l w_l - \xi|u_l|w_l\big].$$

By the definition of $\mathcal{A}$ and Proposition 5.1, we have $\big|c_k\big(\widehat{\boldsymbol{\beta}}\big)\big| = \lambda w_k/2$ for $k \in \mathcal{A}$ and $\big|c_k\big(\widehat{\boldsymbol{\beta}}\big)\big| < \lambda w_k/2$ for $k \in \mathcal{A}^c$. Therefore, we have

$$f_{\mathbf{u}}'(0) = \frac{-2\mathbf{u}_{\mathcal{A}^c}'\mathbf{X}_{\mathcal{A}^c}'\big(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\big) + \lambda\|\mathbf{u}_{\mathcal{A}^c}'\mathbf{w}_{\mathcal{A}^c}\|_1}{\widehat{\sigma}^2(\lambda)} - \big[\xi u_l s_l w_l - \xi|u_l|w_l\big] > 0,$$

unless $\mathbf{u}_{\mathcal{A}^c} = \mathbf{0}_{\mathcal{A}^c}$ [and $u_l s_l \ge 0$]. Therefore, the claim follows when $\mathcal{A} = \emptyset$. Now we may assume the worst case $\mathbf{u}_{\mathcal{A}^c} = \mathbf{0}_{\mathcal{A}^c}$ [and $u_l s_l \ge 0$]. Then, the first derivative becomes, by Proposition 5.1,

$$f_{\mathbf{u}}'(h) = \frac{-\lambda\mathbf{u}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}} + 2h\mathbf{u}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\mathbf{u}_{\mathcal{A}}}{\frac{1}{n}\big\|\mathbf{y} - \mathbf{X}_{\mathcal{A}}\big(\widehat{\boldsymbol{\beta}}_{\mathcal{A}} + h\mathbf{u}_{\mathcal{A}}\big)\big\|^2} + \xi\mathbf{u}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}}.$$

Since $f_{\mathbf{u}}'(0) = 0$, we must have

$$f_{\mathbf{u}}''(0+) = \frac{2\mathbf{u}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\mathbf{u}_{\mathcal{A}}\big\|\mathbf{y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}\big\|^2 - \lambda^2\mathbf{u}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}\mathbf{u}_{\mathcal{A}}}{\frac{1}{n}\big\|\mathbf{y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}\big\|^4} \ge 0,$$

for any $\mathbf{u}_{\mathcal{A}}$ [with $u_l s_l \ge 0$], which is true if and only if $2n\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}} - \xi\lambda\mathbf{S}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}}\mathbf{w}_{\mathcal{A}}'\mathbf{S}_{\mathcal{A}}$ is positive semi-definite, whether $\widehat{\boldsymbol{\beta}}$ is a breakpoint or not. □

Proposition 2.1 shows which solution of (15) can be a solution of (14), and therefore we can track down the solution path for the L_1-penalized normal loglikelihood method if the lasso solution path is available. Figure 1(b) shows a simple possible curve for $\sigma_j^2(\lambda_j)$ and the lines λ_j/ξ for some ξ (shown in dashed lines). From the plot, one sees that not every lasso solution is a legitimate solution for (14) and some ξ has multiple local minima.

Unlike the equi-angular method, it is not clear how to find a particular solution of (14) when the lasso solution path is not available. Because $\lambda_j/\sigma_j^2(\lambda_j)$ is not monotone, it is impossible to determine whether a solution exists below or above the current λ_j. If the penalty ξ is small like ξ_1 in Figure 1(b), we can use the same strategy as that of the equi-angular method. If $\lambda_j/\widehat{\sigma}_j^2(\lambda_j) < \xi$, we increase λ_j, and conversely if $\lambda_j/\widehat{\sigma}_j^2(\lambda_j) > \xi$, we decrease λ_j. If the inequality changes its direction, we have the correct $\mathcal{A}_j$ and hence the exact λ_j that is the smaller (or double) root of $\lambda_j = \xi\widehat{\sigma}_j^2(\lambda_j)$. We used this modified algorithm in the simulation study below and were able to cut down substantially the computation time compared with the iterative quadratic approximation algorithm.

3. Simulation

In this section, we investigate the performance of the proposed methods for various kinds of covariance matrix via simulation. We also compare them with three existing methods; the L_1-penalized normal loglikelihood method of Huang et al. [10], the banding method of Bickel and Levina [1], and the sample covariance matrix. All the methods except the sample covariance matrix were fitted using the proposed DWL algorithm. They were implemented in the R software. We applied all the methods to the following six $m \times m$ covariance matrices.

1. $T\Sigma_1 T' = D$ with $\phi_{j,j-1} = 0.8$ and $\phi_{jk} = 0$ for $k < j-1$ and $2 \le j \le m$, and $\sigma_j^2 = 1$ for $1 \le j \le m$.
2. Σ_2 is same as Σ_1 except $\sigma_j^2 = 16$ for odd j.
3. $T\Sigma_3 T' = D$ with $\phi_{jk} = 0.5^{j-k}$ for $k < j$ and $2 \le j \le m$, and $\sigma_j^2 = 1$ for $1 \le j \le m$.
4. Σ_4 is same as Σ_3 except $\sigma_j^2 = 16$ for odd j.
5. Σ_5 was chosen from a sample covariance matrix of m stock returns during a certain period of time, but we deliberately set $\phi_{jk} = 0$ if $|\phi_{jk}| < 0.01$ to create sparsity.
6. Σ_6 is same as Σ_5 except that σ_j^2 is multiplied by 16 for odd j.

Note that the modified Cholesky factors for Σ_1 through Σ_6 are all parsimonious. While Σ_1 through Σ_4 are of trivial form, Σ_5 and Σ_6 were chosen to mimic the practical situations in finance and economics. The even-numbered covariance matrices were modified from the odd-numbered ones to highlight the difference between Huang et al.'s method and the equi-angular method.

We used the entropy loss(Δ_1) and the Kullback-Leibler loss(Δ_2) to measure the accuracy of a covariance matrix estimate, which are defined as follows:

$$\Delta_1(\Sigma, \widehat{\Sigma}) = \mathrm{tr}(\Sigma^{-1}\widehat{\Sigma}) - \log|\Sigma^{-1}\widehat{\Sigma}| - m,$$
$$\Delta_2(\Sigma, \widehat{\Sigma}) = \mathrm{tr}(\widehat{\Sigma}^{-1}\Sigma) - \log|\widehat{\Sigma}^{-1}\Sigma| - m,$$

where Σ is the true covariance matrix and $\widehat{\Sigma}$ is the estimate. The entropy loss was used in Huang et al. [10] and is more appropriate for the covariance matrix, while the Kullback-Leibler loss was used in Levina et al. [12] and is more appropriate for the inverse covariance matrix. We also considered two quadratic loss functions Δ_3 and Δ_4:

$$\Delta_3(\Sigma, \widehat{\Sigma}) = \mathrm{tr}\big(\Sigma^{-1}\widehat{\Sigma} - \mathbf{I}\big)^2, \quad \Delta_4(\Sigma, \widehat{\Sigma}) = \mathrm{tr}\big(\widehat{\Sigma}^{-1}\Sigma - \mathbf{I}\big)^2.$$

Table 1: The averages and standard errors in parenthesis of the entropy losses (Δ_1) and the Kullback-Liebler losses (Δ_2) for normally distributed data. The variables remained in their original ordering. The tuning parameters were chosen by validation of another 100 observations from the same distribution. The sample size n is 100. The number of simulation runs is 200.

m	Σ	Δ	Sample	Bickel et al.	Huang et al.	$\widehat{\Sigma}_s$	$\widehat{\Sigma}_a$
30	Σ_1	Δ_1	5.271 (0.025)	0.590 (0.008)	1.200 (0.012)	1.100 (0.011)	1.189 (0.013)
		Δ_2	8.419 (0.069)	0.622 (0.009)	1.482 (0.019)	1.354 (0.016)	1.473 (0.018)
	Σ_2	Δ_1	5.287 (0.024)	0.601 (0.007)	1.649 (0.015)	1.562 (0.013)	1.171 (0.012)
		Δ_2	8.420 (0.064)	0.635 (0.008)	1.989 (0.023)	1.869 (0.021)	1.409 (0.019)
	Σ_3	Δ_1	5.231 (0.024)	1.439 (0.016)	1.791 (0.014)	1.714 (0.013)	1.768 (0.014)
		Δ_2	8.308 (0.068)	1.560 (0.017)	2.285 (0.026)	2.148 (0.023)	2.238 (0.024)
	Σ_4	Δ_1	5.317 (0.025)	1.591 (0.017)	2.087 (0.016)	2.019 (0.015)	1.755 (0.012)
		Δ_2	8.517 (0.070)	1.696 (0.016)	2.621 (0.026)	2.438 (0.025)	2.153 (0.021)
	Σ_5	Δ_1	5.268 (0.024)	4.656 (0.044)	2.325 (0.014)	2.025 (0.012)	2.060 (0.013)
		Δ_2	8.402 (0.068)	4.413 (0.029)	2.965 (0.025)	2.535 (0.023)	2.590 (0.023)
	Σ_6	Δ_1	5.264 (0.025)	9.514 (0.134)	2.799 (0.018)	3.095 (0.019)	2.325 (0.014)
		Δ_2	8.367 (0.072)	6.423 (0.030)	3.505 (0.031)	3.714 (0.029)	2.853 (0.025)
80	Σ_1	Δ_1	49.578 (0.095)	1.619 (0.014)	3.985 (0.024)	3.477 (0.021)	3.712 (0.021)
		Δ_2	311.814 (2.685)	1.711 (0.016)	5.366 (0.039)	4.673 (0.037)	4.994 (0.036)
	Σ_2	Δ_1	49.443 (0.091)	1.603 (0.014)	7.691 (0.049)	6.661 (0.032)	3.728 (0.022)
		Δ_2	311.694 (2.570)	1.686 (0.016)	10.103 (0.078)	8.834 (0.060)	4.788 (0.035)
	Σ_3	Δ_1	49.667 (0.086)	4.000 (0.018)	5.293 (0.025)	5.068 (0.021)	5.189 (0.023)
		Δ_2	317.154 (2.730)	4.291 (0.024)	7.618 (0.053)	7.187 (0.048)	7.521 (0.052)
	Σ_4	Δ_1	49.504 (0.099)	4.302 (0.034)	6.969 (0.035)	6.524 (0.026)	5.319 (0.024)
		Δ_2	311.481 (2.911)	4.699 (0.029)	10.239 (0.079)	8.865 (0.057)	7.385 (0.048)
	Σ_5	Δ_1	49.602 (0.087)	26.010 (0.153)	8.328 (0.031)	7.528 (0.023)	7.423 (0.024)
		Δ_2	311.630 (2.541)	19.318 (0.051)	12.164 (0.069)	10.837 (0.060)	10.746 (0.053)
	Σ_6	Δ_1	49.522 (0.096)	76.428 (0.580)	16.047 (0.094)	16.294 (0.053)	10.757 (0.034)
		Δ_2	311.909 (2.751)	30.710 (0.066)	20.763 (0.086)	18.396 (0.085)	13.830 (0.065)

We generated normally distributed data with mean $\mathbf{0}$ and covariance matrix Σ, and then estimated $\widehat{\Sigma}$ using the data by all methods considered. The sample size was 100 and we repeated this process 200 times. To compare the performance of the estimators, we calculated the average losses (risk) and the corresponding standard errors. Since the modified Cholesky decomposition is not permutation invariant, we repeated the whole process with a random permutation of the variables before estimation to study the sensitivity to

Table 2: The averaged percentages and the corresponding standard errors in parenthesis of Type-I and Type-II errors in identifying zero coefficients in the Cholesky factor T. Type-I error represents falsely identifying a zero as a nonzero. Type-II error means falsely identifying a nonzero as a zero. The figures are from the same simulation of Table 1.

m	Σ	Type	Bickel et al.	Huang et al.	$\widehat{\Sigma}_s$	$\widehat{\Sigma}_a$
30	Σ_1	I	0.00% (0.00%)	14.59% (0.19%)	13.91% (0.21%)	14.48% (0.21%)
		II	0.00% (0.00%)	0.00% (0.00%)	0.00% (0.00%)	0.00% (0.00%)
	Σ_2	I	0.03% (0.03%)	21.79% (0.19%)	21.68% (0.19%)	13.73% (0.19%)
		II	0.00% (0.00%)	1.11% (0.09%)	0.69% (0.08%)	0.75% (0.08%)
	Σ_3	I	-% (-%)	-% (-%)	-% (-%)	-% (-%)
		II	74.52% (0.19%)	59.21% (0.17%)	59.72% (0.17%)	59.34% (0.16%)
	Σ_4	I	-% (-%)	-% (-%)	-% (-%)	-% (-%)
		II	72.50% (0.31%)	56.39% (0.19%)	56.64% (0.17%)	60.17% (0.16%)
	Σ_5	I	31.76% (0.27%)	28.57% (0.40%)	22.80% (0.37%)	22.18% (0.38%)
		II	44.61% (0.33%)	49.22% (0.24%)	49.52% (0.21%)	50.39% (0.22%)
	Σ_6	I	40.35% (0.55%)	34.90% (0.47%)	33.18% (0.41%)	23.14% (0.37%)
		II	35.11% (0.56%)	46.56% (0.29%)	38.94% (0.25%)	46.87% (0.20%)
80	Σ_1	I	0.00% (0.00%)	6.44% (0.05%)	6.72% (0.05%)	6.45% (0.05%)
		II	0.00% (0.00%)	0.00% (0.00%)	0.00% (0.00%)	0.00% (0.00%)
	Σ_2	I	0.00% (0.00%)	12.94% (0.05%)	12.34% (0.05%)	6.32% (0.05%)
		II	0.00% (0.00%)	1.30% (0.06%)	0.74% (0.05%)	0.94% (0.05%)
	Σ_3	I	-% (-%)	-% (-%)	-% (-%)	-% (-%)
		II	90.07% (0.05%)	81.53% (0.05%)	81.50% (0.06%)	81.60% (0.05%)
	Σ_4	I	-% (-%)	-% (-%)	-% (-%)	-% (-%)
		II	88.52% (0.10%)	79.18% (0.05%)	79.37% (0.06%)	81.66% (0.05%)
	Σ_5	I	14.82% (0.17%)	12.52% (0.11%)	11.76% (0.09%)	11.70% (0.11%)
		II	77.46% (0.20%)	74.63% (0.10%)	72.29% (0.09%)	73.29% (0.08%)
	Σ_6	I	21.31% (0.13%)	12.48% (0.22%)	17.85% (0.09%)	12.37% (0.10%)
		II	68.84% (0.16%)	74.19% (0.22%)	66.49% (0.09%)	69.95% (0.09%)

permutation of each method.

For tuning the penalty parameters, we generated additional 100 validation data in each run, using the same normal distribution, and chose the penalty parameter that maximizes the likelihood of $\widehat{\Sigma}$ given the validation data. We also tried the K-fold cross-validated loglikelihood criterion by randomly dividing the data into K groups. For each group of data, its loglikelihood is calculated based on the covariance matrix estimated using the other $K-1$ groups only. Then the K-fold cross-validated loglikelihood criterion is defined as the sum of the K loglikelihoods. For instance, for normally distributed data, we have

$$\mathrm{CV}(\cdot) = \frac{1}{K}\sum_{k=1}^{K}\left(n_k \log\left|\widehat{\Sigma}_{-k}(\cdot)\right| + \sum_{i\in I_k}(\mathbf{y}^i)'\widehat{\Sigma}_{-k}^{-1}(\cdot)\mathbf{y}^i\right),$$

where I_k is the index set for the k-th group, n_k is the size of I_k, and $\widehat{\Sigma}_{-k}(\cdot)$ is the covariance matrix estimated excluding the k-th group of data. Then, we chose the penalizing parameter that minimizes CV. The two methods ($K=5$ was used) gave similar results, and thus we only report the results from the method using 100 validation data.

Table 1 summarizes the simulation results for Δ_1 and Δ_2. The results for Δ_3 and Δ_4 are omitted because they are similar to those reported in Table 1. To compare the ability of each method in capturing the sparsity, we report the percentages of unidentified true zeros in the Cholesky factor T (type-I error) and the percentages of the nonzeros in T mistakenly identified as zero (type-II error) in Table 2. Based on the simulation study, the loss functions provide similar results for most covariance matrices considered. But in some cases, the loss functions disagreed (e.g., Σ_5 and Σ_6 with the banding method), and in such cases the procedure for tuning the penalty parameter acted more friendly toward the loss functions for the inverse covariance matrix (Δ_2 and Δ_4). This indicates that it would make little sense to judge the methods based on Δ_1 or Δ_3. The conclusions that follow are well-supported by all loss functions.

For Σ_1 and Σ_2, whose Cholesky factors are banded, the banding method was the best. Although the Cholesky factors of Σ_3 and Σ_4 are not banded, since their entries decay exponentially from the diagonal, the banding method was still the best among all the methods considered. Σ_5 and Σ_6, however, showed the drawbacks of the banding method. It worked poorly for the non-banded covariance matrices, suggesting that the use of the banding method to general covariance matrices might be inappropriate.

Contrary to the banding method, the method of Huang et al. and the two proposed methods work reasonably well for all covariance matrices with the equi-angular method outperforming the others in general. Furthermore, the equi-angular method also has the smallest difference between odd-numbered and even-numbered covariance matrices. In particular, there was almost no difference for the first two pairs of the covariance matrices, indicating that the method indeed stably penalized the coefficients. It should also be pointed out that, except for Σ_5 with $m = 80$, the equi-sparse method slightly outperforms the equi-angular method for the odd-numbered covariance matrices. This is understandable because the associated regressions look similar.

The method of Huang et al. and the equi-angular method are supposed to be the same for Σ_1 and Σ_3, but minor difference exists because of the uncertainty in the realized residual variances. As shown in Table 2, the type-I and type-II errors for Σ_1 and Σ_3 are also similar for the two estimation methods. However, the performance of the method of Huang et al. deteriorates for Σ_2 and Σ_4 whereas that of the equi-angular

Table 3: The averages and standard errors in parenthesis of the entropy losses (Δ_1) and the Kullback-Liebler losses (Δ_2) for normally distributed data. Every setting is same as that for Table 1 except that the variables were randomly permuted before estimation.

m	Σ	Δ	Sample	Bickel et al.	Huang et al.	$\widehat{\Sigma}_s$	$\widehat{\Sigma}_a$
30	Σ_1	Δ_1	5.258 (0.025)	5.974 (0.104)	1.755 (0.016)	1.564 (0.014)	1.559 (0.016)
		Δ_2	8.388 (0.067)	7.465 (0.063)	2.065 (0.022)	1.838 (0.021)	1.853 (0.024)
	Σ_2	Δ_1	5.296 (0.023)	5.483 (0.186)	2.100 (0.019)	1.736 (0.014)	1.444 (0.013)
		Δ_2	8.457 (0.061)	7.563 (0.067)	2.453 (0.026)	2.024 (0.021)	1.686 (0.019)
	Σ_3	Δ_1	5.289 (0.025)	5.879 (0.099)	2.150 (0.015)	2.097 (0.014)	2.076 (0.015)
		Δ_2	8.433 (0.070)	6.139 (0.045)	2.608 (0.023)	2.523 (0.024)	2.524 (0.025)
	Σ_4	Δ_1	5.293 (0.023)	5.581 (0.103)	3.280 (0.029)	2.984 (0.021)	2.879 (0.023)
		Δ_2	8.422 (0.060)	7.743 (0.061)	3.815 (0.033)	3.503 (0.029)	3.402 (0.028)
	Σ_5	Δ_1	5.236 (0.023)	5.704 (0.090)	2.383 (0.015)	2.107 (0.014)	2.132 (0.014)
		Δ_2	8.300 (0.061)	5.094 (0.037)	3.047 (0.026)	2.640 (0.023)	2.709 (0.024)
	Σ_6	Δ_1	5.289 (0.024)	6.339 (0.164)	3.810 (0.028)	3.479 (0.026)	3.116 (0.024)
		Δ_2	8.481 (0.068)	7.160 (0.060)	4.330 (0.036)	3.891 (0.028)	3.501 (0.028)
80	Σ_1	Δ_1	49.483 (0.096)	97.795 (1.498)	6.445 (0.036)	5.520 (0.029)	5.185 (0.027)
		Δ_2	308.775 (2.699)	50.495 (0.178)	7.857 (0.054)	6.888 (0.049)	6.479 (0.041)
	Σ_2	Δ_1	49.451 (0.087)	345.259 (8.256)	10.377 (0.064)	7.755 (0.034)	4.925 (0.024)
		Δ_2	309.777 (2.378)	79.296 (0.453)	12.419 (0.079)	9.704 (0.061)	5.995 (0.040)
	Σ_3	Δ_1	49.524 (0.091)	67.203 (1.309)	7.304 (0.036)	7.161 (0.029)	6.797 (0.029)
		Δ_2	312.798 (2.541)	36.046 (0.129)	9.228 (0.053)	9.063 (0.054)	8.723 (0.054)
	Σ_4	Δ_1	49.349 (0.092)	179.905 (4.293)	19.575 (0.132)	16.075 (0.118)	14.669 (0.121)
		Δ_2	308.977 (2.681)	65.420 (0.292)	17.917 (0.074)	15.487 (0.071)	15.260 (0.072)
	Σ_5	Δ_1	49.275 (0.096)	33.352 (0.228)	8.719 (0.030)	7.711 (0.021)	7.518 (0.023)
		Δ_2	306.556 (2.552)	21.736 (0.059)	12.990 (0.075)	10.959 (0.052)	10.841 (0.055)
	Σ_6	Δ_1	49.416 (0.099)	102.046 (1.664)	25.821 (0.224)	17.857 (0.070)	15.287 (0.062)
		Δ_2	312.233 (2.870)	39.589 (0.143)	23.962 (0.130)	17.987 (0.068)	15.725 (0.056)

method remains the same. From Table 2, the under-performance of the method of Huang et al. is caused by its inability to shrink the regression coefficients under imbalanced penalties. Since Σ_5 has heterogeneous residual variances, the equi-angular method fares better, and the difference was amplified for Σ_6. The equi-angular method has smaller type-I and type-II errors for Σ_5 and Σ_6 than the Huang et al.'s method. The only exception occurs in the type-II errors when $m = 30$.

As shown in Table 3, re-ordering the variables introduces some difficulties for each estimation method, except the sample covariance matrix. But the equi-angular method remains the best. As expected, the banding method encounters the biggest trouble. The advantage of the equi-sparse method for Σ_1 and Σ_3 disappeared because the associated regressions no longer look similar, but the method still fares better than the equi-angular method for Σ_5 with $m = 30$. Overall, the two proposed methods are less affected by the permutation of the variables.

4. Empirical Study

We apply the proposed methods to daily stock returns. Returns of 80 stocks from January 2, 1990 to December 31, 2007 were collected, but the effective sample period is from January 2, 1993 and December 31, 2007. The data of the first three years were used only for estimation and parameter tuning purposes. At the beginning of each month starting from January 1993, the covariance matrix was estimated using the past $N \in \{6, 12, 18, 24\}$ months of daily returns. Then we chose the global minimum variance (GMV) portfolio based on the estimated covariance matrix. The portfolio was held for a month, and at the end of the month the average daily return and the risk (standard deviation) of the portfolio were recorded. The Sharpe ratio, which is the average daily return divided by the risk, was also recorded monthly. We collected the evaluation statistics for 180 months until December 2007.

The tuning parameter was chosen to be the one that did best for the previous four months in a cross validation manner. Specifically, to choose the tuning parameter for $\widehat{\Sigma}_i(\cdot)$ at the beginning of the i-th month, we used daily returns of the past $N + 4$ months, i.e., from the $(i - N - 4)$-th month to the $(i - 1)$-th month. The period contains five consecutive N-month rolling windows, from the $(i - 4)$-th to the i-th, so let $\{\widehat{\Sigma}_j(\cdot)\}_{i-4 \le j \le i}$ be the estimated covariance matrices from the windows. Then, for each j, we validate the daily returns of 4 months outside the j-th window by the normal loglikelihood for $\widehat{\Sigma}_j(\cdot)$. That is, we chose the tuning parameter to be the maximizer of

$$\sum_{j=i-4}^{i} llk\big(\widehat{\Sigma}_j(\cdot)|\mathbf{r}_{i,-j}\big),$$

where $\mathbf{r}_{i,-j}$ denotes the daily returns from the $(i - N - 4)$-th month to the $(i - 1)$-th month, excluding the N-month portion that was used to estimate $\widehat{\Sigma}_j(\cdot)$.

Table 4 reports the averages of the recorded monthly statistics. Except for $N = 6$, the equi-angular method has the smallest mean risks. To gain some insight into the difference in risks, we calculated the means and standard errors of the paired differences of the risk of other methods against the equi-angular method. Although the differences might not be serially independent because of the overlapping windows, most of the mean differences are higher than its two standard-error limits. The method of Huang et al. has the highest Sharpe ratio, but the differences in Sharpe ratio are small for the equi-angular method. We

Table 4: Results of the stock return analysis.

	Sample	Bickel et al.	Huang et al.	$\widehat{\Sigma}_s$	$\widehat{\Sigma}_a$
Estimated using the past 6 months' daily return data					
Risk	0.9643%	0.6651%	0.6912%	0.6819%	0.6700%
ΔRisk	0.2944%	-0.0049%	0.0213%	0.0119%	-%
SE(ΔRisk)	(0.0178%)	(0.0054%)	(0.0027%)	(0.0019%)	(-%)
Mean Return	0.0394%	0.0477%	0.0538%	0.0495%	0.0501%
Sharpe Ratio	0.061	0.102	0.113	0.104	0.108
Gross Exp.	4.709	1.724	1.409	1.432	1.433
Estimated using the past 12 months' daily return data					
Risk	0.7336%	0.6662%	0.6682%	0.6662%	0.6564%
ΔRisk	0.0772%	0.0098%	0.0118%	0.0098%	-%
SE(ΔRisk)	(0.0088%)	(0.0061%)	(0.0020%)	(0.0020%)	(-%)
Mean Return	0.0465%	0.0472%	0.0539%	0.0486%	0.0504%
Sharpe Ratio	0.087	0.096	0.113	0.103	0.108
Gross Exp.	3.061	2.046	1.585	1.610	1.586
Estimated using the past 18 months' daily return data					
Risk	0.7047%	0.6774%	0.6706%	0.6688%	0.6611%
ΔRisk	0.0436%	0.0163%	0.0095%	0.0076%	-%
SE(ΔRisk)	(0.0072%)	(0.0062%)	(0.0019%)	(0.0018%)	(-%)
Mean Return	0.0493%	0.0531%	0.0549%	0.0505%	0.0514%
Sharpe Ratio	0.092	0.101	0.114	0.105	0.108
Gross Exp.	2.666	2.165	1.662	1.686	1.666
Estimated using the past 24 months' daily return data					
Risk	0.7018%	0.6877%	0.6737%	0.6767%	0.6685%
ΔRisk	0.0333%	0.0192%	0.0052%	0.0082%	-%
SE(ΔRisk)	(0.0063%)	(0.0060%)	(0.0016%)	(0.0017%)	(-%)
Mean Return	0.0501%	0.0529%	0.0551%	0.0507%	0.0515%
Sharpe Ratio	0.093	0.099	0.109	0.100	0.104
Gross Exp.	2.481	2.176	1.716	1.742	1.712

also calculated the gross exposure, which is the L_1-norm of the weight vector of the portfolio and closely related to the performance of the portfolio. The method of Huang et al. and the two proposed methods have similar gross exposure measures. The poor performance of the sample covariance matrices is clearly seen in the study. If the covariance matrix is poorly estimated, the uncertainty in portfolio weights increases that in turn lead to poor performance. Overall, in the empirical study, the method of Huang et al. and the two proposed methods work fairly well.

5. Algorithms for the Weighted Lasso

5.1. Characterization of the Weighted Lasso Solution

In this subsection, we characterize the solution of the weighted lasso regression that allows weighted penalties for different coefficients. This will become the basis for the proposed algorithm in the next subsection. Let $\widehat{\boldsymbol{\beta}}$ be the solution of the following weighted lasso problem

$$\text{minimize } \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{k=1}^{m} \lambda_k |\beta_k|, \tag{17}$$

where $\mathbf{y}$ is an $n \times 1$ vector, $\mathbf{X}$ is an $n \times m$ design matrix, $\boldsymbol{\beta}$ is an $m \times 1$ vector, and $\lambda_k \geq 0$ are the penalties. Let $\mathcal{A} = \{k : \widehat{\beta}_k \neq 0\}$ be the index set of the nonzero coefficients in $\widehat{\boldsymbol{\beta}}$ and $\mathbf{S} = \operatorname{diag}(s_1, \ldots, s_m)$ the sign matrix for $\widehat{\boldsymbol{\beta}}$, where $s_k = \operatorname{sign}(\widehat{\beta}_k)$. Denote the penalty vector by $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)'$, and let $\widehat{\boldsymbol{\beta}}_{\mathcal{A}} = (\ldots, \widehat{\beta}_k, \ldots)'_{k \in \mathcal{A}}$ and $\mathbf{X}_{\mathcal{A}} = [\ldots, \mathbf{x}_k, \ldots]_{k \in \mathcal{A}}$, where $\mathbf{x}_k$ is the k-th column of $\mathbf{X}$. Here we assume $n \geq m$, so that $\mathbf{X}'\mathbf{X}$ is positive definite. This makes the objective function of (17) strictly convex.

Since the objective function is differentiable with respect to $\boldsymbol{\beta}_{\mathcal{A}}$ at $\widehat{\boldsymbol{\beta}}$, we have

$$-\mathbf{X}'_{\mathcal{A}}(\mathbf{y} - \mathbf{X}_{\mathcal{A}}\widehat{\boldsymbol{\beta}}_{\mathcal{A}}) + \mathbf{S}_{\mathcal{A}}\boldsymbol{\gamma}_{\mathcal{A}} = \mathbf{0}_{\mathcal{A}}, \tag{18}$$

where $\boldsymbol{\gamma} = \frac{1}{2}\boldsymbol{\lambda}$, and obtain the closed form solution

$$\widehat{\boldsymbol{\beta}}_{\mathcal{A}} = (\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}})^{-1}\mathbf{X}'_{\mathcal{A}}\mathbf{y} - (\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}})^{-1}\mathbf{S}_{\mathcal{A}}\boldsymbol{\gamma}_{\mathcal{A}}. \tag{19}$$

For each k, define

$$c_k(\boldsymbol{\beta}) := \mathbf{x}'_k(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = -\frac{1}{2}\frac{d\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2}{d\beta_k}, \tag{20}$$

which has two interpretations. It is the derivative of the objective function with respect to β_k and can be seen as the covariance between $\mathbf{x}_k$ and the residual $\mathbf{y} - \mathbf{X}\boldsymbol{\beta}$. Then (18) implies

$$c_k(\widehat{\boldsymbol{\beta}}) = s_k\gamma_k, \quad k \in \mathcal{A}. \tag{21}$$

Also, by the optimality of $\widehat{\boldsymbol{\beta}}$, the rate at which the first term of (17) decreases must be smaller than or equal to the rate at which the second term increases for any changes of $\boldsymbol{\beta}$. Therefore, we have

$$|c_k(\widehat{\boldsymbol{\beta}})| \leq \gamma_k, \quad k \notin \mathcal{A}. \tag{22}$$

Note that the conditions (21) and (22) are closely related to the so-called Karush-Kuhn-Tucker (KKT) conditions for constrained optimization problems. And any vector $\widehat{\boldsymbol{\beta}}$ satisfying the two conditions is locally optimal. Since the objective function is strictly convex, the vector is also globally optimal. The following proposition characterizes the solution of the weighted lasso problem.

Proposition 5.1. *For any given $\widehat{\boldsymbol{\beta}}$, let $\mathcal{A}$ and $\mathbf{S}$ be as defined above. Then, $\widehat{\boldsymbol{\beta}}$ is the solution of* (17) *if and only if the conditions* (21) *and* (22) *are satisfied with $\gamma_k = \frac{1}{2}\lambda_k$, in which case, $\widehat{\boldsymbol{\beta}}_{\mathcal{A}}$ is given by* (19).

5.2. DWL algorithm

We now propose a new algorithm for the weighted lasso problem (17). The algorithm starts with the initial $\boldsymbol{\beta}^0$ which is the solution for a different penalty vector $\boldsymbol{\lambda}^0$. Our goal is to change $\boldsymbol{\lambda}^0$ and $\boldsymbol{\beta}^0$ in some way so that we can end up with the solution for the penalty vector $\boldsymbol{\lambda}$. The following proposition simply rephrases Proposition 5.1 and provides the condition on the initial $\boldsymbol{\beta}^0$.

Proposition 5.2. *For any given $\boldsymbol{\beta}^0$, let $\mathcal{A}$ and $\mathbf{S}$ be as defined above. Then, $\boldsymbol{\beta}^0$ is the solution of* (17) *with $\lambda_k = 2\gamma_k^0$, where*

$$\gamma_k^0 = \big|c_k(\boldsymbol{\beta}^0)\big|, \quad k \in \mathcal{A} \tag{23}$$

and

$$\gamma_k^0 \geq \big|c_k(\boldsymbol{\beta}^0)\big|, \quad k \notin \mathcal{A}, \tag{24}$$

if and only if

$$s_k = sign\big(c_k(\boldsymbol{\beta}^0)\big), \quad k \in \mathcal{A}. \tag{25}$$

When the condition holds, the solution $\boldsymbol{\beta}_{\mathcal{A}}^0$ can be rewritten as

$$\boldsymbol{\beta}_{\mathcal{A}}^0 = \big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{X}_{\mathcal{A}}'\mathbf{y} - \big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{S}_{\mathcal{A}}\boldsymbol{\gamma}_{\mathcal{A}}^0.$$

Following Proposition 5.2, we can begin with any initial vector $\boldsymbol{\beta}^0$ which satisfies the sign condition (25). Note that the class of the possible initial values is huge and the trivial ones include $\big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{X}_{\mathcal{A}}'\mathbf{y}$ for any index subset $\mathcal{A}$. They also include the zero vector $\mathbf{0}$ and the OLS solution $(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$, which are the only possible starting points of the LARS-lasso algorithm.

Suppose $\boldsymbol{\gamma}^0$ is a vector that satisfies the conditions (23) and (24). Then, by Proposition 5.2, $\boldsymbol{\beta}^0$ is the solution for the penalty vector $2\boldsymbol{\gamma}^0$. The strategy of the proposed algorithm is to change $\boldsymbol{\gamma}$ from $\boldsymbol{\gamma}^0$ to $\frac{1}{2}\boldsymbol{\lambda}$ linearly and to keep $\boldsymbol{\beta}$, which is initially $\boldsymbol{\beta}^0$, as the solution for the penalty vector $2\boldsymbol{\gamma}$. Note that there is some freedom in choosing $\boldsymbol{\gamma}^0$ because of the inequality in (24), and we suggest the trajectory for $\boldsymbol{\gamma}$ as follows.

$$\gamma_k(\alpha) = (1-\alpha)\big|c_k(\boldsymbol{\beta}^0)\big| + \frac{\alpha}{2}\lambda_k, \quad k \in \mathcal{A}, \tag{26}$$

$$\gamma_k(\alpha) = (1-\alpha)\Gamma + \frac{\alpha}{2}\lambda_k, \quad k \notin \mathcal{A}, \tag{27}$$

for $0 \leq \alpha \leq 1$, where Γ is a constant which is strictly greater than $\max\limits_{1\leq k\leq m}\big|c_k(\boldsymbol{\beta}^0)\big|$.

Let $\boldsymbol{\beta}(\alpha)$ be the solution for the penalty vector $2\boldsymbol{\gamma}(\alpha)$, and suppose that α increases from 0 continuously. Then, $\boldsymbol{\gamma}(\alpha)$ changes according to (26) and (27). And $\boldsymbol{\beta}_{\mathcal{A}}(\alpha)$ follows, by (19),

$$\boldsymbol{\beta}_{\mathcal{A}}(\alpha) = \boldsymbol{\beta}^0 - \alpha\big(\mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}\big)^{-1}\mathbf{S}_{\mathcal{A}}\frac{d\boldsymbol{\gamma}_{\mathcal{A}}(\alpha)}{d\alpha}, \tag{28}$$

and $c_k(\boldsymbol{\beta}(\alpha))$ evolves according to

$$\begin{aligned} c_k(\boldsymbol{\beta}(\alpha)) &= s_k\gamma_k(\alpha), \quad k \in \mathcal{A}, \\ &= c_k\big(\boldsymbol{\beta}^0\big) - \alpha \mathbf{x}_k' \mathbf{X}_{\mathcal{A}} \frac{d\boldsymbol{\beta}_{\mathcal{A}}(\alpha)}{d\alpha}, \quad k \notin \mathcal{A}. \end{aligned} \tag{29}$$

so long as the following three conditions hold:

$$\gamma_k(\alpha) = |c_k(\boldsymbol{\beta}(\alpha))|, \quad k \in \mathcal{A}, \tag{30}$$

$$\gamma_k(\alpha) \geq |c_k(\boldsymbol{\beta}(\alpha))|, \quad k \notin \mathcal{A}, \tag{31}$$

$$s_k = \operatorname{sign}\big(c_k(\boldsymbol{\beta}(\alpha))\big), \quad k \in \mathcal{A}. \tag{32}$$

While (30) is guaranteed by (29), (31) and (32) are subject to break as α increases from 0. (31) breaks when $|c_k(\boldsymbol{\beta}(\alpha))|$ becomes larger than $\gamma_k(\alpha)$ for some $k \notin \mathcal{A}$. (32) breaks when $\beta_k(\alpha)$ changes its sign for some $k \in \mathcal{A}$. Once a condition breaks at $\alpha = \alpha^*$, (28) and (29) no longer hold for $\alpha > \alpha^*$. However, owing to the following two lemmas, we can adjust $\mathcal{A}$ and $\mathbf{S}$ properly so that (28) and (29) continue to hold. Lemma 5.1 deals with the condition (31) and Lemma 5.2 deals with the condition (32).

Lemma 5.1. *Assume that $|c_k(\boldsymbol{\beta})|$ and γ_k has just agreed at $\alpha = \alpha^*$ for some $k \notin \mathcal{A}$ in the course of α increasing from 0. Then, after inserting k into $\mathcal{A}$ with $s_k = sign\big(c_k(\boldsymbol{\beta}(\alpha^*))\big)$, it follows that*

$$s_k \frac{d\beta_k(\alpha)}{d\alpha}\bigg|_{\alpha^*+} > 0,$$

and hence $\boldsymbol{\beta}(\alpha^* + \delta)$ remains as the solution for the penalty vector $2\boldsymbol{\gamma}(\alpha^* + \delta)$ for small $\delta > 0$.

Proof. By assumption, we have $|c_k(\boldsymbol{\beta}(\alpha^*))| = \gamma(\alpha^*)$ and $s_k \frac{dc_k(\boldsymbol{\beta}(\alpha))}{d\alpha}\Big|_{\alpha^*-} > \frac{d\gamma_k(\alpha)}{d\alpha}$. Let $\mathcal{B} = \mathcal{A} \cup \{k\}$, and let $\mathbf{A} = \mathbf{X}_{\mathcal{A}}'\mathbf{X}_{\mathcal{A}}$, $\mathbf{b} = \mathbf{X}_{\mathcal{A}}'\mathbf{x}_k$, $c = \mathbf{x}_k'\mathbf{x}_k$, and $d = c - \mathbf{b}'\mathbf{A}^{-1}\mathbf{b}$. Note that, by (28) and (29),

$$s_k \frac{dc_k(\boldsymbol{\beta}(\alpha))}{d\alpha}\bigg|_{\alpha^*-} = s_k \mathbf{b}'\mathbf{A}^{-1}\mathbf{S}_{\mathcal{A}} \frac{d\boldsymbol{\gamma}_{\mathcal{A}}(\alpha)}{d\alpha} > \frac{d\gamma_k(\alpha)}{d\alpha}.$$

Since $d > 0$, this implies

$$\begin{aligned} s_k \frac{d\beta_k(\alpha)}{d\alpha}\bigg|_{\alpha^*+} &= -s_k \mathbf{e}_k' \big(\mathbf{X}_{\mathcal{B}}'\mathbf{X}_{\mathcal{B}}\big)^{-1} \mathbf{S}_{\mathcal{B}} \frac{d\boldsymbol{\gamma}_{\mathcal{B}}(\alpha)}{d\alpha} \\ &= \frac{1}{d} s_k \mathbf{b}'\mathbf{A}^{-1}\mathbf{S}_{\mathcal{A}} \frac{d\boldsymbol{\gamma}_{\mathcal{A}}(\alpha)}{d\alpha} - \frac{1}{d}\frac{d\gamma_k(\alpha)}{d\alpha} > 0, \end{aligned}$$

where $\mathbf{e}_k$ is a unit vector of length $|\mathcal{B}|$ with 1 for the entry corresponding to k. Since $\beta_k(\alpha^*) = 0$, this implies $s_k\beta_k(\alpha^* + \delta) > 0$ and (32) is satisfied for k at $\alpha = \alpha^* + \delta$. □

Lemma 5.2. *Assume that β_k has just reached 0 at $\alpha = \alpha^*$ for some $k \in \mathcal{A}$ in the course of α increasing from 0. Then, after removing k from $\mathcal{A}$, it follows that*

$$s_k \frac{dc_k(\boldsymbol{\beta}(\alpha))}{d\alpha}\bigg|_{\alpha^*+} < \frac{d\gamma_k(\alpha)}{d\alpha},$$

and hence $\boldsymbol{\beta}(\alpha^* + \delta)$ remains as the solution for the penalty vector $2\boldsymbol{\gamma}(\alpha^* + \delta)$ for small $\delta > 0$.

Proof. By assumption, we have $\beta_k(\alpha^*) = 0$ and $s_k \frac{d\beta_k(\alpha)}{d\alpha}\Big|_{\alpha^*-} < 0$ where $s_k = sign\big(\beta_k(\alpha^*-)\big)$. Let $\mathcal{B} = \mathcal{A} - \{k\}$, and let $\mathbf{A} = \mathbf{X}'_{\mathcal{B}}\mathbf{X}_{\mathcal{B}}$, $\mathbf{b} = \mathbf{X}'_{\mathcal{B}}\mathbf{x}_k$, $c = \mathbf{x}'_k\mathbf{x}_k$, and $d = c - \mathbf{b}'\mathbf{A}^{-1}\mathbf{b}$. Simple matrix algebra shows

$$\begin{aligned} s_k \frac{d\beta_k(\alpha)}{d\alpha}\Big|_{\alpha^*-} &= -s_k \mathbf{e}'_k \big(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\big)^{-1} \mathbf{S}_{\mathcal{A}} \frac{d\boldsymbol{\gamma}_{\mathcal{A}}(\alpha)}{d\alpha} \\ &= \frac{1}{d} s_k \mathbf{b}'\mathbf{A}^{-1}\mathbf{S}_{\mathcal{B}} \frac{d\boldsymbol{\gamma}_{\mathcal{B}}(\alpha)}{d\alpha} - \frac{1}{d}\frac{d\gamma_k(\alpha)}{d\alpha} < 0, \end{aligned}$$

where $\mathbf{e}_k$ is a unit vector of length $|\mathcal{A}|$ with 1 for the entry corresponding to k. Since $d > 0$,

$$s_k \frac{dc_k(\boldsymbol{\beta}(\alpha))}{d\alpha}\Big|_{\alpha^*+} = s_k \mathbf{b}'\mathbf{A}^{-1}\mathbf{S}_{\mathcal{B}} \frac{d\boldsymbol{\gamma}_{\mathcal{B}}(\alpha)}{d\alpha} < \frac{d\gamma_k(\alpha)}{d\alpha}.$$

Since $|c_k(\boldsymbol{\beta}(\alpha^*))| = \gamma_k(\alpha^*)$, this implies $|c_k(\boldsymbol{\beta}(\alpha^* + \delta))| < \gamma_k(\alpha^* + \delta)$ and (31) is satisfied for k at $\alpha = \alpha^* + \delta$. □

In summary, as α goes from 0 to 1, if (31) breaks for some $k \notin \mathcal{A}$, we adjust the solution by inserting k into $\mathcal{A}$ with $s_k = sign\big(c_k(\boldsymbol{\beta}(\alpha))\big)$. If (32) breaks for some $k \in \mathcal{A}$, we adjust by removing k from $\mathcal{A}$. After the adjustment, we restart the whole process from the beginning regarding the current values as the initial values. By repeating the procedure until α can reach 1 with no violation of (31) or (32), we obtain the solution. Details of the proposed algorithm are given below.

1. For given $\boldsymbol{\beta}^0$, initialize $\mathcal{A}$ and $\mathbf{S}$, calculate $\boldsymbol{\gamma}^0$ and $c_k\big(\boldsymbol{\beta}^0\big)$, and find $\frac{d\boldsymbol{\gamma}(\alpha)}{d\alpha}$, $\frac{d\boldsymbol{\beta}(\alpha)}{d\alpha}$, and $\frac{dc_k(\boldsymbol{\beta}(\alpha))}{d\alpha}$.
2. Find the smallest $\alpha^* \geq 0$ at which either (31) or (32) breaks. If $\alpha^* \geq 1$, then set $\alpha^* = 1$.
3. Update $\boldsymbol{\gamma}^0 \leftarrow \boldsymbol{\gamma}(\alpha^*)$, $\boldsymbol{\beta}^0 \leftarrow \boldsymbol{\beta}(\alpha^*)$ and $c_k\big(\boldsymbol{\beta}^0\big) \leftarrow c_k(\boldsymbol{\beta}(\alpha^*))$.
4. If $\alpha^* = 1$, then stop.
5. Adjust $\mathcal{A}$ and $\mathbf{S}$ according to Lemma 5.1 or Lemma 5.2. Update $\frac{d\boldsymbol{\beta}(\alpha)}{d\alpha}$ and $\frac{dc_k(\boldsymbol{\beta}(\alpha))}{d\alpha}$. Go to step 2.

It is important to note that (28) and (29) exhibit the linearity of β_k's and c_k's in α. That makes it possible to find α^* in step 2 at $O(m)$ time complexity. Also note that updating $\big(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\big)^{-1}$ or its equivalent (for example, the ordinary Cholesky decomposition of $\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}$) when $\mathcal{A}$ is updated in step 5 can be done at $O(m^2)$ time complexity. Therefore, a single iteration of the algorithm costs $O(m^2)$ amount of computing time. The number of iterations needed to complete the algorithm depends on the number of nonzero coefficients in the solution. Actually, the number of iterations will be slightly but not much more than the number of nonzero coefficients because some coefficients may change their signs more than once during the algorithm. Overall, we can obtain the solution of the general lasso problem at the same level of time complexity needed to obtain the solution of the ordinary least squares problem.

The DWL algorithm can be seen as an extension of the LARS-lasso algorithm. Indeed, the DWL algorithm is identical to the LARS-lasso algorithm if $\lambda_k \equiv \lambda$. Note that the DWL algorithm also assumes the so-called one-at-a-time condition (see Efron et al. [8]). At every breakpoint of the condition (31) or

(32), only a single k must be involved. If two or more conditions break at the same time (same α), we have to decide the next direction with extra caution, or we can use the technique of jittering. We omit the performance analysis of the algorithm since it will be similar to the LARS-lasso algorithm, and we refer to Efron et al. [8].

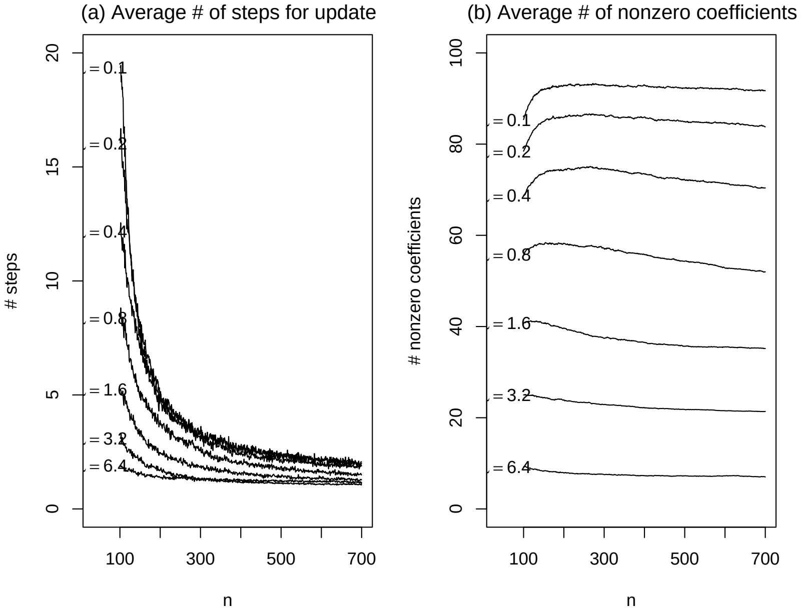
5.3. DWL-update Algorithm

It is true that we could use the LARS-lasso algorithm even when λ_k are heterogeneous, by changing the scale of the explanatory variables. But, note that the DWL algorithm can do something more. An advantage of the DWL algorithm is the ability to use the prior information about the nonzero index set $\mathcal{A}$ of the solution, if available. Recall that we can start the algorithm with initial $\boldsymbol{\beta}^0 = \left(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\right)^{-1}\mathbf{X}'_{\mathcal{A}}\mathbf{y}$ for any index subset $\mathcal{A}$. If there is no prior information about the solution, we would have to start with the zero vector $\mathbf{0}$ or the OLS solution $(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$. However, in some cases when we can reasonably guess the set $\mathcal{A}$ of the solution, if $\left(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\right)^{-1}$ or its equivalent is available at $O(m^2)$ time complexity, we can start with $\boldsymbol{\beta}^0 = \left(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\right)^{-1}\mathbf{X}'_{\mathcal{A}}\mathbf{y}$. If the guessed set $\mathcal{A}$ is the correct solution or close to it, we can obtain the solution by only a few iterations, regardless of the number of nonzero coefficients in the solution.

The situation where a reasonable guess for $\mathcal{A}$ is available occurs when we want to update the solution after an arrival of new data point. Suppose that we already have the solution of the general lasso problem for n data points. And suppose that the $(n + 1)$-th data point becomes available. This is the case we encounter often in time series analysis, machine learning literature, and many others. Note that the set $\mathcal{A}$ of the current solution is an excellent guess for that of the new solution because a single data point cannot change the solution substantially. Since we can update $\left(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\right)^{-1}$ or the ordinary Cholesky decomposition of $\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}$ at $O(m^2)$ amount of time, we can start the algorithm with $\boldsymbol{\beta}^0 = \left(\mathbf{X}'_{\mathcal{A}}\mathbf{X}_{\mathcal{A}}\right)^{-1}\mathbf{X}'_{\mathcal{A}}\mathbf{y}$ and get the new solution within a small number of iterations. If the sample size is moderately enough, we can update the solution at almost $O(m^2)$ time complexity.

To see how much time we can save from the updating algorithm, we simulated the model $y = \mathbf{x}'\boldsymbol{\beta} + e$ where $\mathbf{x} \sim \mathcal{N}_{100}(\mathbf{0}, \mathbf{I})$, $e \sim \mathcal{N}(0, 7^2)$, and $\boldsymbol{\beta} = (\underbrace{3, \ldots, 3}_{20}, \underbrace{1, \ldots, 1}_{20}, \underbrace{0.3, \ldots, 0.3}_{20}, \underbrace{0, \ldots, \ldots, 0}_{40})'$. Starting with the sample size $n = 100$, we fitted the lasso with penalty $\lambda_n = n\lambda$ for some fixed λ. We added new data points one by one and updated the lasso solution by the updating algorithm until $n = 700$. The number of algorithm steps needed to complete the update and the number of nonzero coefficients in the solution were recorded for every data point. This entire process was repeated 300 times and Figure 2 shows their means for several λ's. We did not run the LARS-lasso algorithm, but note that the number of nonzero coefficients is the minimum number of iterations in the LARS-lasso algorithm. So, the number of nonzero coefficients provides rough information concerning the total number of iterations needed. For smaller n around 100, the update cost is relatively high because the lasso solution is not stable. But we can see it is still more efficient than performing the LARS-lasso algorithm for every new data point. As the sample size increases, the lasso

Figure 2: The number of steps needed for updating and the number of nonzero coefficients in the lasso solution for each data point. These are averages from 300 times of simulations for various λ. The number of nonzero coefficients provides rough information concerning the number of iterations needed when the lasso solution is completely rebuilt.


(a) Average # of steps for update

(b) Average # of nonzero coefficients

solution stabilizes fairly quickly. After about $n = 200$, the average number of steps goes below 5 for every λ, that is, no matter how many nonzero coefficients exist in the solution.

5.4. Adding/Removing Variables

Another useful feature of the proposed DWL algorithm is the ease of adding and removing variables. Suppose $\widehat{\boldsymbol{\beta}}$ is the current solution and we want to add a new explanatory variable x_k into the lasso with penalty λ_k. If $\lambda_k \geq 2\left|c_k\left(\widehat{\boldsymbol{\beta}}\right)\right|$, then $\beta_k = 0$ is the solution with no change in other coefficients by proposition 5.1. If $\lambda_k < 2\left|c_k\left(\widehat{\boldsymbol{\beta}}\right)\right|$, we can apply the DWL algorithm with $\boldsymbol{\beta}_k^0 = 0$ and

$$\gamma_k(\alpha) = \alpha\left|c_k\left(\widehat{\boldsymbol{\beta}}\right)\right| + \frac{\alpha}{2}\lambda_k.$$

Removing a variable x_k from the lasso is also simple. If $\beta_k = 0$, nothing more than simply removing the variable directly, because it does not affect the conditions in Proposition 5.1. If $\beta_k \neq 0$, we can increase the corresponding penalty λ_k until β_k becomes 0, and then we can drop the variable. The number of iterations needed for adding or removing a variable will depend on how the variable of interest is correlated with others. But it will be usually much smaller than the number of iterations needed to rebuild the lasso solution except some extreme cases.

6. Discussion

In this paper, we proposed two new penalizing methods for estimating high-dimensional covariance matrix when the sample size is greater than the dimension. We provided a new point of view in fairly penalizing two or more regressions with a single penalizing parameter. The idea is not limited to the covariance matrix estimation problem, but can be applied to any kind of problem involving multiple penalized least squares problems. Although we focused on the L_1 penalty, it can naturally be extended to other kinds of penalties such as the L_2 penalty and the SCAD (Fan and Li [9]). Our limited simulation study and empirical data analysis show that the proposed methods and algorithm work well compared with other methods available in the literature.

Since the two proposed methods and that of Huang et al. are based on the lasso, they all have the same asymptotic properties. If we use some nonconvex penalty function, like the SCAD, they will have certain oracle property. We can also consider the adaptive lasso (Zou [23]). The lasso does not have the oracle property in general, but the adaptive lasso obtains the oracle property by giving weighted penalties for different coefficients. These issues deserve further study.

References

[1] P. J. Bickel and E. Levina (2008), Regularized Estimation of Large Covariance Matrices, *Annals of Statistics*, 36, 199-227

[2] P. J. Bickel and E. Levina (2008), Covariance Regularization by Thresholding, *Annals of Statistics*, 36, 2577-2604

[3] R. J. Boik (2002), Spectral Models for Covariance Matrices, *Biometrika*, 89, 159-182

[4] T. Y. M. Chiu, T. Leonard, and K. W. Tsui (1996), The Matrix-Logarithm Covariance Model, *Journal of the American Statistical Association*, 91, 198-210

[5] A. d'Aspremont, O. Banerjee, and L. El Ghaoui (2008), First-Order Methods for Sparse Covariance Selection, *SIAM Journal on Matrix Analysis and Applications*, 30, 56-66

[6] A. P. Dempster (1972), Covariance Selection, *Biometrics*, 28, 157-175

[7] P. J. Diggle and A. P. Verbyla (1998), Nonparametric Estimation of Covariance Structure in Longitudinal data, *Biometrics*, 54, 401-415

[8] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani (2004), Least Angle Regression, *Annals of Statistics*, 32, 407-499

[9] J. Fan and R. Li (2001), Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties, *Journal of the American Statistical Association*, 96, 1348-1360

[10] J. Z. Huang, N. Liu, M. Pourahmadi, and L. Liu (2006), Covariance Matrix Selection and Estimation via Penalized Normal Likelihood, *Biometrika*, 93, 85-98

[11] T. Leonard and J. S. J. Hsu (1992), Bayesian Inference for a Covariance Matrix, *Annals of Statistics*, 36, 1669-1696

[12] E. Levina, A. Rothman, and J. Zhu (2008), Sparse Estimation of Large Covariance Matrices via a Nested Lasso Penalty, *Annals of Applied Statistics*, 2, 1, 245-263

[13] H. Markowitz (1952), Portfolio Selection, *Journal of Finance*, 7, 77-91

[14] N. Meinshausen and P. Bühlmann (2006), High-Dimensional Graphs and Variable Selection with the Lasso, *The Annals of Statistics*, 34, 1436-1462

[15] M. Pourahmadi (1999), Joint mean-covariance models with applications to longitudinal data: Unconstrained parameterization, *Biometrika*, 86, 677-690

[16] R. Tibshirani (1996), Regression Shrinkage and Selection via the Lasso, *Journal of the Royal Statistical Society*, Series B, 58, 267-288

[17] R. S. Tsay (2005), Analysis of Financial Time Series, John Wiley & Sons, Inc.

[18] F. Wong, C. K. Carter, and R. Kohn (2003), Efficient Estimation of Covariance Selection Models, *Biometrika*, 90, 809-830

[19] W. B. Wu and M. Pourahmadi (2003), Nonparametric Estimation of Large Covariance Matrices of Longitudinal Data, *Biometrika*, 90, 831-844

[20] R. Yang and J. O. Berger (1994), Estimation of a Covariance Matrix Using the Reference Prior, *Annals of Statistics*, 22, 1195-1211

[21] Y. Q. Yin (1986), Limiting Spectral Distribution for a Class of Random Matrices, *Journal of Multivariate Analysis*, 20, 50-68

[22] M. Yuan and Y. Lin (2007), Model Selection and Estimation in the Gaussian Graphical Model, *Biometrika*, 94, 19-35

[23] H. Zou (2006), The Adaptive Lasso and Its Oracle Properties, *Journal of the American Statistical Association*, 101, 1418-1429