SVD-Based Screening for the Graphical Lasso

Yasuhiro Fujiwara[†], Naoki Marumo[‡], Mathieu Blondel[‡], Koh Takeuchi[‡], Hideaki Kim[‡], Tomoharu Iwata[‡], Naonori Ueda[‡]
[†]NTT Communication Science Laboratories, NTT Software Innovation Center, Osaka University
[‡]NTT Communication Science Laboratories
{fujiwara.yasuhiro, marumo.naoki, mathieu.blondel, takeuchi.koh,
kin.hideaki, iwata.tomoharu, ueda.naonori}@lab.ntt.co.jp

Abstract

The graphical lasso is the most popular approach to estimating the inverse covariance matrix of high-dimension data. It iteratively estimates each row and column of the matrix in a round-robin style until convergence. However, the graphical lasso is infeasible due to its high computation cost for large size of datasets. This paper proposes *Sting*, a fast approach to the graphical lasso. In order to reduce the computation cost, it efficiently identifies blocks in the estimated matrix that have nonzero elements before entering the iterations by exploiting the singular value decomposition of data matrix. In addition, it selectively updates elements of the estimated matrix expected to have nonzero values. Theoretically, it guarantees to converge to the same result as the original algorithm of the graphical lasso. Experiments show that our approach is faster than existing approaches.

1 Introduction

As a result of development of the Internet technologies, large datasets are common in many fields [Shiokawa *et al.*, 2015; Mishima and Fujiwara, 2015; Nakatsuji *et al.*, 2014]. Since real-world data typically has high-dimensionality, many researchers are interested in estimating the graph structure of a Gaussian Markov random field in the many variable setting. The graph structure is a natural representation of the conditional dependence among many variables. In the graph structure, each node is a single variable and an edge between two variables implies that they have conditional dependency [Fujiwara *et al.*, 2017; 2014]. For real-world data, the graph structure typically results in a complete graph where every pair of nodes is connected by an edge. This is because real-world data inevitably includes noise; some of the dependencies indicated by the structure might be due to noise. In order to increase the robustness of the graph structure against data noise, it is important to estimate a sparse structure from high-dimensional data so that the graph structure can effectively represent essential relationships among variables.

AI researchers have proposed approaches to estimating sparse undirected graphical models that exploit L_1 regularization [Grechkin *et al.*, 2015; Chen *et al.*, 2010]. Many studies assume that the observations have a multivariate Gaussian distribution with mean μ and covariance matrix $\mathbf{\Sigma}$. Thus, the conditional independence can be represented by the inverse covariance matrix, $\mathbf{\Theta}(= \mathbf{\Sigma}^{-1})$. If the (i, j)-th element of matrix $\mathbf{\Theta}$ is zero (i.e., $\Theta[i, j] = 0$), then the i-th and j-th variables are conditionally independent. The approaches for estimating sparse graph structures impose the penalty of L_1 to estimate matrix $\mathbf{\Theta}$ so as to increase its sparsity. The graphical lasso by Friedman et al. is the most popular approach [Friedman *et al.*, 2008]. It is based on penalized maximum-likelihood estimation and has been widely adopted by statisticians and computer scientists [Zhao *et al.*, 2015]. If $\mathbf{S}$ is an empirical covariance matrix such that $\mathbf{S} = \mathbf{X}^\top \mathbf{X}$ where $\mathbf{X}$ is a $N \times P$ data matrix of N observations and P variables, the graphical lasso estimates the sparse graph by maximizing the following objective function:

$$\log \det \mathbf{\Theta} - \mathrm{Tr}(\mathbf{S}\mathbf{\Theta}) - \rho \|\mathbf{\Theta}\|_1 \quad (1)$$

where det denotes the determinant, Tr is the trace, and $\| \cdot \|_1$ is the L_1 norm. In Equation (1), ρ is a positive L_1 regularization parameter that determines the sparseness of the estimated graph; if ρ is large, the graph has greater sparsity. Note that the problem of Equation (1) is convex [Friedman *et al.*, 2008]. The graphical lasso exploits a coordinate descent algorithm that updates coefficients of the lasso to solve the problem [Fujiwara *et al.*, 2016a; 2016b]. Specifically, it iteratively computes each row and column of the estimated matrix as a solution of the lasso problem in a round-robin manner until convergence. Since the graphical lasso can obtain the exact maximization of the above L_1-penalized log-likelihood, it can effectively estimate the graph structure from high dimensional data.

Due to its high effectiveness, the graphical lasso is actively used on the front lines of various fields. One example is market risk management by computing the portfolio risk of financial assets such as credit default swaps (CDS) [Neuberg and Glasserman, 2016]. In market risk management, it is crucial to accurately estimate a correlation matrix so as to minimize the risk of underestimating the true variance of the portfolio. By effectively reducing unimportant pairwise dependencies among assets, the graphical lasso can estimate the correlation matrix in an economically meaningful way. In fact, it has revealed high correlations such as New York Times - Gannett and Ford - General Motors. In addition, the graphical

lasso is being used in detecting psychiatric disorders [Rosa *et al.*, 2015]. Many psychiatric disorders, such as depression and schizophrenia, result from brain connectivity disorders. Thus, they are effectively detected from the fMRI signals of brains. Beyond those above, the graphical lasso is used in many other applications such as generating novel hypotheses in strategic management [Li *et al.*, 2016] and inferring biological networks [Vinciotti *et al.*, 2016].

Although the graphical lasso is widely used, the original algorithm by Friedman et al. incurs high computation cost. In order to improve the efficiency of the graphical lasso, a screening approach was proposed to locate blocks in the estimated inverse covariance matrix that can have nonzero elements [Witten *et al.*, 2011]. The approaches by Friedman et al. are a standard in estimating the inverse covariant matrix; the famous method, *glasso*, is based on the coordinate descent algorithm and the screening approach [Friedman *et al.*, 2015]. However, since the screening approach needs to compute all elements of the empirical covariance matrix before entering the iterations, it can incur high computation cost.

This paper proposes *Sting* as a novel and efficient approach for the graphical lasso. To improve the efficiency, we efficiently identify blocks in the estimated matrix that have nonzero elements by computing the singular value decomposition (SVD) of data matrix. Since the use of SVD avoids computing all elements of the covariance matrix, we can improve the efficiency of the graphical lasso. In addition, we selectively update coefficients of the lasso; we iteratively update only those coefficients expected to have nonzero values until convergence so as to increase the efficiency. After convergence, we check unselected coefficients to ensure they do not have nonzero values. Since our approach can safely prune unnecessary update computations, it provably guarantees to converge to the same result as the original algorithm of the graphical lasso. With the enhanced efficiency, the proposed approach can improve the effectiveness of AI-based decision-making, risk management, and data science. In the case of psychiatric disorder detection, disease diagnosis is an important application of AI [Xu *et al.*, 2016], and our approach can realize even small delays in the onset and in the progress of depression and schizophrenia; we can effectively reduce the burden of psychiatric disorders such as depression and schizophrenia by providing adequate institutional and home healthcare based on the proposed approach.

The remainder of this paper is organized as follows: Section 2 describes related work. Section 3 gives an overview of our research background. Section 4 introduces our approach. Section 5 reviews our experiments and their results. Section 6 provides our conclusions.

2 Related Work

The screening approach proposed by Friedman et al. is the most famous way of improving the efficiency of the graphical lasso [Witten *et al.*, 2011]. This approach computes small blocks of nonzero elements in the inverse covariance matrix. Theoretically, it is derived from the KKT condition of the optimization problem of Equation (1). Therefore, it does not sacrifice the exact maximization property of the graphical lasso. Hsieh et al. proposed approximation approaches based on Newton's method [Hsieh *et al.*, 2013]. It improves the efficiency of Newton's method in estimating the inverse covariance matrix since straightforward implementation of the method suffers high computation cost. However, this approach does not guarantee the same results as the graphical lasso unlike our proposal. Liu et al. proposed StART, which can effectively choose the regularization parameter for high dimensional inferencing of undirected graphs based on edge stability [Liu *et al.*, 2010]. It uses the least amount of regularization that simultaneously makes a graph sparse and replicable under random sampling. Liu et al. also proposed a nonparanormal approach that uses a semiparametric Gaussian copula for high dimensional inference [Liu *et al.*, 2012]. It efficiently computes the inverse covariance matrix by using nonparametric rank-based statistics to estimate the correlation matrix. In this paper, we proposed more efficient approach than the above approaches.

3 Preliminaries

Given $N \times P$ data matrix $\mathbf{X}$ of N observations and P variables, the graphical lasso estimates inverse matrix $\mathbf{\Theta}$ of covariance matrix $\mathbf{\Sigma}$ from matrix $\mathbf{X}$ by maximizing the penalized log-likelihood of Equation (1) [Friedman *et al.*, 2008]. Let $\mathbf{S}$ be an empirical covariance matrix, it optimizes each row and corresponding column of $\mathbf{W} = \mathbf{S} + \rho\mathbf{I}$ in coordinate descent fashion where $\mathbf{I}$ is an identity matrix. Specifically, it first partitions matrices $\mathbf{W}$ and $\mathbf{S}$ as follows:

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_{11} & \mathbf{w}_{12} \\ \mathbf{w}_{12}^\top & w_{22} \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} \mathbf{S}_{11} & \mathbf{s}_{12} \\ \mathbf{s}_{12}^\top & s_{22} \end{pmatrix}, \qquad (2)$$

where $\mathbf{W}_{11}$ and $\mathbf{S}_{11}$ are $p-1 \times p-1$ matrices, $\mathbf{w}_{12}$ and $\mathbf{s}_{12}$ are column vectors of length $p-1$, and w_{22} and s_{22} are scalars. The graphical lasso iteratively permutates each row/column of matrices $\mathbf{W}$ and $\mathbf{S}$ by exploiting the coordinate descent so that target columns are always the last to compute the solution of the lasso. Let $\hat{\beta}_i$ be the i-th coefficient of the lasso that corresponds to the i-column of matrix $\mathbf{W}_{11}$, the graphical lasso updates coefficient $\hat{\beta}_i$ as follows:

$$\hat{\beta}[i] \leftarrow S(s_{12}[i] - \textstyle\sum_{j \neq i} W_{11}[j,i]\hat{\beta}_j, \rho)/W_{11}[i,i] \qquad (3)$$

where S is the soft-threshold operator (i.e., $S(z,\rho) = \mathrm{sign}(z)(|z| - \rho)_+$), $s_{12}[i]$ is the i-th element of $\mathbf{s}_{12}$, and $W_{11}[j,i]$ is the (j,i)-th element of $\mathbf{W}_{11}$. Note that the diagonal of matrix $\mathbf{W}$ remains unchanged in the iterations. In addition, coefficients obtained by the lasso are typically sparse. If $\hat{\boldsymbol{\beta}}$ is a column vector of length $p-1$ whose i-th element is $\hat{\beta}[i]$, the graphical lasso fills in the corresponding column and row of matrix $\mathbf{W}$ by computing $\mathbf{w}_{12} = \mathbf{W}_{11}\hat{\boldsymbol{\beta}}$. These procedures are repeated until matrix $\mathbf{W}$ reaches convergence. The graphical lasso estimates elements of inverse matrix $\mathbf{\Theta}$ after the iterations. If $\hat{\boldsymbol{\theta}}_{12}$ is a column vector of length $p-1$ in matrix $\mathbf{\Theta}$ that corresponds to vector $\mathbf{w}_{12}$, it computes the estimated matrix as $\hat{\boldsymbol{\theta}}_{12} = -\hat{\boldsymbol{\beta}}/(w_{22} - \mathbf{w}_{12}^\top\hat{\boldsymbol{\beta}})$. This equation indicates that $\hat{\boldsymbol{\theta}}_{12}$ is a rescaling of vector $\hat{\boldsymbol{\beta}}$.

In order to reduce the high computation cost, Friedman et al. proposed the screening approach as described in Section 2

[Witten *et al.*, 2011]. It is based on the observation that the i-th variable would be an isolated node in the estimated graph if $|S[i,j]| \leq \rho$ holds for all j-th variables such that $j \neq i$ in empirical covariance matrix $\mathbf{S}$. Specifically, it computes an adjacency matrix where each pair of nodes is connected if $|S[i,j]| > \rho$ holds, and then computes blocks of connected components. The screening approach limits the update computations to those coefficients that are contained in the connected components; it can effectively reduce the number of coefficients of the lasso by exploiting the block structures. The screening approach is the most popular approach for implementing the graphical lasso. In fact, the standard method of the graphical lasso, *glasso*, improves the efficiency by exploiting the screening approach [Friedman *et al.*, 2015]. However, it still needs to compute all elements of the empirical covariance matrix to obtain the adjacency matrix. Therefore, the graphical lasso incurs high computation cost. If T is the number of update computations for each row/column of matrix $\mathbf{\Sigma}$ and B is the numbers of nonzero coefficients of the lasso for each row/column of the estimated matrix, the graphical lasso requires $O(NP^2 + PBT)$ time.

4 Proposed Method

The proposed approach, Sting, can efficiently estimate the inverse covariance matrix and so reduce the high computation cost of the graphical lasso. First, we overview the idea underlying Sting. That is followed by a full description including its theoretical properties.

4.1 Ideas

So as to reduce the number of coefficients updated in each iteration, the existing algorithms of the graphical lasso compute each covariance of matrix $\mathbf{S}$ and add an edge if $|S[i,j]| > \rho$ holds to obtain the block structures. This indicates that it must compute all the covariances of matrix $\mathbf{S}$. In addition, it updates all the coefficients obtained by the block structures in round-robin style even though almost all the coefficients are expected to be zero due to the sparseness property of the lasso. To reduce the computation cost, our approach efficiently computes upper and lower bounds of covariance $S[i,j]$ by exploiting SVD. Only if the covariance is determined to have an edge by the bounds, we exactly compute $S[i,j]$; this avoids computing all the covariances of matrix $\mathbf{S}$. In addition, Sting iteratively updates selected coefficients expected to have nonzero values until convergence. Since our approach updates only a few of the coefficients of the block structures, it can efficiently solve the lasso. Since our approach checks unselected coefficients whether they have nonzero values, it can provably guarantee to converge to the same results as the graphical lasso.

4.2 Upper and Lower Bounds

Before the iterations, we compute the upper and lower bounds of covariance $S[i,j]$. This approach approximates data matrix $\mathbf{X}$ of $N \times P$ size by computing SVD where the target rank is set to n ($n \ll N$). SVD is orthogonal transformation and gives high approximation quality in terms of squared loss [Press *et al.*, 2007]. Let $\mathbf{U}$ be a unitary matrix of $N \times n$ size and $\tilde{\mathbf{X}}$ be the transformed matrix of $n \times P$ size. Matrix $\mathbf{X}$ is represented as $\mathbf{X} \approx \mathbf{U}\tilde{\mathbf{X}}$. Therefore, let $\mathbf{x}_i$ and $\tilde{\mathbf{x}}_i$ be the i-th column vector of matrix $\mathbf{X}$ and $\tilde{\mathbf{X}}$, respectively, which yields $\mathbf{x}_i \approx \mathbf{U}\tilde{\mathbf{x}}_i$. The upper bound is defined as follows:

Definition 1 *Let $\overline{S}[i,j]$ be the upper bound of covariance $S[i,j]$, $\overline{S}[i,j]$ is given as follows:*

$$\overline{S}[i,j] = \tfrac{1}{2}\{\|\mathbf{x}_i\|_2^2 + \|\mathbf{x}_j\|_2^2 - \textstyle\sum_{k=1}^{n}(\tilde{x}_i[k] - \tilde{x}_j[k])^2\} \quad (4)$$

where $\|\cdot\|_2$ is the L_2-norm of a column vector.

Note that we can efficiently compute the L_2-norm of the column vectors in $O(NP)$ time before the iterations. Similarly, we can efficiently compute the SVD of matrix $\mathbf{X}$ by using the existing approach of [Halko *et al.*, 2011]; this takes $O(NP \log n)$ time. The lower bound is defined as follows:

Definition 2 *$\underline{S}[i,j]$ is the lower bound of covariance $S[i,j]$ and is given by the following equation:*

$$\underline{S}[i,j] = \tfrac{1}{2}\{-\|\mathbf{x}_i\|_2^2 - \|\mathbf{x}_j\|_2^2 + \textstyle\sum_{k=1}^{n}(\tilde{x}_i[k] + \tilde{x}_j[k])^2\} \quad (5)$$

We introduce the following lemma that $\overline{S}[i,j]$ and $\underline{S}[i,j]$ give the upper and lower bounds, respectively:

Lemma 1 *For each covariance $S[i,j]$ of matrix $\mathbf{S}$, we have $\underline{S}[i,j] \leq S[i,j] \leq \overline{S}[i,j]$.*

Proof We first prove that $S[i,j] \leq \overline{S}[i,j]$ holds. For each element of column vector $\mathbf{x}_i$ and $\mathbf{x}_j$, we have $x_i[k]x_j[k] = \frac{1}{2}\{(x_i[k])^2 + (x_j[k])^2 - (x_i[k] - x_j[k])^2\}$ [Blitzstein and Hwang, 2014]. Since $\mathbf{S} = \mathbf{X}^\top\mathbf{X}$, we have

$$\begin{aligned} S[i,j] = \mathbf{x}_i^\top \mathbf{x}_j &= \textstyle\sum_{k=1}^{N} x_i[k]x_j[k] \\ &= \tfrac{1}{2}\textstyle\sum_{k=1}^{N}\{(x_i[k])^2 + (x_j[k])^2 - (x_i[k] - x_j[k])^2\} \\ &= \tfrac{1}{2}\{\|\mathbf{x}_i\|_2^2 + \|\mathbf{x}_j\|_2^2 - \textstyle\sum_{k=1}^{N}(x_i[k] - x_j[k])^2\} \end{aligned}$$

Since SVD is orthogonal transformation, $\sum_{k=1}^{N}(x_i[k] - x_j[k])^2 = \sum_{k=1}^{N}(\tilde{x}_i[k] - \tilde{x}_j[k])^2$ holds. In addition, $(\tilde{x}_i[k] - \tilde{x}_j[k])^2 \geq 0$ holds. Therefore, we have

$$S[i,j] \leq \tfrac{1}{2}\{\|\mathbf{x}_i\|_2^2 + \|\mathbf{x}_j\|_2^2 - \textstyle\sum_{k=1}^{n}(\tilde{x}_i[k] - \tilde{x}_j[k])^2\} = \overline{S}[i,j]$$

Similarly, we have the property of $\underline{S}[i,j] \leq S[i,j]$ since $x_i[k]x_j[k] = \frac{1}{2}\{-(x_i[k])^2 - (x_j[k])^2 + (x_i[k] + x_j[k])^2\}$ holds for each element of column vector $\mathbf{x}_i$ and $\mathbf{x}_j$. □

In terms of computation cost, it takes $O(n)$ time to compute the bounds from Definition 1 and 2 if we have the L_2-norm of each column vector in matrix $\mathbf{X}$ and SVD of the matrix. This indicates that we can compute the upper and lower bounds, $\overline{S}[i,j]$ and $\underline{S}[i,j]$, much faster than $S[i,j]$ since it needs $O(N)$ time to compute $S[i,j] = \sum_{k=1}^{N} x_i[k]x_j[k]$.

As described in Section 3, the screening approach by Friedman et al. obtains the block structures by computing the adjacency matrix from matrix $\mathbf{S}$. We can effectively identify edges in the adjacency matrix by exploiting the bounds. Theoretically, our approach is based on the following lemma:

Lemma 2 *The node pair of the i-th and j-th variables must meet the condition of $\overline{S}[i,j] > \rho$ or $\underline{S}[i,j] < -\rho$ if the node pair have an edge in the adjacency matrix.*

Proof In order to prove Lemma 2, we show that the node pair of the i-th and j-th variables does not have an edge in the adjacency matrix if $\overline{S}[i,j] \leq \rho$ and $\underline{S}[i,j] \geq -\rho$. As described in Section 3, the adjacency matrix has an edge between the i-th and j-th variables if and only if $|S[i,j]| > \rho$ holds. If $\overline{S}[i,j] \leq \rho$ and $\underline{S}[i,j] \geq -\rho$, we have $-\rho \leq \underline{S}[i,j] \leq S[i,j] \leq \overline{S}[i,j] \leq \rho$ from Lemma 1. Therefore, the node pair does not have an edge in the adjacency matrix. As a result, if the node pair of the i-th and j-th variables has an edge, we have $\overline{S}[i,j] > \rho$ or $\underline{S}[i,j] < -\rho$. □

This lemma indicates that we can avoid computing $S[i,j]$ when obtaining edges in the adjacency matrix if $\underline{S}[i,j] \geq -\rho$ and $\overline{S}[i,j] \leq \rho$; we compute $S[i,j]$ only if $\underline{S}[i,j] < -\rho$ or $\overline{S}[i,j] > \rho$. Thus, we can avoid computing all the covariances of matrix $\mathbf{S}$ when exploiting the screening approach.

4.3 Selective Update

As described in Section 3, the screening approach updates the coefficients of the block structures in round-robin style even though almost all the coefficients are zero due to the sparseness property of the lasso. In order to further improve efficiency, we update only those coefficients expected to have nonzero values until convergence and then check unselected coefficients after the iterations.

Our approach is based on the observation that, if $|S[i,j]| > \rho$, the (i,j)-th element of inverse matrix $\mathbf{\Theta}$ is likely to have nonzero value and thus the corresponding coefficient of the lasso is expected to have nonzero value [Langfelder and Horvath, 2008]. Note that vector $\hat{\boldsymbol{\theta}}_{12}$ is a rescaling of vector $\hat{\boldsymbol{\beta}}$ as described in Section 3. Let B_i be the block that contains the i-th variable obtained by the screening approach and $\mathbb{B}_i$ be a set such that $\mathbb{B}_i = \{j | B_j = B_i\}$; the screening approach updates the i-th coefficient by exploiting $\mathbb{B}_i - i$. Instead, our approach updates the corresponding coefficients given by using $\mathbb{U}_i$ until convergence from Equation (3) for the i-th variable where $\mathbb{U}_i$ is initialized as $\mathbb{U}_i = \{j | S[i,j] > \rho \wedge j \in \mathbb{B}_i \wedge j \neq i\}$. Note that it is clear that $\mathbb{U}_i \subseteq \mathbb{B}_i$ and we can efficiently obtain $\mathbb{U}_i$ by exploiting the upper and lower bounds. In addition, if $\mathbb{U}'_i$ is a set such that $\mathbb{U}'_i = \{j | j \notin \mathbb{U}_i \wedge j \in \mathbb{B}_i \wedge j \neq i\}$, we check the coefficients in $\mathbb{U}'_i$ whether they have nonzero values after the iterations. Our approach exploits set $\mathbb{U}_i + \mathbb{U}'_i$ in computing the i-th coefficient. Since $\mathbb{U}_i + \mathbb{U}'_i = \mathbb{B}_i - i$ holds, our approach can obtain the same coefficient of the the i-th variable as the screening approach that exploits $\mathbb{B}_i - i$. This approach is theoretically based on the following property:

Lemma 3 *Let $\hat{\boldsymbol{\beta}}'$ be a converged coefficient vector for set $\mathbb{U}_i + \mathbb{U}'_i$. If (1) coefficient vector $\hat{\boldsymbol{\beta}}$ reaches convergence after the iterations for set $\mathbb{U}_i$ and (2) $\hat{\beta}[i] = 0$ and $\hat{\beta}'[i] = 0$ hold for all i such that $i \in \mathbb{U}'_i$, we have $\hat{\boldsymbol{\beta}}' = \hat{\boldsymbol{\beta}}$ after computing vector $\hat{\boldsymbol{\beta}}'$ from vector $\hat{\boldsymbol{\beta}}$ by using coordinate descent.*

Proof Since $\hat{\beta}[i] = 0$ and $\hat{\beta}'[i] = 0$ hold $\forall i \in \mathbb{U}'_i$, we have $\hat{\boldsymbol{\beta}}' = \hat{\boldsymbol{\beta}}$ for all i such that $i \in \mathbb{U}'_i$. If $\hat{\beta}'[i] = 0$ holds $\forall i \in \mathbb{U}'_i$, it is clear that such coefficients do not have any impact on the update results from Equation (3). In addition, coefficient vector $\hat{\boldsymbol{\beta}}$ has already reached convergence for $\mathbb{U}_i$. Therefore, $\hat{\beta}[i] = \hat{\beta}'[i]$ holds $\forall i \in \mathbb{U}_i$ after the update computations to obtain coefficient vector $\hat{\boldsymbol{\beta}}'$ if we use coefficient vector $\hat{\boldsymbol{\beta}}$ in the update computation. As a result, $\hat{\boldsymbol{\beta}}' = \hat{\boldsymbol{\beta}}$ holds. □

Lemma 3 indicates that, while the screening approach updates coefficients in $\mathbb{B}_i$ in round-robin style, our approach can selectively update the coefficients in $\mathbb{U}_i$ ($\mathbb{U}_i \subseteq \mathbb{B}_i$) without sacrificing accuracy by checking the coefficients in $\mathbb{U}'_i$. While the coefficients in $\mathbb{U}_i$ are iteratively updated, we do not perform the iterative computation for coefficients in $\mathbb{U}'_i$; this improves efficiency. In the proposed approach, if a coefficient in $\mathbb{U}'_i$ has a nonzero value, we add the coefficient to set $\mathbb{U}_i$ and recursively perform these procedures until we have $\hat{\beta}'[i] = 0$ for all i such that $i \in \mathbb{U}'_i$. We show a detailed algorithm of this approach in the next section.

Algorithm 1 Sting

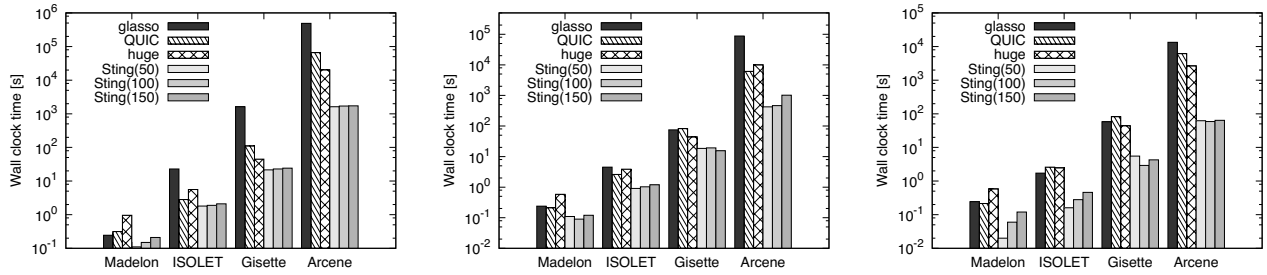
Input: matrix $\mathbf{X}$, parameter ρ, target rank of SVD n
Output: matrix $\mathbf{\Theta}$

```
1: compute rank-n SVD of matrix X;
2: A = 0;
3: for i = 1 to P do
4:     for j = 1 to P do
5:         compute the upper and lower bounds of S[i, j];
6:         if S̄[i, j] > ρ or S̲[i, j] < −ρ then
7:             compute S[i, j];
8:             if S[i, j] > ρ or S[i, j] < −ρ then
9:                 add edge e[i, j] to adjacency matrix A;
10: compute the blocks of connected components from adjacency matrix A;
11: for i = 1 to m do
12:     compute S^(i);
13:     W^(i) = S^(i) + ρI^(i);
14:     repeat
15:         for each j ∈ B_i do
16:             compute U_j;
17:             repeat
18:                 compute β̂ until convergence by using U_j;
19:                 compute U'_j;
20:                 for each k ∈ U'_j do
21:                     compute β̂[k] from Equation (3);
22:                     if β̂[k] ≠ 0 then
23:                         add k to U_j;
24:             until ∀k ∈ U'_j, β̂[k] = 0
25:             w_12^(i) = W_11^(i) β̂;
26:     until W^(i) reaches convergence
27:     compute Θ^(i) from W^(i);
    return Θ;
```

4.4 Algorithm

Algorithm 1 gives a full description of our approach. In Algorithm 1, $\mathbf{A}$ is the adjacency matrix of size $P \times P$ needed by the screening approach and m is the number of blocks obtained by the screening approach. By following the paper of the screening approach [Witten *et al.*, 2011], it processes each block one by one. In addition, let $\mathbf{S}^{(i)}$, $\mathbf{W}^{(i)}$, $\mathbf{I}^{(i)}$, $\mathbf{w}_{12}^{(i)}$, $\mathbf{W}_{11}^{(i)}$, and $\mathbf{\Theta}^{(i)}$ be matrix and vector of the i-th block that correspond to $\mathbf{S}$, $\mathbf{W}$, $\mathbf{I}$, $\mathbf{w}_{12}$, $\mathbf{W}_{11}$, and $\mathbf{\Theta}$, respectively.

Our approach starts by computing the SVD of data matrix $\mathbf{X}$ and initializing adjacency matrix $\mathbf{A}$ (lines 1-2). In order to efficiently obtain edges in the adjacency matrix, it computes the upper and lower bounds of each covariance of matrix $\mathbf{S}$ and, if the bounds meet the condition to have an edge (Lemma 2), it exactly computes elements of the em-



(1) $\rho = 0.5$ (2) $\rho = 0.7$ (3) $\rho = 0.9$

Figure 1: Estimation time of each approach.

pirical covariance matrix (lines 3-9). Then, it computes the blocks from matrix $\mathbf{A}$ (line 10). Following the screening approach, it computes the elements of matrix $\mathbf{\Theta}$ by exploiting the block structures. For each block, it first computes matrix $\mathbf{S}^{(i)}$ and $\mathbf{W}^{(i)}$ (lines 12-13). Then, it iteratively computes the coefficients by exploiting $\mathbb{U}_j$ until convergence (lines 15-18). It terminates the iterations if $\hat{\beta}[k] = 0$ holds for all k such that $k \in \mathbb{U}'_j$ based on Lemma 3 (lines 19-24). After the iterations, it updates matrix $\mathbf{W}^{(i)}$ (line 25). It performs these procedures until matrix $\mathbf{W}^{(i)}$ reaches convergence (line 26).

Sting has the following property for the computation cost:

Lemma 4 *Let t be the number of update computations for each row/column of the proposed approach, it requires $O(NP\log n + nP^2 + PBt + NP^2/m)$ time to estimate elements of the inverse covariance matrix.*

Proof As shown in Algorithm 1, it first computes the SVD of data matrix $\mathbf{X}$ that needs $O(NP\log n)$ time [Halko *et al.*, 2011]. To obtain the block structures, it computes the upper and lower bounds for all covariances of matrix $\mathbf{S}$ in $O(nP^2)$ time. In addition, it computes the covariances of matrix $\mathbf{S}$ corresponding to the blocks in $O(NP^2/m)$ time. So as to reduce the number of update computations, it first updates the coefficients expected to have nonzero values until convergence and then checks whether the other coefficients have nonzero values in $O(PBt+P^2/m)$ time. Therefore, the computation cost of Sting is $O(NP\log n+nP^2+PBt+NP^2/m)$. □

Sting has the following property for the estimation results:

Lemma 5 *Our approach converges to the same results as the graphical lasso in estimating matrix $\mathbf{\Theta}$.*

Proof As shown in Algorithm 1, it computes the (i,j)-th covariance of matrix $\mathbf{S}$ if we have $\overline{S}[i,j] > \rho$ or $\underline{S}[i,j] < -\rho$. This indicates that it does not compute the covariance of matrix $\mathbf{S}$ if $\overline{S}[i,j] \leq \rho$ and $\underline{S}[i,j] \geq -\rho$ hold. From Lemma 2, it is clear that the (i,j)-th covariance does not have an edge in the adjacency matrix in that case. Therefore, our approach can exactly obtain the same adjacency matrix as the graphical lasso. In addition, although it selectively updates coefficients in solving the lasso, it can obtain the same convergence results by checking unselected coefficients based on Lemma 3. As a result, since the optimization problem of Equation (1) is convex [Friedman *et al.*, 2008], the proposed approach converges to the same results as the graphical lasso. □

As shown in Theorem 4 and 5, the proposed approach theoretically converges to the same results as the graphical lasso with improved efficiency.

5 Experimental Evaluation

In this section, we experimentally evaluate the efficiency of our approach. We perform experiments on the datasets of *Madelon*, *ISOLET*, *Gisette*, and *Arcene*. These datasets have 600, 6238, 1000, and 700 observations, respectively. In addition, these datasets have 500, 618, 5000, and 10000 variables, respectively. Details of the datasets are shown in the UC Irvine Machine Learning Repository[1]. In the datasets, each column vector is standardized to have mean zero and variance one [Tan *et al.*, 2015]. In the experiments, we compare the proposed approach to *glasso*, *QUIC*, and *huge*. As described in Section 3, *glasso* is the most popular method for the graphical lasso based on the coordinate descent algorithm and the screening approach. *QUIC* is a recent method based on the Newton's method [Hsieh *et al.*, 2013]. Note that *QUIC* does not guarantee the same results as the graphical lasso as described in Section 3. In addition, *huge* is a state-of-the-art method for the graphical lasso that is based on the coordinate descent algorithm [Friedman *et al.*, 2008] and recent approaches [Liu *et al.*, 2012; 2010] detailed in Section 2. This method can yield the same results as the graphical lasso. In the experiments, we used the latest versions; *glasso* 1.8 [Friedman *et al.*, 2015], *QUIC* 1.1 [Hsieh *et al.*, 2015], and *huge* 1.2.7 [Zhao *et al.*, 2015]. In this section, "glasso", "QUIC", and "huge" represent the results of *glasso*, *QUIC*, and *huge*, respectively. In addition, "Sting(n)" denotes the results of our approach where we set the target rank of SVD to n. We set the convergence threshold to 10^{-4}. We conducted all experiments on a Linux 2.70 GHz Intel Xeon server. While *glasso* is implemented in Fortran, sting, *QUIC*, and *huge* are implemented in C/C++.

5.1 Efficiency

We evaluated the estimation time of each approach. Figure 1 shows the results where we set L_1 regularization parameter, ρ, to 0.5, 0.7, and 0.9. Note the results of the proposed approach include the time taken to compute the SVD of the data matrix before entering the iterations.

As shown in Figure 1, our approach is much faster than the previous approaches. Specifically, our approach is up to 300 times faster than *glasso*, up to 100 times faster than $QUIC$, and up to 45 times faster than $huge$. As described in Section 4.2, Sting does not compute all the elements of the empirical covariance matrix by exploiting the upper and

[1] https://archive.ics.uci.edu/ml/index.html

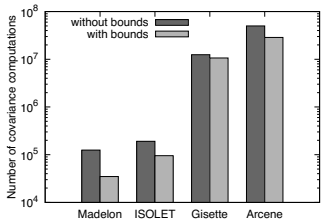


Figure 2: Number of covariance computations.

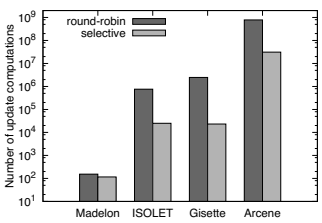


Figure 3: Number of update computations.

lower bounds. In addition, it selectively updates coefficients that are expected to have nonzero values as described in Section 4.3. As a result, our approach is superior to the existing approaches in terms of computation time as shown in Figure 1. This figure also indicates that the estimation time of our approach only slightly varies against the target rank of SVD, n. As described in Section 4.2, it takes $O(n)$ time to compute the upper and lower bounds by SVD. Therefore, we can increase the efficiency of computing the bounds if rank n have a small value. On the other hand, we can reduce the number of covariance computations if rank n has a large value. In conclusion, we can efficiently compute the bounds by reducing the target rank at the sacrifice of the number of covariance computations. Due to the trade-off derived from rank n, our approach is not so sensitive to the number of the target rank. In addition, Figure 1 indicates that the efficiency of Sting rises with ρ. This is because the proposed approach effectively exploits the sparseness property of the graphical lasso. Note that the proposed approach is theoretically guaranteed to converge to the same estimation results as the graphical lasso as shown in Theorem 5. Therefore, Figure 1 and Theorem 5 indicate that our approach can successfully reduce the computation time while providing the same guaranteed solution as the graphical lasso in estimating the inverse covariance matrix. Note that *huge* yield the same result as the graphical lasso while *QUIC* is an approximate approach.

5.2 Effectiveness

In the following experiments, we examine the effectiveness of the two core methods: (1) computing the upper and lower bounds and (2) selective update in the coordinate descent.

Covariance computations

As described in Section 4.2, we compute the upper and lower bounds of each covariance in matrix $\mathbf{S}$ to efficiently obtain the block structures used in the screening approach. In this section, we evaluate the number of covariance computations needed to evaluate the effectiveness of this approach. Figure 2 shows the results where we set $\rho = 0.7$ and $n = 100$. In this figure, "without bounds" represents the number of covariance computations where the upper and lower bounds are not used in obtaining the block structures; i.e., all the covariances in matrix $\mathbf{S}$ are computed. In addition, "with bounds" indicates the results when we used the upper and lower bounds in obtaining the block structures.

As shown in Figure 2, our approach can effectively reduce the number of covariance computations by exploiting the upper and lower bounds. For computing the block structures, the screening approach by Friedman et al. needs to compute all the covariances in matrix $\mathbf{S}$ to obtain the covariances that have high absolute values. Therefore, it incurs the high computation cost of $O(NP^2)$ where N and P are the numbers of observations and variables, respectively. Even though our approach needs to compute the bounds for all covariances in matrix $\mathbf{S}$, the computation cost is small since we can efficiently compute the bounds by using SVD of rank n; the computation cost of this procedure is $O(nP^2)$ where $n \ll N$. In addition, since we can effectively approximate the data matrix by exploiting SVD, our approach can successfully prune the unlikely covariances that have low absolute values. As a result, by exploiting the upper and lower bounds, we can effectively reduce the number of covariance computations needed to obtain the block structure in the screening approach.

Update computations

In order to improve the efficiency, we selectively update the coefficients expected to have nonzero values as described in Section 4.3. We plot the number of update computations needed to compute the coefficients in Figure 3 to show the effectiveness of this approach. In this figure, "round-robin" is the number of update computations where the coefficients are updated in round-robin style. In addition, "selective" indicates the results where the coefficients are selectively updated by using our approach. We set $\rho = 0.7$ and $n = 100$.

Figure 3 indicates that our approach can effectively reduce the number of updated computations. As described in Section 3, the graphical lasso computes the blocks of connected components from the adjacent matrix. While an edge between a variance pair in the adjacent matrix indicates that the variance pair has a covariance of high absolute value, the covariances can be included in the same block of connected components even if they do not have high absolute value. In addition, if covariance has low absolute value, the coefficient corresponding to the covariance is expected to be zero due to the L_1 penalty of the objective function [Langfelder and Horvath, 2008]. As a result, the coefficients in the graphical lasso have the property of sparseness. This indicates that the graphical lasso performs unnecessary update computations where many coefficients result in zero since it updates the coefficients of each block in round-robin style. On the other hand, our approach selectively updates only those coefficients that correspond to covariances with high absolute values. Since we do not iteratively compute the unlikely coefficients, we can effectively reduce the number of update computations in obtaining the inverse covariance matrix.

6 Conclusions

We proposed Sting, an efficient algorithm that improves the efficiency of the graphical lasso. Our approach computes the upper and lower bounds of covariances and selectively updates the coefficients of the lasso expected to have nonzero values. Experiments showed that it offers improved efficiency over existing approaches. Sting will allow many the graphical lasso-based applications to be processed more efficiently, and will improve the effectiveness of AI-based applications.

References

[Blitzstein and Hwang, 2014] Joseph K. Blitzstein and Jessica Hwang. *Introduction to Probability*. Chapman and Hall/CRC, 2014.

[Chen *et al.*, 2010] Xi Chen, Yan Liu, Han Liu, and Jaime G. Carbonell. Learning Spatial-temporal Varying Graphs with Applications to Climate Data Analysis. In *AAAI*, 2010.

[Friedman *et al.*, 2008] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse Inverse Covariance Estimation with the Graphical Lasso. *Biostatistics*, 9(3):432–441, July 2008.

[Friedman *et al.*, 2015] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. *Package 'glasso'*. ftp://cran.r-project.org/pub/R/web/packages/glasso/glasso.pdf, 2015.

[Fujiwara *et al.*, 2014] Yasuhiro Fujiwara, Go Irie, Shari Kuroyama, and Makoto Onizuka. Scaling Manifold Ranking bBsed Image Retrieval. *PVLDB*, 8(4):341–352, 2014.

[Fujiwara *et al.*, 2016a] Yasuhiro Fujiwara, Yasutoshi Ida, Junya Arai, Mai Nishimura, and Sotetsu Iwamura. Fast Algorithm for the Lasso Based L1-graph Construction. *PVLDB*, 10(3):229–240, 2016.

[Fujiwara *et al.*, 2016b] Yasuhiro Fujiwara, Yasutoshi Ida, Hiroaki Shiokawa, and Sotetsu Iwamura. Fast Lasso Algorithm via Selective Coordinate Descent. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 1561–1567, 2016.

[Fujiwara *et al.*, 2017] Yasuhiro Fujiwara, Naoki Marumo, Mathieu Blondel, Koh Takeuchi, Hideaki Kim, Tomoharu Iwata, and Naonori Ueda. Scaling Locally Linear Embedding. In *ACM SIGMOD*, pages 1479–1492, 2017.

[Grechkin *et al.*, 2015] Maxim Grechkin, Maryam Fazel, Daniela M. Witten, and Su-In Lee. Pathway Graphical Lasso. In *AAAI*, pages 2617–2623, 2015.

[Halko *et al.*, 2011] Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2):217–288, 2011.

[Hsieh *et al.*, 2013] Cho-Jui Hsieh, Mátyás A. Sustik, Inderjit S. Dhillon, Pradeep Ravikumar, and Russell A. Poldrack. BIG & QUIC: Sparse Inverse Covariance Estimation for a Million Variables. In *NIPS*, pages 3165–3173, 2013.

[Hsieh *et al.*, 2015] Cho-Jui Hsieh, Matyas A. Sustik, Inderjit S. Dhillon, and Pradeep Ravikumar. *Package 'QUIC'*. https://cran.r-project.org/web/packages/QUIC/index.html, 2015.

[Langfelder and Horvath, 2008] Peter Langfelder and Steve Horvath. WGCNA: an R Package for Weighted Correlation Network Analysis. *BMC Bioinformatics*, 9, 2008.

[Li *et al.*, 2016] Mei Li, Ying Lin, Shuai Huang, and Craig Crossland. The Use of Sparse Inverse Covariance Estimation for Relationship Detection and Hypothesis Generation in Strategic Management. *Strategic Management Journal*, 37(1):86–97, 2016.

[Liu *et al.*, 2010] Han Liu, Kathryn Roeder, and Larry A. Wasserman. Stability Approach to Regularization Selection (StARS) for High Dimensional Graphical Models. In *NIPS*, pages 1432–1440, 2010.

[Liu *et al.*, 2012] Han Liu, Fang Han, Ming Yuan, John D. Lafferty, and Larry A. Wasserman. High Dimensional Semiparametric Gaussian Copula Graphical Models. In *ICML*, 2012.

[Mishima and Fujiwara, 2015] Takeshi Mishima and Yasuhiro Fujiwara. Madeus: Database Live Migration Middleware under Heavy Workloads for Cloud Environment. In *ACM SIGMOD*, pages 315–329, 2015.

[Nakatsuji *et al.*, 2014] Makoto Nakatsuji, Yasuhiro Fujiwara, Hiroyuki Toda, Hiroshi Sawada, Jinguang Zheng, and James Alexander Hendler. Semantic Data Representation for Improving Rensor Factorization. In *AAAI*, pages 2004–2012, 2014.

[Neuberg and Glasserman, 2016] Richard Neuberg and Paul Glasserman. The Correlation Structure of Credit Default Swaps. *Columbia Business School Research Paper*, 2016.

[Press *et al.*, 2007] William H. Press, Saul A. Teukolsky, William T Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition*. Cambridge University Press, 2007.

[Rosa *et al.*, 2015] Maria J. Rosa, Liana Portugal, Tim Hahn, Andreas J. Fallgatter, Marta I. Garrido, John Shawe-Taylor, and Janaina Mourão Miranda. Sparse Network-based Models for Patient Classification Using fMRI. *NeuroImage*, 105:493–506, 2015.

[Shiokawa *et al.*, 2015] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka. SCAN++: Efficient Algorithm for Finding Clusters, Hubs and Outliers on Large-scale Graphs. *PVLDB*, 8(11):1178–1189, 2015.

[Tan *et al.*, 2015] Kean Ming Tan, Daniela M. Witten, and Ali Shojaie. The Cluster Graphical Lasso for Improved Estimation of Gaussian Graphical Models. *Computational Statistics & Data Analysis*, 85:23–36, 2015.

[Vinciotti *et al.*, 2016] Veronica Vinciotti, Ernst C. Wit, Rick Jansen, Eco J. C. N. de Geus, Brenda W. J. H. Penninx, Dorret I. Boomsma, and Peter A. C. 't Hoen. Consistency of Biological Networks Inferred from Microarray and Sequencing Data. *BMC Bioinformatics*, 17(1):254, 2016.

[Witten *et al.*, 2011] Daniela M. Witten, Jerome H. Friedman, and Noah Simon. New Insights and Faster Computations for the Graphical Lasso. *Journal of Computational and Graphical Statistics*, 20(4):892–900, 2011.

[Xu *et al.*, 2016] Zenglin Xu, Shandian Zhe, Yuan Qi, and Peng Yu. Association Discovery and Diagnosis of Alzheimer's Disease with Bayesian Multiview Learning. *JAIR*, 56:247–268, 2016.

[Zhao *et al.*, 2015] Tuo Zhao, Xingguo Li, Han Liu, Kathryn Roeder, John Lafferty, and Larry Wasserman. *Package 'huge'*. https://cran.r-project.org/web/packages/huge/huge.pdf, 2015.