A Distributable Convex Approach for Graph Structure Discovery

Chao Han, Nouf Albarakati, Xi Hang Cao, Zoran Obradovic
{chao.han,nouf,xi.hang.cao,zoran.obradovic}@temple.edu
Department of Computer and Information Science, Temple University
Philadelphia, Pennsylvania

ABSTRACT

Discovering the interdependent structure among variables plays a key role in knowledge discovery in many real-world applications. Given a sequence of p real-valued variables, the task is to estimate the entire graph structure with p^2 pairwise relationships. This problem is computationally challenging since the number of unknown relationships to estimate grows quadratically with respect to the number of variables. In order to solve this problem, many methods have been proposed to cast the structure discovery as an inverse covariance estimation problem by modeling the high dimensional data using a Gaussian graphical model. They focus on how to efficiently estimate the entire precision matrix by developing more advanced optimization algorithms in a sequential manner. A number of methods are also developed to select neighborhood or perform structure learning on categorical data, which are out of this study. A tuning-insensitive approach was proposed to estimate the precision matrix of Gaussian Graphical model in a distributed manner. But it over-parametrized the problem to achieve the tuning-insensitivity. In this study, we proposed a novel framework to discover the underlying graph structure in a distributed manner with a straightforward parametrization. The idea is to decompose the structure discovery task into multiple sub-tasks, such that a column of the precision matrix is estimated in a sub-task. We also proved the distributability and the convexity of the global task. Additionally, we empirically demonstrated the effectiveness and efficiency of our proposed framework by conducting extensive experiments with comparison to a number of state-of-the-art methods on synthetic datasets. Our case study on a real-world application is also demonstrated to be reliable.

KEYWORDS

Structure Discovery, Probabilistic Graphical Model

ACM Reference Format:
Chao Han, Nouf Albarakati, Xi Hang Cao, Zoran Obradovic. 2019. A Distributable Convex Approach for Graph Structure Discovery. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In numerous important real-world problems, for example, forecasting stock price on financial markets [31], predicting user intents on social networks [20], ranking querying results in information retrieval [4], co-expression study of various genes on bioinformatics [28], and functional understanding of different brain regions [17], data lie in a high dimensional space due to the co-existence of many interdependent variables. Rather than recognizing individual behavior of a variable, people are often more interested in how variables affect each other. All of these applications therefore raise a problem of discovering the structure of the underlying weighted graph, where variables are regarded as nodes, and the weight on the edge is the strength of the connection between two variables. As a result, downstream applications benefit tremendously from the discovery of the inter-relationships among variables. In practice, the evolution of stock price can be better understood and predicted by inferring the dependencies among different stocks [10, 31], and the recommendation of querying results is more reliable by considering the inner structure among them [23].

To recover the underlying dependent structure among variables, the joint distribution of all variables is typically modeled as a Multivariate Normal Distribution with zero mean and inverse covariance matrix Λ, also known as precision matrix. So the structure discovery problem is converted into the precision estimation problem. However, the precision estimation is challenging since the number of unknown pairwise dependencies is quadratic with respect to the number of variables.

Gaussian Graphical Model (also known as Gaussian Markov Random Fields) [21, 25] has hence been proposed to solve the precision estimation problem by modelling the joint distribution of variables as a Multivariate Normal distribution. Graphical Lasso (also known as Sparse Inverse Covariance Estimation) [1, 6, 30], as a more efficient solver for precision estimation, is able to estimate a sparse precision matrix by adding a Lasso penalty to the Gaussian Graphical Model. Many studies afterwards contributed to the efficiency of Graphical Lasso by developing various faster optimization algorithms [5, 10, 13, 14, 16, 26, 27]. However, given the fact that the entire precision matrix is always involved in the calculation of sub-gradients, all of the aforementioned methods have to be estimate all entries from the precision matrix in a sequential manner.

A computationally efficiently method was proposed to discover the graph structure by using Lasso to identify relevant neighborhoods for each variable. This method however focuses on the selection of neighboring variables and cannot estimate the precision matrix [19]. A tuning-insensitive approach is proposed to estimate the Gaussian Graphical Model column-by-column [18]. To guarantee tuning-insensitivity, this method is computationally costly due to required over-parameterization of estimation.

In our study, a new framework is proposed to discover the structure in a distributed manner. Unlike traditional methods for inverse covariance estimation, our method proposes to model joint product of the conditional probabilities of each variable given neighboring variables. In this way, one can decompose the entire estimation task into many sub-tasks, and estimate a single column of the precision matrix in each sub-task. Also, different from previously proposed computationally efficient methods, our method developed a more straightforward formulation of the estimation problem with fewer computations. Although it requires the tuning of hyperparameter, it is still practical for real-world applications as there is only one hyperparameter. We also prove distributability and convexity of our proposed method. Furthermore, we provide empirical evidence for the effectiveness and efficiency by comparing with several state-of-the-art methods for precision estimation on synthetic datasets across extensive experimental settings. The results provide evidence that our proposed model is more effective regardless of size of training data. Our method has a considerable speed advantage over alternative approaches even without parallelization. It is several orders of magnitudes faster than alternative methods when all sub-tasks are ideally distributed. We applied the proposed method for discovering the underlying relationships of daily stock price between different companies. The discovery has turned out to be very consistent with the real-world facts.

In summary, following are the main contributions of this study:

- We proposed a distributed framework to estimated the precision matrix from high dimensional data with a straightforward parameterization.
- We proved that the proposed framework is characterized by distributability and convexity.
- We conducted extensive experiments on synthetic datasets to show the efficiency and effectiveness of our framework. Discoveries from our method are also validated to be reliable on real-world case studies.

2 RELATED WORK

In this section, we present the development of methods for precision estimation, and introduce relevant works on the general dependency network.

Gaussian Graphical Model [21, 25] is proposed to identify the inter-relationships among multiple continuous random variables. However, as the number of pairwise relationship grows quadratically with the increment of the number of the variables, Gaussian Graphical Model is not feasible for precision estimation in many real-world problems. To efficiently address the precision estimation problem, Graphical Lasso was developed to learn a sparse precision matrix. A number of papers have investigated in solving sparse precision estimation in a more efficient way. Initially, a block coordinate descent method was developed to solve the dual form of Graphical Lasso [1]. Inspired of this work, a coordinate descent method was proposed to solve the row-subproblem in the dual form as well [6]. Later, an augmented Lagrangian method was applied to solve the smooth log-likelihood and non-smooth regularization part in an alternative way [26]. The same group also proposed to solve the primal form of Graphical Lasso via a greedy coordinate descent algorithm [27]. A projected Quasi-Newton method was also proposed to solve the primal form [5]. In contrast to these first order methods, QUIC, a second order Newton coordinate descent method, was developed to solve Graphical Lasso based on the quadratic approximation of its objection [14, 15]. As the state-of-the-art for sparse precision estimation, QUIC was optimized in further studies by either being applied to solve sub-problems via divide and conquer [13], or paralleling the block coordinate descent algorithm [16]. Recently, a new study also develop a message-passing algorithm using Alternating Direction Method of Multipliers (ADMM) [2] to efficiently solve time-varying graphical Lasso [10]. Although these methods are successfully applied, their capability is bounded by the fact that the entire precision matrix has to be estimated simultaneously.

In contrast to any of described approaches, our proposed method can completely discover the structure from distributable sub-tasks by modelling the local dependencies of each variable. From this point of view, the general dependency network is relevant to our research [12]. It models the inter-dependencies among variables via a cyclic Bayesian network, such that the joint probability of variables can be approximated by the product of conditional probability of each variable given the others. This is previously studied with a more general assumption of distribution (e.g., exponential family [29]), in which Gibbs Sampling is used for density estimation. This idea is also adapted for multiple output prediction problems including multi-label classification [8, 9], and structured regression [11], where the inter-connections among output variables are learned while performing prediction. Previous work however concentrates on the predictive modelling in which each output variable is assumed to depend on its feature vector and all other output variables.

A neighborhood selection method was proposed to discovery the graph structure by applying lasso to select relevant variables for each variable [19]. Although this approach can select variables, it is unable to exactly estimate the precision matrix. Basically, neighborhood selection is a sub-task of the precision estimation. A similar idea is also applied for the structure learning of the Ising model [24], in which the conditional probability of an individual binary variable given other variables is modelled to discover the structure rather than modelling the joint probability of all variables. Our work is different as we specifically focus on the precision estimation of the Gaussian Graphical model where all variables are real-valued. A tuning-free method is introduced to decompose the estimation of the precision matrix into estimations of columns [18]. However, this method cannot directly estimate the columns of the precision matrix as it estimates intermediate variables with more computations to achieve tuning-insensitivity. Our work provides a more straightforward formulation of the estimation problem such that the columns of the precision matrix are estimated directly and in parallel. We demonstrate computational advantages of this framework theoretically and empirically.

3 PROBLEM STATEMENT

Given a problem with m data examples of p continuous variables, let $x_i^j \in \mathcal{R}$ denotes variable i at the j_{th} data example, $\bar{x_i}^j$ denotes all except variable i at the j_{th} data example. Similarly, $\mathbf{x}_i \in \mathcal{R}^m$ denotes the collection of variable i on all data examples, and $\bar{X_i} \in \mathcal{R}^{m\times(p-1)}$ denotes the collection of all variables except variable i at all data examples. All variables in the j_{th} data example is represented as $\boldsymbol{x}^j \in \mathcal{R}^p$.

The task of structure discovery is to discover the interdependencies among all p variables given the data $X \in \mathcal{R}^{m\times p}$. In other words, we would like to identify whether two variables are conditional independent or not given other variables. This structure discovery problem is typically tackled by Gaussian Graphical Model.

4 GAUSSIAN GRAPHICAL MODEL

In Gaussian Graphical Model, the collection of variables $\boldsymbol{x} \in \mathcal{R}^p$ is assumed to follow a Multivariate Normal Distribution with zero mean.

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \Lambda^{-1})$$

where Λ is the precision matrix.

With m i.i.d. data examples $\boldsymbol{x}^1, \boldsymbol{x}^2, \cdots, \boldsymbol{x}^m$, the structure discovery problem is formulated as estimating the precision matrix such that the log-likelihood of data is maximized.

$$\Lambda^* = \underset{\Lambda\succeq 0}{\operatorname{argmax}} \frac{1}{m}\sum_{j=1}^{m} \log P(\boldsymbol{x}^j; \Lambda)$$

Sparse inverse covariance estimation is another classical extension to Gaussian Graphical model. It is proposed to accelerate the discovery of structure by assuming the precision matrix is off-diagonally sparse. It is formulated as

$$\Lambda^* = \underset{\Lambda\succeq 0}{\operatorname{argmax}} \frac{1}{m}\sum_{j=1}^{m} \log P(\boldsymbol{x}^j; \Lambda) + \alpha\|\Lambda\|_*$$

, where $\|\cdot\|_*$ is a ℓ_1 norm penalty on off-diagonal entries.

5 PROPOSED METHOD

As we can see from the precision estimation problem, the problem size (p^2) easily becomes huge since it grows quadratically with respect to the size of variables (p), which makes it computational costly to solve in practice. All introduced works for precision estimation are limited by the fact that all entries of the precision matrix have to be estimated simultaneously. In our study, instead of trying to discover all pieces of the entire structure in one job, we propose a distributed structure discovery (DSD) framework which can completely recover the underlying structure distributedly.

5.1 Illustration

The idea is very straightforward.Rather than considering the complicated interconnections between all variables together, we factorize the problem into multiple sub-problems, while only exploring the sub-dependencies related to one variable in each sub-problem. The comparison of two frameworks with a concrete example, which involves four variables $\{x_1, x_2, x_3, x_4\}$, is illustrated in Figure 1.

In Figure 1(a), the structure discovery task is addressed by fitting the data into a Gaussian Graphical Model such that the joint distribution is a multivariate Normal $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \Lambda^{-1})$. In this example, all undirected pairwise dependencies are marked using solid edges. For example, Λ_{12} and Λ_{21} from the precision matrix Λ are associated with the pairwise connection between x_1 and x_2.

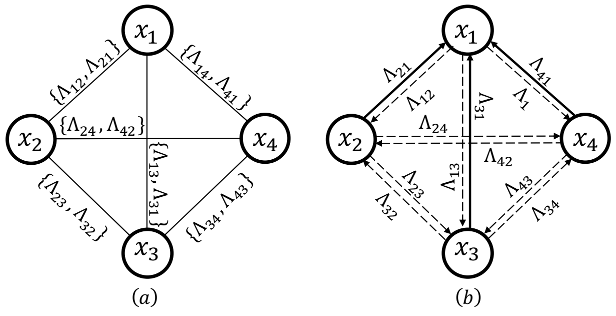

Figure 1: Comparison of structure discovery and distributed structure discovery. (a) The standard precision estimation framework with joint distribution $P(x_1, x_2, x_3, x_4)$. (b) DSD framework from the local view of x_1: $P(x_1|x_2, x_3, x_4)$)

However, under the framework of distributed structure discovery, the sub-task i only explores the dependencies towards x_i by maximizing the conditional probability $P(x_i|\bar{\boldsymbol{x}_i})$, where $\bar{\boldsymbol{x}_i} = [x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_p]$ is a collection of all variables except x_i. The sub-problem for x_1, $P(x_1|\bar{\boldsymbol{x}_1}) = P(x_1|x_2, x_3, x_4)$, is presented in Figure 1(b), where the all local dependencies (i.e., Λ_{21}, Λ_{31} and Λ_{41}) towards $\boldsymbol{x}_1$ are marked using solid directed edges and other dependencies are marked using dashed directed edges. To reveal the entire structure on this specific example, DSD aims to maximize the product of all conditional likelihood in all sub-problems:

$$\max P(x_1|\bar{\boldsymbol{x}_1})P(x_2|\bar{\boldsymbol{x}_2})P(x_3|\bar{\boldsymbol{x}_3})P(x_4|\bar{\boldsymbol{x}_4})$$

5.2 Distributed Structure Discovery

Without the loss of generality, the negative pseudo log-likelihood of the proposed method can be written as

$$L = \sum_{i=1}^{p} L_i = \sum_{i=1}^{p} -\frac{1}{m}\sum_{j=1}^{m} \log P(x_i^j|\bar{\boldsymbol{x}_i}^j; \Lambda_i) \tag{1}$$

, where $\Lambda_i = [\Lambda_{ii}, \cdots, \Lambda_{pi}]$ is the i_{th} column of the precision matrix Λ. The structure discovery problem using DSD is hence formulated as

$$\begin{aligned} \Lambda^* = [\Lambda_1^*, \cdots, \Lambda_p^*] = \underset{[\Lambda_1,\cdots,\Lambda_p]}{\operatorname{argmin}} \; L \\ \text{subject to } \Lambda_{11} > 0, \cdots, \Lambda_{pp} > 0 \end{aligned} \tag{2}$$

So far we have not discussed why this model works and how it works. To clearly answer these questions, in the following sections, we introduce two characteristics of our method: distributability and convexity.

5.3 Distributability

In this section, we prove why the original precision estimation problem can be solved distributedly without information loss, and introduce the benefits brought by this nice property.

THEOREM 1. *Given the assumption that $\boldsymbol{x} = [x_i; \bar{\boldsymbol{x}_i}]$ follows a multivariate normal distribution*

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \begin{bmatrix} \Lambda_{ii} & \Lambda_{\bar{\boldsymbol{x}}_i x_i}^T \\ \Lambda_{\bar{\boldsymbol{x}}_i x_i} & \Lambda_{\bar{\boldsymbol{x}}_i\bar{\boldsymbol{x}}_i} \end{bmatrix}^{-1}) \tag{3}$$

the conditional distribution of x given $\bar{\mathbf{x}}$ is a univariate Gaussian distribution

$$x_i|\bar{\mathbf{x}_i} \sim \mathcal{N}(-\Lambda_{ii}^{-1}\Lambda_{\bar{\mathbf{x}}_i x_i}^T \bar{\mathbf{x}}_i, \Lambda_{ii}^{-1}) \quad (4)$$

where Λ_{ii} is the inverse covariance of x_i, $\Lambda_{\bar{\mathbf{x}}_i x_i}$ is Λ_i without Λ_{ii}, and $\Lambda_{\bar{\mathbf{x}}_i \bar{\mathbf{x}}_i}$ is the partial precision matrix among $\bar{\mathbf{x}}_i$ in Λ

PROOF. It can be proved by simply expanding the exponent part of (3) and omitting independent terms of x_i first, and then matching the remained terms with the exponent part of a univariate Gaussian. ■ □

According to Theorem 1, we have: (1) The conditional probability $P(x_i|\bar{\mathbf{x}_i};\Lambda_i)$ is a univariate Gaussian, which has analytic expression for its probability density function; (2) The i_{th} column of Λ, Λ_i, can be recovered from the i_{th} sub-task. So the structure discovery task can be completely accomplished from p sub-tasks. (3) Although Λ_{ij} and Λ_{ji} are estimated in two different sub-tasks (sub-task i and sub-task j), their estimation are still guaranteed to be very similar (not exactly the same due to the noise from data). In the applications where symmetric inverse covariance matters, it can be approximated by

$$\Lambda = \frac{1}{2}(\hat{\Lambda} + \hat{\Lambda^T}) \quad (5)$$

It was proved that Λ is a good estimator if $\hat{\Lambda}$ is a good estimator [3].

Table 1: Time complexity comparison of different frameworks. p is the number of nodes. $f(\cdot)$ is the time complexity of optimization methods used in existing works. $g(\cdot)$ is the time complexity of optimization method used for each sub-task in DSD

Framework	Time Complexity
existing works	$O(f(p^2))$
DSD	$O(p * g(p))$
Ideal DSD	$O(g(p))$

Moreover, the distributability also suggests the superiority of DSD in terms of efficiency compared with existing frameworks. The comparison of time complexity of different frameworks is summarized in Table 1. Let $f(\cdot)$ denote the time complexity function for the optimization algorithm in existing works. Then the time complexity for existing precision estimation frameworks is given by $O(f(p^2))$, where p is the number of variables, and p^2 is the number of unknown pairwise dependencies to estimate. Let $g(\cdot)$ denotes the time complexity function for the optimization algorithm used in each sub-task of DSD, then the time complexity for solving all sub-tasks in sequential is $O(p * g(p))$ since there are only p unknown dependencies in each sub-task. The time complexity of solving all sub-tasks in parallel (Ideal DSD) is hence $O(g(p))$. Given that $g(n) \in \Theta(f(n))$, we can draw a conclusion that

$$O(g(p)) << O(p * g(p)) < O(f(p^2)) \quad (6)$$

. More empirical evidence about efficiency is provided in Section 6.3.

5.4 Convexity

Thanks to the distributability, all partial structure are guaranteed to be completely recovered from all sub-tasks in DSD. In this section, we show why the partial structure can be precisely recovered. By re-denoting Λ_{ii} using λ_i, and $\Lambda_{\bar{\mathbf{x}}_i x_i}$ using $\boldsymbol{\theta}_i$, and adapting (4) from Theorem 1 to the i_{th} sub-problem properly, we have

$$x_i|\bar{\mathbf{x}_i} \sim \mathcal{N}(-\lambda_i^{-1}\boldsymbol{\theta}_i^T \bar{\mathbf{x}_i}, \lambda_i^{-1}) \quad (7)$$

The negative log-likelihood L_i is hence written as:

$$\begin{aligned} L_i &= -\frac{1}{m}\sum_{j=1}^{m} \log P(x_i^j|\bar{\mathbf{x}_i}^j;\lambda_i,\boldsymbol{\theta}_i) \\ &\triangleq \frac{1}{m}\sum_{j=1}^{m}(x_i^j + \lambda_i^{-1}\boldsymbol{\theta}_i^T\bar{\mathbf{x}_i}^j)^2\lambda_i - \log\lambda_i \quad (8) \\ &= \frac{1}{m}\|\mathbf{x}_i + \lambda_i^{-1}\bar{X}_i\boldsymbol{\theta}_i\|_2^2\lambda_i - \log\lambda_i \end{aligned}$$

Then the convexity of DSD is proven in Theorem 1.

THEOREM 2. *The distributed structure discovery framework described in (2) is global convex.*

PROOF. The idea is to show that the optimization problem in each sub-task is convex. Then the entire problem can be proved to be convex since the parameters in sub-tasks are distinct to each other. In order to prove that each sub-task is convex, we need to show that the Hessian matrix H is PSD, where the Hessian matrix for the i_{th} sub-task is given by:

$$H_i = \begin{bmatrix} A_i & B_i \\ B_i^T & C_i \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 L_i}{\partial \boldsymbol{\theta}_i{}^2} & \frac{\partial^2 L_i}{\partial \boldsymbol{\theta}_i \partial \lambda_i} \\ \frac{\partial^2 L_i}{\partial \boldsymbol{\theta}_i \partial \lambda_i}^T & \frac{\partial^2 L_i}{\partial \lambda_i{}^2} \end{bmatrix}$$

As we can easily prove that $C_i \geq 0$ and $A_i - B_iC_i^{-1}B_i^T \geq 0$, the Hessian matrix H_i is therefore proved to be PSD based on Schur complement [7]. ■ □

Now, we formally demonstrate that partial structure from sub-tasks can be precisely recovered since the convexity ensures the estimation to be global optimal. Combining distributability and convexity, the proposed method is justified to be effective, because the entire underlying structure can be completely and precisely revealed from sub-tasks. More empirical evidence are provided in Section 6.4.

Regularization In order to avoid overfitting, we added a regularizor to penalize the learning of off-diagonal entries of a partial precision matrix in each sub-task. The optimization of sub-task i is rewritten as:

$$\underset{\lambda_i>0,\boldsymbol{\theta}_i}{\operatorname{argmin}} \frac{1}{m}\|\mathbf{x}_i + \lambda_i^{-1}\bar{X}_i\boldsymbol{\theta}_i\|_2^2\lambda_i - \log\lambda_i + \alpha\phi(\theta_i) \quad (9)$$

where α is the hyperparamter which controls the penalty on off-diagonal entries and $\phi(\cdot)$ is the regularization function. In this study, we specifically use ℓ_2 norm as the realization of $\phi(\cdot)$ for the ease of prototyping. This can be easily extended to use ℓ_1 norm or any other (convex) regularizors by adapting various solvers like proximal algorithm [22]. Although ℓ_2 norm doesn't lead to sparse solutions, real-world cases can be studied by retrieving the most (or least) significant connections from our estimation (see more details in Section 7). The parameter study of α is presented at Section 6.5.

Algorithm 1 DSD: Distributed Structure Discovery

```
Input: $X$ and $\alpha$.
1: Initialization: $\lambda$, $\boldsymbol{\theta}$.
2: for $i$ from 1 to $p$ do                    ▷ Distributing here
3:     Prepare $\boldsymbol{x}_i$ and $\bar{X}_i$ from $X$
4:     Solve $\boldsymbol{\theta}_i^*, \lambda_i^* = \mathrm{argmin}_{\lambda_i > 0, \boldsymbol{\theta}_i} L_i(\boldsymbol{x}_i, \bar{X}_i; \lambda, \boldsymbol{\theta}, \alpha)$
5:     Assign $\Lambda_{ii}^* = \lambda_i^*$ and $\Lambda_{\bar{i}i}^* = \boldsymbol{\theta}_i^*$
6: Return $\Lambda^*$.
```

The overall learning algorithm for DSD is summarized in Algorithm 1, where $\Lambda_{\bar{i}i}$ denotes the i_{th} column of Λ without the i_{th} entry. We apply Quasi-Newton to solve each convex sub-task in our implementation.

6 EXPERIMENTS

Previously we explained why our proposed method is efficient and effective (accurate) given its nature of distributability and convexity. In order to find quantitative evidence for efficiency and effectiveness, we conducted experiments on the synthetic dataset where the ground truth is available.

6.1 Comparison Methods

To characterize the capability of our proposed method, we compare it against two state-of-the-art methods for structure discovery. For all of experiments related to DSD in this section, we directly evaluate on the precision matrix learned from DSD without conducting any post-processing (e.g., nearest symmetric estimation).

- QUIC: It is the state-of-the-art algorithm for sparse inverse covariance estimation using the Newton coordinate descent method [14, 15]. In our experiment, we used its Python implementation [1] for comparison. We do not compare with Big&Quic [16] since it is an extension of QUIC with more efficient optimization method, while the comparison aims to show the difference between our proposed distributed framework and the existing framework.
- TVGL: Time-varying Graphical Lasso is the most recent work that solves structure discovery on time-varying networks by developing a novel ADMM algorithm with closed-form solutions on sub-problems [10]. We compare with it using its original implementation in Python [2]. We apply it on stationary networks by setting the penalty of temporal difference as 0 and assume there is only one time window. We use TVGL with $\ell 1$ penalty in our experiments as choosing other penalties for TVGL leads to worse and much slower performance in practice.

6.2 Synthetic Data Generation

Before stepping into experiments, we explain the process of generating synthetic datasets. The overall idea is to sample data from a multivariate normal distribution with a predefined precision matrix. Given a graph with p nodes and p^2 variables, we initialize the precision matrix $\Lambda \in \mathcal{R}^{p \times p}$ by randomly sampling each entry from a uniform distribution. Then, we make it symmetrical by averaging its lower-triangular entries to its upper-triangular entries, and gradually increase the values of diagonal entries until it becomes diagonally dominant and hence Positive Semidefinite (PSD). After that, we sample the data from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Lambda}^{-1})$ with zero mean and precision matrix as Λ. In order to comprehensively evaluate the performances (effectiveness & efficiency) of comparison methods at different problem scales in terms of the number of unknown dependencies (#dep), we generate several datasets with different number of nodes (p) varying in $[100, 200, 300, 400, 500]$ in our experiments. For each dataset, we generate 3000 data examples.

[1] https://github.com/skggm/skggm

[2] https://github.com/davidhallac/TVGL

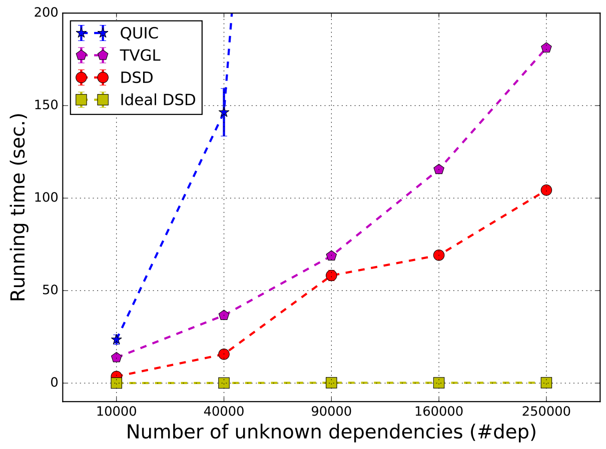

Figure 2: Efficiency evaluation on different problem scales. DSD presents the actual running time of our model, and Ideal DSD presents the estimated running time when all sub-tasks are fully distributed.

6.3 Evidence for the Efficiency

In this experiment, we demonstrate the superior efficiency of our method brought by its nature of distributability.

We conduct experiments on five datasets with different problem scales (#dep ranges from 10000 to 250000). At each scale, all comparison methods are trained on 3 different 1000 random data examples for each choice of regularization parameter α. The hyperparameter α is selected from $[0, 1e-4, 1e-3, 1e-2, 1e-1, 1]$. We use the hyperparameter which obtains the lowest prediction error in predicting ground truth precision matrix. Then we report the corresponding efficiency of the selected hyperparameters. The results are presented using the mean and standard deviation of running time of 3 trials. All of the experiments from this section are conducted in sequential on the same machine.

The results are presented in Figure 2. Here, DSD stands for the actual running time our proposed method where sub-tasks are executed sequentially. Ideal DSD stands for the estimated running time of the ideal case when all sub-tasks are fully distributed. The running time of Ideal DSD is approximated by the running time of DSD divided by p, (i.e., the number of sub-tasks). From the result, we can see that TVGL is around 5 to 10 times faster than QUIC given the advantage of their proposed ADMM solver [10]. DSD is however up to 2 times faster than TVGL. DSD brings a clear benefit because of its lightweight computational framework mentioned in Table 1. If we compare Ideal DSD with other approaches, it is at least 2 orders of magnitude (100 times) faster than any of them. For example, it can solve problems which other competing approaches take hours to solve within 1 second. Although the performance of Ideal DSD is the theoretical lower bound of DSD, the gap between the efficiency of Ideal DSD and DSD still indicates the potential space for DSD to be improved. The acceleration of DSD is decided by the number of distributable computing resources and the protocol for distribution.

6.4 Evidence for the Effectiveness

In this experiment, we exhibit the effectiveness of our proposed method. Given the availability of ground truth in synthetic data, the performance is evaluated in terms prediction error (MSE) in predicting the ground truth precision matrix.

6.4.1 On the Generalization Performance. First, we investigate the generalization performance of all approaches by showing how prediction error changes when the size of training data examples varies. We demonstrate that our proposed DSD is capable of generalizing well with less training data examples. Specifically, we conduct experiments on four datasets with different problem scales (number of unknown dependnecies) from $[10000, 40000, 90000, 160000]$. For each problem scale, we conduct experiments with different train sizes: $m \in [200, 400, 600, 800, 1000]$. For each train size, the experiment is repeated on three different m examples for hyperparameter tuning. The hyperparameter is also selected from $[0, 1e-4, 1e-3, 1e-2, 1e-1, 1]$.

The results are presented in Figure 3. Concerning MSE, our proposed method outperforms the other two baselines across most of the experimental configurations except cases when the train size is very limited ($m = 200$). It is consistent with our assumption that DSD has smaller model complexity (proportional to the number of parameters) given its nature of distributability and hence generalize better. From the results, we also note that DSD generalizes worse on datasets with fewer variables, and better on datasets with more variables. For example, on both Figure 3a and Figure 3b, the gaps between the best MSE and the worse MSE of DSD are larger than 0.2, but less than 0.1 in both Figure 3c and Figure 3d. On the other side, the biases of DSD are smaller on datasets with fewer variables and higher on datasets with more variables. For example, the gap between the performance of DSD and the second best approach TVGL is larger when the problem scale is smaller. Therefore, those two counterparts (generalization ability and bias) together guarantee the effectiveness of DSD under various problem scales and train sizes.

6.4.2 On the Effectiveness with Sufficient Data. The experimental setting here is identical to the experimental setting for efficiency study in section 6.3. The results are shown in Figure 4. According to the results, we can see that our proposed method constantly performs the best compared to baselines. The overall performance is better with smaller graph size and gets worse with more challenging problem scales. This results again demonstrated our conclusion on the effectiveness of DSD that it predicts well but generalizes fine with fewer variables, and behaves oppositely with more variables.

6.5 Sensitivity Study

α is the only, albeit essential, hyperparameter of our proposed method. It is used to control the penalties on the estimated precision matrix. In order to study the impact of this hyperparameter, we conduct experiments on datasets with different problem scales. At each scale, we investigate how the choice of hyperparameter affects the prediction error (MSE) by varying α among $[0, 1e-4, 1e-3, 1e-2, 1e-1, 1]$. We obtain the results using 1000 data examples for training. The results are presented in Figure 5. As we can see from the results, when α is large, the prediction error is always large across all problem scales. When α is small, the prediction becomes worse as the problem scale grows due to the overfitting. The best performance is achieved when α is neither too large nor too small. $\alpha = 0.01$ is a relatively stable choice across various problem scales.

7 REAL-WORLD STUDY

Data The advantage of DSD on synthetic datasets with quantitative results was demonstrated in Section 6. In this section, we present an application of DSD on a real-world problem. We explore the inter-relationship between the variation of daily stock price among different companies. The idea is to discover significant connections from the underlying graph where companies are nodes. In particular, we conduct our study on the stock prices of S&P 500 companies from the beginning of 2004 to the end of 2007. We do not use the data afterward to avoid the bias caused by the 2008 global financial crisis[3]. Instead of identifying the dependencies on the stock price, we focus on the analysis of price variation, i.e., the difference between closing price and opening price, since this metric is more informative for the growth of a company.

Precision Estimation To discover the structure from stock variations, we first normalize the data into zero means and then apply DSD on the normalized data to estimate the precision matrix. However, due to the distributability of DSD, the learned precision matrix is very close to but not exactly symmetric as we introduced in Section 5.3. So, we conduct nearest positive semidefinite estimation on the learned precision matrix for post-processing. We first symmetrize the estimated precision matrix by averaging its upper triangular matrix and its lower triangular matrix, and then gradually increase its diagonal entries by a tiny number until it becomes PSD. Usually the matrix is PSD right after finding its nearest symmetric estimation.

Partial Correlation The precision matrix itself can tell the conditional independence between nodes well, but is not very informative for finding the most significant connections. We alternatively use partial correlation as the metric. Partial correlation is used to measures the degree of association between multiple interdependent variables. The partial correlation Q_{ij} between variable i and variable j is defined as:

$$Q_{ij} = \Lambda_{ij} / \sqrt{\Lambda_{ii} \cdot \Lambda_{jj}}$$

Then we assume two companies are strongly positively dependent to each other if their partial correlation is significant.

[3]https://en.wikipedia.org/wiki/Financial_crisis_of_2007-2008

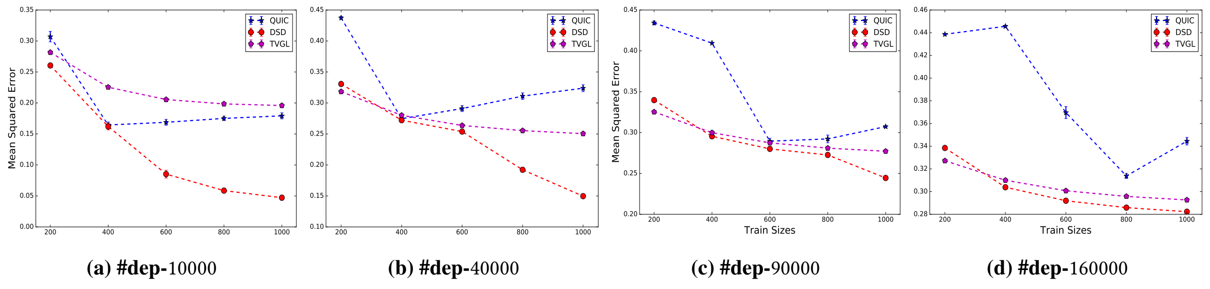

Figure 3: Generalization performance for different problem scales and train sizes

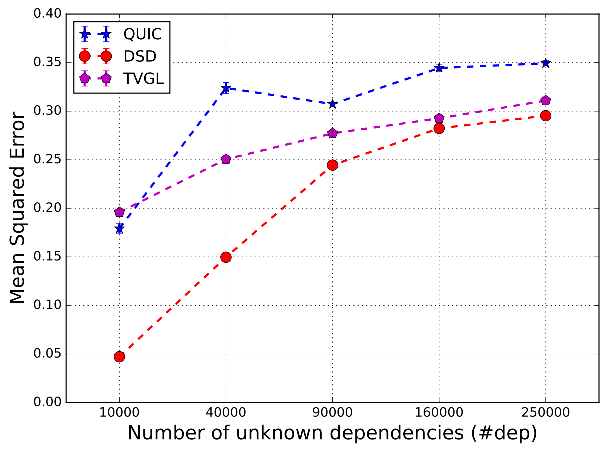

Figure 4: Effectiveness evaluation on different problem scales. $m = 1000$

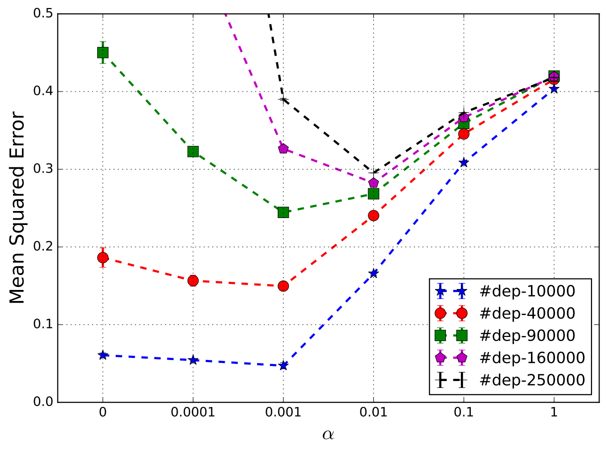

Figure 5: Sensitivity study of α on different problem scales. $m = 1000$

Setting In our experiment, we apply DSD to three different ranges of the dataset: (a). from 2004.01.01 to 2005.12.31; (b). from 2006.01.01 to 2007.12.31; and (c). from 2004.01.01 to 2007.12.31. The discovery with top 20 significant links (high partial correlation) of all date ranges is presented in Figure 6. In order to validate our discovery, we also compare our results with pre-defined primary sectors of companies. The discovery is positive if most of the estimated links are intra-sector links, and negative if most links are inter-sector links. The intra-sector links and inter-sector links are presented via solid edges and dashed edges respectively in the figure. The width of the edge is positively related to the partial correlation. The 8 pre-defined sectors are marked with various colors.

Inter-Sector links & Intra-Sector links From the results, we can observe that most of our revealed connections are intra-sector links across all data ranges. We also note that the most significant connections detected by DSD are within the energy sector, which is marked with red color. It reflects the inflation of oil price caused by the 2000s energy crisis [4]. The oil price roughly became about 5 times more expensive from 2002 to 2008 in the US. We also note that the strongest dependency found between 2004 and 2005 is a inter-sector link between Yahoo and Amazon. It is, however, a false negative since the retail business of Amazon is driven by information technology. 2004-2005 is one of the most prosperous periods for the market for information technology. The market capacities of both Amazon and Yahoo have increased to the peak before 2008. There are also another three visible inter-sector links across all time ranges: Sony and Canon, Home depot and Walmart, and Toyota and Canon. Although these companies own different types of business, they are still related to each other given the proximity of non-business factors. For example, Home Depot can provide downstream service to Walmart customers once they purchase furniture. Sony, Canon, and Toyota are all from Japan, so their performance are affected by the global economy of Japan. All of these fact-consistent discoveries validate the capability of our proposed method again. The alternative approaches also have nice discovery with many intra-sector links. However, as it is difficult to compare the importance of two different intra-sector links, we cannot qualitatively compare with other methods in this study.

8 CONCLUSION AND FUTURE WORK

In this paper, we propose a framework to discover structure from high dimensional data in a distributed manner. Our method is proven to be able to recover the entire precision matrix from multiple sub-tasks given its nature of distributatbility and convexity. The effectiveness and efficiency of the proposed method are demonstrated via extensive experiments on synthetic datasets. The ability to discover the underlying structure is also validated in real-world cases. In our future work, the proposed framework potentially has great space to

[4]https://en.wikipedia.org/wiki/2000s_energy_crisis

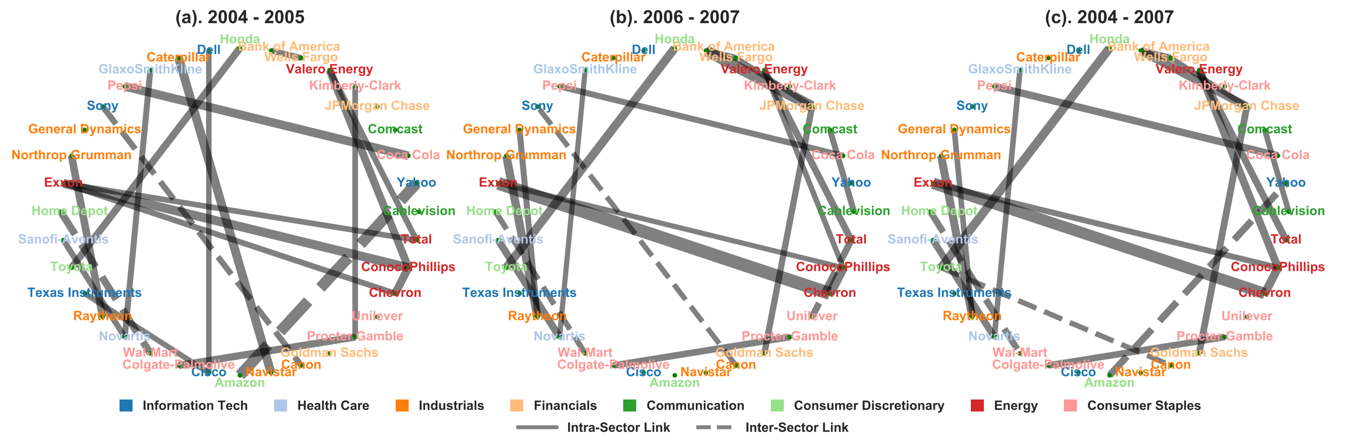

Figure 6: Graph visualization of top 20 links with the largest partial correlation of variation of stock price. (a). from 2004 to 2005 (b). from 2006 to 2007 (c). from 2004 to 2007

be extended for different purposes by adopting various advanced optimization algorithms or protocols for distribution.

ACKNOWLEDGEMENT

This research was supported in part by the NSF grants IIS-1842183, SES-1659998, CNS-1625061 and AFRL Contract No. FA8750-18-C-0041, subcontact 555027-78056.

REFERENCES

[1] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. 2008. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *The Journal of Machine Learning Research* 9 (2008), 485–516.
[2] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning* 3, 1 (2011), 1–122.
[3] Tony Cai, Weidong Liu, and Xi Luo. 2011. A constrained 1 minimization approach to sparse precision matrix estimation. *J. Amer. Statist. Assoc.* 106, 494 (2011), 594–607.
[4] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. ACM, 129–136.
[5] John Duchi, Stephen Gould, and Daphne Koller. 2012. Projected subgradient methods for learning sparse gaussians. *arXiv preprint arXiv:1206.3249* (2012).
[6] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2008. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* 9, 3 (2008), 432–441.
[7] Jean Gallier. 2010. The Schur complement and symmetric positive semidefinite (and definite) matrices. *Technical report, Penn Engineering* (2010).
[8] Yuhong Guo and Suicheng Gu. 2011. Multi-label classification using conditional dependency networks. In *IJCAI Proc. International Joint Conf. on Artificial Intelligence*.
[9] Yuhong Guo and Wei Xue. 2013. Probabilistic Multi-Label Classification with Sparse Feature Learning.. In *IJCAI Proc. International Joint Conf. on Artificial Intelligence*.
[10] David Hallac, Youngsuk Park, Stephen Boyd, and Jure Leskovec. 2017. Network inference via the time-varying graphical lasso. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 205–213.
[11] Chao Han, Mohamed F Ghalwash, and Zoran Obradovic. 2017. Continuous Conditional Dependency Network for Structured Regression.. In *AAAI*. 1962–1968.
[12] David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. 2001. Dependency networks for inference, collaborative filtering, and data visualization. *The Journal of Machine Learning Research* 1 (2001), 49–75.
[13] Cho-Jui Hsieh, Arindam Banerjee, Inderjit S Dhillon, and Pradeep K Ravikumar. 2012. A divide-and-conquer method for sparse inverse covariance estimation. In *Advances in Neural Information Processing Systems*. 2330–2338.
[14] Cho-Jui Hsieh, Inderjit S Dhillon, Pradeep K Ravikumar, and Mátyás A Sustik. 2011. Sparse inverse covariance matrix estimation using quadratic approximation. In *Advances in neural information processing systems*. 2330–2338.
[15] Cho-Jui Hsieh, Mátyás A Sustik, Inderjit S Dhillon, and Pradeep Ravikumar. 2014. QUIC: quadratic approximation for sparse inverse covariance estimation. *The Journal of Machine Learning Research* 15, 1 (2014), 2911–2947.
[16] Cho-Jui Hsieh, Mátyás A Sustik, Inderjit S Dhillon, Pradeep K Ravikumar, and Russell Poldrack. 2013. BIG & QUIC: Sparse inverse covariance estimation for a million variables. In *Advances in neural information processing systems*. 3165–3173.
[17] Xiangnan Kong and Philip S Yu. 2014. Brain network analysis: a data mining perspective. *ACM SIGKDD Explorations Newsletter* 15, 2 (2014), 30–38.
[18] Han Liu and Lie Wang. 2017. Tiger: A tuning-insensitive approach for optimally estimating gaussian graphical models. *Electronic Journal of Statistics* 11, 1 (2017), 241–294.
[19] Nicolai Meinshausen, Peter Bühlmann, et al. 2006. High-dimensional graphs and variable selection with the lasso. *The annals of statistics* 34, 3 (2006), 1436–1462.
[20] Mathew Monfort, Anqi Liu, and Brian D Ziebart. 2015. Intent Prediction and Trajectory Forecasting via Predictive Inverse Linear-Quadratic Regulation.. In *AAAI*. 3672–3678.
[21] Nasser M Nasrabadi. 2007. Pattern recognition and machine learning. *Journal of electronic imaging* 16, 4 (2007), 049901.
[22] Neal Parikh, Stephen Boyd, et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1, 3 (2014), 127–239.
[23] Tao Qin, Tie-Yan Liu, Xu-Dong Zhang, De-Sheng Wang, and Hang Li. 2009. Global ranking using continuous conditional random fields. In *Advances in neural information processing systems*. 1281–1288.
[24] Pradeep Ravikumar, Martin J Wainwright, John D Lafferty, et al. 2010. High-dimensional Ising model selection using 1-regularized logistic regression. *The Annals of Statistics* 38, 3 (2010), 1287–1319.
[25] Havard Rue and Leonhard Held. 2005. *Gaussian Markov random fields: theory and applications*. CRC press.
[26] Katya Scheinberg, Shiqian Ma, and Donald Goldfarb. 2010. Sparse inverse covariance selection via alternating linearization methods. In *Advances in neural information processing systems*. 2101–2109.
[27] Katya Scheinberg and Irina Rish. 2010. Learning sparse Gaussian Markov networks using a greedy coordinate ascent approach. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 196–212.
[28] Joshua M Stuart, Eran Segal, Daphne Koller, and Stuart K Kim. 2003. A gene-coexpression network for global discovery of conserved genetic modules. *science* 302, 5643 (2003), 249–255.
[29] Eunho Yang, Pradeep Ravikumar, Genevera I Allen, and Zhandong Liu. 2015. Graphical models via univariate exponential family distributions. *The Journal of Machine Learning Research* 16, 1 (2015), 3813–3847.
[30] Ming Yuan and Yi Lin. 2007. Model selection and estimation in the Gaussian graphical model. *Biometrika* 94, 1 (2007), 19–35.
[31] Xiao-Tong Yuan and Tong Zhang. 2014. Partial gaussian graphical model estimation. *IEEE Transactions on Information Theory* 60, 3 (2014), 1673–1687.