Journal of Machine Learning Research 21 (2020) 1-54 Submitted 7/18; Revised 2/20; Published 6/20

# Fast Bayesian Inference of Sparse Networks with Automatic Sparsity Determination

**Hang Yu** HYU1@E.NTU.EDU.SG
**Songwei Wu** WUSO0002@E.NTU.EDU.SG
*School of Electrical and Electronic Engineering*
*Nanyang Technological University*
*50 Nanyang Avenue, 639798 Singapore*

**Luyin Xin** XINL0002@E.NTU.EDU.SG
*School of Physical and Mathematical Sciences*
*Nanyang Technological University*
*50 Nanyang Avenue, 639798 Singapore*

**Justin Dauwels** JDAUWELS@NTU.EDU.SG
*School of Electrical and Electronic Engineering*
*Nanyang Technological University*
*50 Nanyang Avenue, 639798 Singapore*

**Editor:** Peter Spirtes

## Abstract

Structure learning of Gaussian graphical models typically involves careful tuning of penalty parameters, which balance the tradeoff between data fidelity and graph sparsity. Unfortunately, this tuning is often a “black art” requiring expert experience or brute-force search. It is therefore tempting to develop tuning-free algorithms that can determine the sparsity of the graph adaptively from the observed data in an automatic fashion. In this paper, we propose a novel approach, named BISN (Bayesian inference of Sparse Networks), for automatic Gaussian graphical model selection. Specifically, we regard the off-diagonal entries in the precision matrix as random variables and impose sparse-promoting horseshoe priors on them, resulting in automatic sparsity determination. With the help of stochastic gradients, an efficient variational Bayes algorithm is derived to learn the model. We further propose a decaying recursive stochastic gradient (DRSG) method to reduce the variance of the stochastic gradients and to accelerate the convergence. Our theoretical analysis shows that the time complexity of BISN scales only *quadratically* with the dimension, whereas the theoretical time complexity of the state-of-the-art methods for automatic graphical model selection is typically a third-order function of the dimension. Furthermore, numerical results show that BISN can achieve comparable or better performance than the state-of-the-art methods in terms of structure recovery, and yet its computational time is several orders of magnitude shorter, especially for large dimensions.

**Keywords:** Gaussian Graphical Models, Structure Learning, Tuning Free, Time Complexity, Variational Bayes, Variance Reduction, Decaying Recursive Stochastic Gradient (DRSG).

## 1. Introduction

Undirected graphical models can compactly encode high-dimensional data and help to interpret the data. As an example, inferring the gene regulatory networks (i.e., which genes may have correlation and which are expressed independently) is currently en vogue, since it is an effective strategy to understand how genes interact in a biological process and to further detect novel disease mechanisms (Su et al., 2018; Jia and Liang, 2018; Zhao and Duan, 2019). Now suppose one has a collection of thousands of genes and a graphical model is used to represent the network where the nodes denote the genes and there exists an edge between two nodes if the corresponding two genes are conditionally dependent. According to the principle of parsimony, we should select the simplest (i.e., sparsest) graphical model that can adequately describe the data. Such a graphical model provides an efficient representation of the gene regulatory network, thus allowing the biologists to focus on a limited number of gene pairs and greatly reducing the time and effort for further analysis. Apart from gene networks, undirected graphical models have traced their origins to many different fields and have been applied in a wide variety of settings, such as social network analysis (Farasat et al., 2015), financial system risk modeling (Hashem and Giudici, 2016; Bianchi et al., 2019), and brain connectivity analysis (Ortiz et al., 2015; Belilovsky et al., 2016). In practice, nevertheless, the structure of the undirected graphical model is typically unknown, and therefore, we aim to learn the graph structure from the data.

Particularly for Gaussian graphical models, the graph structure is characterized by the precision (inverse covariance) matrix: a zero off-diagonal entry in the precision matrix denotes the conditional independence of two variables as well as the absence of the corresponding edge in the graphical model. Hence, when variables are Gaussian distributed in an undirected graphical model, the objective is to seek a sparse precision matrix that can best describe the data. In this paper, we focus on the structure learning of Gaussian graphical models with automatic sparsity determination. In the following review, we divide the large body of literature on graphical model selection into three categories, including tuning-sensitive, tuning-insensitive, and tuning-free methods. For the first class, we also review the corresponding methods for selecting the tuning parameter.

Algorithms in the first category typically employ frequentist methods that maximize the likelihood (Friedman et al., 2008; Banerjee et al., 2008; Duchi et al., 2008; Scheinberg et al., 2010; Rolfs et al., 2012; Hsieh et al., 2014, 2013; Treister and Turek, 2014; Tarzanagh and Michailidis, 2018; Zhang et al., 2018b; Bollhöfer et al., 2019) or pseudo-likelihood (Meinshausen and Bühlmann, 2006) of the precision matrix with an $\ell_1$ norm penalty on the matrix. Given the penalty parameter (a.k.a. regularization parameter) in front of the $\ell_1$ norm, the resulting problem is convex and can be solved via optimization algorithms. Considerable efforts have been undertaken to increase the scalability of such algorithms, such as BIG&QUIC (Hsieh et al., 2013) and BCDIC (Treister and Turek, 2014); at their heart lies the insight that only a small proportion of elements in the precision matrix needs to be updated to guarantee convergence under certain conditions. More precisely, it has been shown in Hsieh et al. (2014) that we only need to update non-zero elements and elements whose gradients are larger than the penalty parameter in each iteration. Indeed, by leveraging the sparsity of the precision matrix, the time complexity of the computational bottleneck of BIG&QUIC and BCDIC is $\mathcal{O}(pm)$, where $p$ is the dimension and $m$ is the average number of non-zero elements in the precision matrix as the algorithm proceeds. As a result, by initializing the precision matrix as a diagonal matrix and choosing a large penalty parameter, the precision matrix can be kept sparse in all iterations, therefore, the resulting algorithms are applicable to problems with one million variables (Hsieh et al., 2013; Treister and Turek, 2014). Nevertheless, there remains a severe drawback: the penalty parameter is typically unknown in real-world applications and it determines the sparsity of the graph estimates. To find a proper value, we can only resort to brute-force search methods, such as cross validation, Akaike information criterion (AIC), Bayesian information criterion (BIC), and stability selection (Meinshausen and Bühlmann, 2010; Liu et al., 2010; Li et al., 2013). To make matters worse, we have to consider small candidates of the penalty parameter. As pointed out in Liu et al. (2010), one popular criterion to choose the penalty parameter is to find the least amount of penalization for which the graph estimates are suitably stable across samples. The rationale behind is to select a graph that is slightly denser than the ground truth, since in practice the false positives can be removed in further analysis whereas false negatives can no longer be recovered as they are buried by the sizable number of true negatives. In other words, it is essential as well as beneficial to test relatively small candidates when seeking a proper value of the penalty parameter, since they can guard against false negatives. Such small candidates, however, can significantly increase $m$ and so incur a prohibitive computational cost for BIG&QUIC (Hsieh et al., 2013) and BCDIC (Treister and Turek, 2014), as shown in Treister et al. (2016) and in the results section of this paper. Consequently, when coupling BIG&QUIC and BCDIC with the aforementioned regularization selection methods, they will not be able to handle one million variables. Theoretically, the worst-case time complexity increases cubically with the dimension, thus jeopardizing their practicality in large-scale problems. Apart from the frequentist methods, we notice that recently decision theory has been employed for structure learning in Gaussian graphical models, where the conditional dependence for every pair of variables in the graph is checked via different hypothesis testings (Lafit et al., 2018; Williams et al., 2018; Li and Maathuis, 2019; Bernal et al., 2019; Leday and Richardson, 2019; Tatikonda et al., 2019). Unfortunately, these methods also introduce some tuning parameters, such as the threshold of edge inclusion in Lafit et al. (2018); Leday and Richardson (2019), and the threshold of predictive accuracy and the confidence level in Williams et al. (2018). Hence, they also suffer from the issue of selecting the tuning parameters.

To mitigate the problems of tuning-sensitive methods, a more satisfying tuning-insensitive approach (TIGER) is proposed in Liu and Wang (2017), where a square root lasso problem is formulated to select the neighbors for each variable individually. It has been proven in Liu and Wang (2017) that this method is asymptotically tuning-free but requires a little effort to tune the penalty parameter among three fixed candidates in the practical finite sample settings. Unfortunately, TIGER suffers from three pitfalls. First, the neighborhood selection procedure is a pseudo-likelihood approximation to the original problem. Although it typically produces a more accurate estimate of the precision matrix, it does not fit the data well as shown in our numerical experiments, since the true likelihood of the precision matrix is not maximized. Second, the estimated precision matrix is not guaranteed to be positive semi-definite. Third, the time complexity is $\mathcal{O}(\min(n,p)p^2)$, where $n$ is the sample size. Typically, we assume $n$ increases linearly with $p$, and thus, the time complexity is still $\mathcal{O}(p^3)$.

Table 1: Comparison between different methods for automatic Gaussian graphical model selection.

| Methods | Tuning-Sensitive | Tuning-Insensitive | Tuning-free | BISN |
|---|---|---|---|---|
| Brute-force search of penalty parameters | Yes | Yes, among only three candidates | No | No |
| Guarantee of positive semi-definiteness | Yes | No | Yes | Yes |
| Scalability w.r.t. dimension $p$ | $\mathcal{O}(p^3)$ | $\mathcal{O}(\min(n, p)p^2)$ | $\mathcal{O}(p^3)$ | $\mathcal{O}(p^2)$ |

The third class of approaches involves Bayesian methods. They regard the penalty parameters as random variables and infer their posterior distributions along with that of the precision matrix adaptively from the data. Such methods garner all the benefits from the Bayesian paradigm, successfully avoiding the complicated tuning problem while considering the uncertainty associated with parameter estimation. Different priors have been explored, including the G-Wishart prior (Dobra et al., 2011; Mohammadi and Wit, 2015), the Laplace prior (Wang, 2012), the spike and slab prior (Wang, 2015), and the horse-shoe prior (Li et al., 2019). Monte-Carlo Markov Chain (MCMC) algorithms have been developed to learn the Bayesian model (Dobra et al., 2011; Mohammadi and Wit, 2015; Wang, 2012, 2015; Li et al., 2019). A key caveat to applying these approaches, however, is that the MCMC algorithms are quite time-consuming, such that they can only scale up to tens or the lower hundreds of variables. As a remedy, variational Bayes techniques have been proposed based on Student $t$-priors (Marlin and Murphy, 2009; Yu and Dauwels, 2015). To ensure the positive semi-definiteness of the estimated precision matrix, Dirac delta functions are used as variational distributions, effectively reducing the problem to point estimation. Unfortunately, such point estimates can be regarded as solving a marginal MAP problem using variational Bayes, and they are quite sensitive to local maxima as pointed out in Liu and Ihler (2013). To alleviate this issue, Wishart distributions are leveraged as the variational distributions in (Yu et al., 2019). Another drawback associated with all these Bayesian methods is that their time complexity also scales cubically with the dimension.

In this paper, we propose a novel approach named BISN (Bayesian Inference of Sparse Networks) for Gaussian graphical model selection. To the best of our knowledge, we are among the first to develop a *tuning-free* algorithm whose theoretical time complexity is only *quadratic* in the number of variables (i.e., *linear* in the number of elements in the precision matrix). To move forward to this goal, we consider the LDL decomposition of the precision matrix, and derive stochastic variational inference algorithms (Khan et al., 2015; Khan and Lin, 2017) from this decomposition. Specifically, we assume that the off-diagonal entries of the precision matrix follow horseshoe priors (Carvalho et al., 2009), which are commonly used for sparse Bayesian learning (Carvalho et al., 2010). We then approximate the exact posterior distribution of elements in the LDL decomposition by tractable variational distributions. By constraining the diagonal entries of $D$ to be non-negative in the variational distributions, the resulting estimated distribution of the precision matrix is guaranteed to be positive semi-definite. We further derive a stochastic proximal gradient algorithm (Khan et al., 2015; Khan and Lin, 2017) to minimize the KL divergence between the variational distribution and the exact posterior. Since it is computationally cheap to evaluate the stochastic gradients, the computational complexity of BISN per iteration is only $\mathcal{O}(p^2)$ given $p$ variables. A novel decaying recursive stochastic gradient (DRSG) method is proposed to reduce the variance of the stochastic gradient and it is proven that the convergence rate of BISN is independent of the dimension. As a result, the time complexity only scales quadratically with the dimension. In our numerical experiments, the computational time of BISN is several orders of magnitude smaller than that of the state-of-art tuning-sensitive and tuning-insensitive methods (Rolfs et al., 2012; Hsieh et al., 2013; Liu and Wang, 2017), yet it achieves comparable or better performance in terms of structure recovery. To summarize, our main contributions are:

1. We propose BISN for Gaussian graphical model selection, a tuning-free algorithm whose computational cost scales quadratically with the dimension. To highlight the appeal of BISN, we further compare different methods for automatic Gaussian graphical model selection in Table 1.

2. We propose to infer the LDL decomposition rather than the precision matrix as in existing works. This reparameterization highly simplifies the evidence lower bound (ELBO) in the variational Bayes framework and enables the derivation of the low-complexity stochastic gradient algorithm to optimize the ELBO (cf. Section 3.5).

3. We propose a novel decaying recursive stochastic gradient (DRSG) approach to speed up the convergence of the stochastic gradient algorithm. The convergence rate of the original stochastic gradient algorithm is $\mathcal{O}(1/\epsilon^2)$, whereas that of DRSG is $\mathcal{O}(1/\epsilon^{3/2})$ with a fixed step size. The proposed DRSG is applicable to the general finite sum optimization problems in which the objective function can be decomposed as the sum of one smooth term that can be nonconvex and one convex term that can be nonsmooth.

4. We apply BISN to analyze stock, gene, and fMRI data and provide some insights into the data based on the inferred network.

The remainder of the paper is structured as follows. In Section 2, we introduce the Bayesian formulation of the proposed model. We then derive the evidence lower bound (ELBO) and the stochastic proximal gradient algorithm in Section 3. We further propose methods to reduce the variance and speed up the convergence in Section 4. Theoretical results for convergence rate and run time guarantee are presented in Section 5. We validate the proposed approach through synthetic and real data in Section 6. Finally, in Section 7, we provide concluding remarks.

## 2. Bayesian Formulation of Gaussian Graphical Models

An undirected graphical model can be defined as a multivariate probability distribution $p(\boldsymbol{x})$ that factorizes according to a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ which consists of nodes $\mathcal{V}$ and edges $\mathcal{E}$. Concretely, each node $j \in \mathcal{V}$ is associated with a random variable $x_j$. An edge $(j,k) \in \mathcal{E}$ is absent if and only if the corresponding two variables $x_j$ and $x_k$ are conditionally independent:

$$p(x_j, x_k | x_{\mathcal{V}|j,k}) = p(x_j | x_{\mathcal{V}|j,k}) p(x_k | x_{\mathcal{V}|j,k}), \tag{1}$$

where $\mathcal{V}|j,k$ denotes all the nodes except $j$ and $k$.

If the random variables $\boldsymbol{x} = [x_1, \cdots, x_p]^T$ corresponding to the nodes on the graph are jointly Gaussian, then the graphical model is called a Gaussian graphical model. Let $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with mean vector $\boldsymbol{\mu}$ and positive-definite covariance matrix $\Sigma$. Since $\boldsymbol{x}$ constitutes a Gaussian graphical model, the precision matrix (the inverse covariance) $K = \Sigma^{-1}$ is sparse with respect to the graph $\mathcal{G}$, i.e., $[K]_{j,k} \neq 0$ if and only if the edge $(j,k) \in \mathcal{E}$. The Gaussian graphical model can be written in an equivalent information form $\mathcal{N}(K^{-1}\boldsymbol{h}, K^{-1})$ with a precision matrix $K$ and a potential vector $\boldsymbol{h} = \Sigma^{-1}\boldsymbol{\mu}$. The corresponding probability density function (PDF) can be written as:

$$p(\boldsymbol{x}) \propto \det(K)^{\frac{1}{2}} \exp\Big(-\frac{1}{2}\boldsymbol{x}^T K \boldsymbol{x} + \boldsymbol{h}^T\boldsymbol{x}\Big). \tag{2}$$

Without loss of generality, we assume that $\boldsymbol{\mu} = 0$ in our model, and so $\boldsymbol{h} = 0$. As such,

$$p(\boldsymbol{x}) \propto \det(K)^{\frac{1}{2}} \exp\Big(-\frac{1}{2}\boldsymbol{x}^T K \boldsymbol{x}\Big). \tag{3}$$

As mentioned in the introduction, we consider here the LDL decomposition of the precision matrix $K$. Since $K$ is symmetric, it can be decomposed as $K = LDL^T$, where $L$ is a lower triangular matrix with ones on the diagonal and $D$ is a diagonal matrix. The above expression (3) can therefore be equivalently written as:

$$p(\boldsymbol{x}|L,D) \propto \prod_{j=1}^{p} D_{jj} \exp\Big(-\frac{1}{2}\boldsymbol{x}^T LDL^T \boldsymbol{x}\Big), \tag{4}$$

where $D_{jj}$ is the $j$-th diagonal elements of $D$.

So as to obtain a sparse precision matrix $K$, we impose horseshoe priors (Carvalho et al., 2009, 2010) on the off-diagonal entries $K_{jk}$ of the precision matrix, which can be interpreted as a scale mixture of Gaussians:

$$p(K_{jk}|\sigma_{jk}, \nu) = \mathcal{N}(0, \nu^2\sigma_{jk}^2), \tag{5}$$

$$p(\sigma_{jk}) = C^+(0,1), \tag{6}$$

where $C^+(0,1)$ is a standard half-Cauchy distribution on the positive reals, and $\nu$ and $\sigma_{jk}$ are respectively the global and local shrinkage parameters. The global shrinkage parameter $\nu$ shrinks all the entries $K_{jk}$ to zero, whereas the heavy-tailed half-Cauchy priors for the local shrinkage parameters $\sigma_{jk}$ allow some $K_{jk}$ to escape from the shrinkage. Therefore, the resulting precision matrix $K$ would be sparse. In comparison with other shrinkage priors, such as Laplacian and Student-$t$ priors, the horseshoe priors are more robust when dealing with unknown sparsity and large non-zero elements, as demonstrated in Carvalho et al. (2009) and Bhadra et al. (2017). To facilitate the variational inference, we employ the parameterization of the horseshoe prior in Neville et al. (2014):

$$p(K_{jk}|\lambda_{jk}, \omega) = \mathcal{N}\big(0, (\omega\lambda_{jk})^{-1}\big) \propto \sqrt{\omega\lambda_{jk}} \exp\Big(-\frac{1}{2}\omega\lambda_{jk}K_{jk}^2\Big), \tag{7}$$

$$p(\lambda_{jk}) = \frac{1}{\pi}\lambda_{jk}^{-\frac{1}{2}}\big(\lambda_{jk}+1\big)^{-1}, \quad \forall \lambda_{jk} > 0, \tag{8}$$

where $\omega = 1/\nu^2$, $\lambda_{jk} = 1/\sigma_{jk}^2$, and $p(\lambda_{jk})$ can be regarded as a Beta prime distribution or a compound Gamma distribution. We further impose a non-informative Jeffreys prior on the global shrinkage parameter $\omega$, that is, $p(\omega) \propto 1/\omega^1$.

On the other hand, for the diagonal elements $K_{jj}$, we assume that $p(K_{jj}) \propto 1$. Taken together, the overall prior on $K$ can be factorized as:

$$p(K|\boldsymbol{\lambda}, \omega) = \prod_{j=1}^{p} \prod_{k=j+1}^{p} p(K_{jk}|\lambda_{jk}, \omega). \tag{9}$$

Since we focus on $LDL^T$ instead of $K$, the above prior distribution can be expressed as follows according to the change-of-variable formula (Kucukelbir et al., 2015):

$$p(L, D|\boldsymbol{\lambda}, \omega) = |\det(J)| \prod_{j=1}^{p} \prod_{k=j+1}^{p} p(L_{j,:}DL_{k,:}^T|\lambda_{jk}, \omega), \tag{10}$$

where $J$ is the Jacobian matrix, and $L_{j,:}$ denotes row $j$ in $L$. Fortunately, $J$ can be permuted as an upper triangular matrix and the absolute value of its determinant has a closed-form expression:

$$|\det(J)| = \prod_{j=1}^{p} D_{jj}^{p-j}. \tag{11}$$

The derivation is outlined in Appendix A. We notice that the previous studies on Bayesian graphical model selection (Dobra et al., 2011; Wang, 2012; Mohammadi and Wit, 2015; Wang, 2015) revolve around intractable priors without closed-form log-partition functions, due to the truncation of sparse-promoting priors on the positive semi-definite cone. As a consequence, only MCMC algorithms can be applied, which are computationally burdensome. By contrast, the proposed prior facilitates the derivation of efficient variational Bayes algorithm. Furthermore, although the above prior does not impose any constraints on the positive definiteness of matrix $K$, it is straightforward to guarantee that $K$ resulting from the variational distribution is positive semi-definite by constraining $D_{jj} \geq 0$ for all $j$ in the variational distribution. In addition, we emphasize that we only constrain $K$ to be sparse but do not impose any constraints on $L$. Note that there are a handful of methods in the literature that enforce sparse constraints directly on $L$ (Smith and Kohn, 2002; Huang et al., 2006), and they are equivalent to learning a sparse acyclic directed graph on $\boldsymbol{x}$ with a predefined order, as pointed out in Marlin and Murphy (2009). Our method, on the other hand, does not suffer from this problem. We obtain the prior distribution of $L$ and $D$ (10) simply from the reparameterization of the distribution of $K$ (9). Depending on the ordering of the variables in $\boldsymbol{x}$, the distribution of $L$ and $D$ in (10) will be different. However, the prior distribution of $K$ in (9) remains the same, regardless of the ordering of the variables.

[1] When prior information is available, such as the estimated number of edges in the graph, informative priors can also be imposed on $\omega$, such as the prior proposed in Piironen et al. (2017).

Finally, given $n$ samples of $\boldsymbol{x}$, the resulting Bayesian model can be factorized as:

$$\begin{aligned} p(\boldsymbol{x}^{\{1:n\}}, L, D, \boldsymbol{\lambda}, \omega) =& p(\boldsymbol{x}^{\{1:n\}}|L, D) p(L, D|\boldsymbol{\lambda}, \omega) p(\boldsymbol{\lambda}) p(\omega) && (12) \\ =& \prod_{i=1}^{n} p(\boldsymbol{x}^{\{i\}}|L, D) |\det(J)| \prod_{j=1}^{p} \prod_{k=j+1}^{p} \Big[ p(L_{j,:} D L_{k,:}^T | \lambda_{jk}, \omega) p(\lambda_{jk}) \Big] \\ & \cdot p(\omega). && (13) \end{aligned}$$

## 3. Proximal-Gradient Stochastic Variational Inference

In this section, we first define the variational Bayes problem to be solved. Next, we introduce the KL (Kullback-Leibler) proximal gradient algorithm and further apply it to solve our problem. Additionally, we discuss how to reduce the computational complexity per iteration from $\mathcal{O}(p^3)$ to $\mathcal{O}(p^2)$ via unbiased stochastic approximation.

### 3.1. Variational Bayes Inference

Our objective is to compute the posterior $p(L, D, \boldsymbol{\lambda}, \omega | \boldsymbol{x}^{\{1:n\}})$. Unfortunately, it is intractable to obtain the posterior in a closed form. Instead, we follow the variational Bayes framework to approximate the intractable true posterior with a tractable variational distribution $q(L, D, \boldsymbol{\lambda}, \omega)$ by minimizing the KL divergence $\mathbb{KL}[q|p] = \int q \log(q/p)$. Equivalently, we aim to maximize the evidence lower bound (ELBO) $\mathcal{L}$ of the data likelihood, that is,

$$\begin{aligned} \log p(\boldsymbol{x}^{\{1:n\}}) =& \log \int q(L, D, \boldsymbol{\lambda}, \omega) \frac{p(\boldsymbol{x}^{\{1:n\}}, L, D, \boldsymbol{\lambda}, \omega)}{q(L, D, \boldsymbol{\lambda}, \omega)} dL dD d\boldsymbol{\lambda} d\omega \\ \geq& \mathbb{E}_q \big[ \log p(\boldsymbol{x}^{\{1:n\}}, L, D, \boldsymbol{\lambda}, \omega) - \log q(L, D, \boldsymbol{\lambda}, \omega) \big] = \mathcal{L}, \qquad (14) \end{aligned}$$

where $\mathbb{E}_q[f(\cdot)]$ denotes the expectation of the function $f(\cdot)$ over the $q$ distribution. Here, we apply the mean-field approximation (Beal et al., 2006) and factorize the variational distribution as:

$$q(L, D, \boldsymbol{\lambda}, \omega) = \Big\{ \prod_j q(D_{jj}) \Big[ \prod_{k>j} q(L_{jk}) q(\lambda_{jk}) \Big] \Big\} q(\omega), \qquad (15)$$

where all factors are chosen from the minimal exponential family (Khan and Lin, 2017) and they can be parameterized by the natural parameters[2] as:

$$\begin{aligned} q(D_{jj}; \alpha_j, \beta_j) &= \mathrm{Ga}(\alpha_j, \beta_j) \propto D_{jj}^{(\alpha_j - 1)} \exp(-\beta_j D_{jj}), && (16) \\ q(L_{jk}; h_{jk}, \zeta_{jk}) &= \mathcal{N}\big(\zeta_{jk}^{-1} h_{jk}, \zeta_{jk}^{-1}\big) \propto \sqrt{\zeta_{jk}} \exp\Big( -\frac{1}{2} \zeta_{jk} L_{jk}^2 + h_{jk} L_{jk} \Big), && (17) \\ q(\omega; a, b) &= \mathrm{Ga}(a, b) \propto \omega^{(a-1)} \exp(-b\omega), && (18) \\ q(\lambda_{jk}; d_{jk}) &= \frac{1}{E_1(d_{jk})} \big(\lambda_{jk} + 1\big)^{-1} \exp\Big( -d_{jk} \big(\lambda_{jk} + 1\big) \Big). && (19) \end{aligned}$$

[2] We actually use the linear transformations of the natural parameters for Gamma and Gaussian distributions, as they are commonly used in practice.

In the above expressions, $\mathrm{Ga}(a, b)$ denotes a Gamma distribution with shape parameter $a$ and rate parameter $b$. Since $q(D_{jj})$ is defined on $(0, \infty)$, the precision matrix resulting from the variational inference is guaranteed to be positive definite. Additionally, the expression of $q(\lambda_{jk})$ in (19) is derived in Appendix B.1.3 in Neville et al. (2014), where $E_1(x) = \int_x^\infty \exp(-t)/t dt$ represents the exponential integral function. Note that $q(\lambda_{jk})$ belongs to the minimal exponential family and there is a one-to-one mapping between the natural parameter $b_{jk}$ and the mean parameter $\mathbb{E}_q[\lambda_{jk}]$[3]:

$$\mathbb{E}_q[\lambda_{jk}] = \frac{1}{d_{jk} \exp(d_{jk}) E_1(d_{jk})} - 1. \tag{20}$$

### 3.2. KL Proximal Gradient

The traditional mean-field variational Bayes (MFVB) approach (Bishop, 2006) and its stochastic version (Hoffman et al., 2013) take full advantage of the geometry of the posterior by using natural gradients, resulting in closed-form update rules and faster convergence than standard gradients. However, they are only applicable to conditionally-conjugate models. In the proposed model (13), the likelihood and prior of $\lambda_{jk}$ and $\omega$ are conjugate, whereas the other factors are not conjugate. To tackle the non-conjugacy, we instead employ the KL proximal gradient methods (Khan et al., 2015; Khan and Lin, 2017) to maximize the lower bound. By using the KL divergence as the proximal term, the resulting algorithm also accommodates the geometry of the posterior and hence enjoys the advantages of MFVB (Khan et al., 2016). In particular, this method reduces to using natural gradients for conjugate pairs of likelihood and prior (Khan and Lin, 2017).

Specifically, suppose that $\boldsymbol{x}$ and $\boldsymbol{z}$ represent the observed and unobserved variables in a Bayesian model respectively. Under the framework of variational Bayes, we intend to find a variational distribution $q(\boldsymbol{z}|\boldsymbol{\theta})$ to approximate the true posterior $p(\boldsymbol{z}|\boldsymbol{x})$ by maximizing the ELBO $\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q(\boldsymbol{z}|\boldsymbol{\theta})]$. Typically, we select $q(\boldsymbol{z}|\boldsymbol{\theta})$ from the minimal exponential family such that there is a one-to-one mapping between the natural parameters $\boldsymbol{\theta}$ and the mean parameters $\boldsymbol{\mu} = \mathbb{E}_q[\phi(\boldsymbol{z})]$ (Khan and Lin, 2017), where $\phi(\boldsymbol{z})$ denotes the sufficient statistics. As a result, the optimization of the ELBO can also be expressed as a maximization over the mean parameters $\boldsymbol{\mu}$. The reparameterized ELBO is represented by $\mathcal{L}(\boldsymbol{\mu})$. The KL proximal gradient method (Khan et al., 2015; Khan and Lin, 2017) then performs the following step until convergence (Khan et al., 2015; Khan and Lin, 2017; Khan et al., 2016):

$$\boldsymbol{\mu}^{(\kappa+1)} = \underset{\boldsymbol{\mu}}{\mathrm{argmin}} \; -\boldsymbol{\mu}^T \nabla_{\boldsymbol{\mu}} \mathcal{L}(\boldsymbol{\mu}^{(\kappa)}) + \frac{1}{\rho^{(\kappa)}} \mathbb{KL}[q(\boldsymbol{z}|\boldsymbol{\mu})|q(\boldsymbol{z}|\mu^{(\kappa)})], \tag{21}$$

where $\rho^{(\kappa)}$ can be regarded as the step size. The above step can also be interpreted as a mirror descent update, and it has been proven in Khan and Lin (2017) that (21) is equivalent to updating the natural parameters as:

$$\boldsymbol{\theta}^{(\kappa+1)} = (1 - \eta^{(\kappa)}) \boldsymbol{\theta}^{(\kappa)} + \eta^{(\kappa)} \nabla_{\boldsymbol{\mu}} \mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})]|_{\boldsymbol{\mu}=\boldsymbol{\mu}^{(\kappa)}}, \tag{22}$$

[3] The product $\exp(d_{jk}) E_1(d_{jk})$ can be calculated efficiently using Lentz's Algorithm (Lentz, 1976; Neville et al., 2014) if $b_{jk} > 10$, while reliable evaluation of both $E_1(d_{jk})$ and $\exp(d_{jk})$ can be achieved by calling build-in functions in R, MATLAB, and C/C++ standard library if $b_{jk} < 10$.

where $\eta^{(\kappa)} = \rho^{(\kappa)}/(1+\rho^{(\kappa)})$. Interestingly, although we propose an update in the mean-parameter space (21), the actual updates can be performed in the natural-parameter space more succinctly (22). In the sequel, we use $\mathcal{L}_1$ to denote $\mathbb{E}_q[\log p(\boldsymbol{x}, \boldsymbol{z})]$, as it is also the first term of the ELBO $\mathcal{L}$.

### 3.3. KL Proximal Gradient for BISN

Owing to the closed-form prior for sparse matrices and the mean-field approximation, $\mathcal{L}_1$ of BISN has an analytical form (see Appendix B for the derivation):

$$\begin{aligned}\mathcal{L}_1 =& \sum_{j=1}^{p}\Big(\frac{n}{2}+p-j\Big)\langle \log D_{jj}\rangle - \frac{n}{2}\operatorname{tr}(M_L M_D M_L^T S) - \frac{n}{2}\operatorname{diag}(S)^T V_L M_D \mathbf{1} \\ & - \frac{1}{4}\operatorname{tr}\Big\{\Lambda\big[(M_L\circ M_L)(M_D\circ M_D+V_D)V_L^T + V_L(M_D\circ M_D+V_D)(M_L\circ M_L)^T \\ & + V_L(M_D\circ M_D+V_D)V_L^T + (M_L\circ M_L)V_D(M_L\circ M_L)^T + (M_L M_D M_L^T)\circ(M_L M_D M_L^T)\big]\Big\} \\ & - \sum_{j=1}^{p}\sum_{k=j+1}^{p}\langle\log(\lambda_{jk}+1)\rangle + \left[\frac{p(p-1)}{4}-1\right]\langle\log\omega\rangle + c, \qquad (23)\end{aligned}$$

where $\langle f(\cdot)\rangle = \mathbb{E}_q[f(\cdot)]$ denotes the expectation of the function $f(\cdot)$ over the $q$ distribution, $c$ summarizes all irrelevant constants, $\circ$ denotes Hadamard product, $\mathbf{1}$ is a column vector of all ones, $S$ is the empirical covariance of the observed data $\boldsymbol{x}^{\{1:n\}}$, $M_L$, $V_L$, $M_D$, and $V_D$ denote the matrices containing the element-wise mean and variance of $L_{jk}$ and $D_{jj}$ respectively, and $\Lambda$ is an off-diagonal matrix with $\Lambda_{jk} = \langle\omega\rangle\langle\lambda_{jk}\rangle$.

Following the framework of KL proximal gradient (Khan and Lin, 2017), the update rules for the natural parameters of the variational distributions (16)-(19) are listed below:

$$h_{jk}^{(\kappa+1)} = (1-\eta^{(\kappa)})h_{jk}^{(\kappa)} + \eta^{(\kappa)}\bigg(\frac{\partial\mathcal{L}_1}{\partial M_{Ljk}} - 2M_{Ljk}\frac{\partial\mathcal{L}_1}{\partial V_{Ljk}}\bigg), \qquad (24)$$

$$\zeta_{jk}^{(\kappa+1)} = (1-\eta^{(\kappa)})\zeta_{jk}^{(\kappa)} - 2\eta^{(\kappa)}\frac{\partial\mathcal{L}_1}{\partial V_{Ljk}}, \qquad (25)$$

$$\alpha_j^{(\kappa+1)} = (1-\eta^{(\kappa)})\alpha_j^{(\kappa)} + \eta^{(\kappa)}\bigg\{\frac{n}{2}+p-j+1-\frac{\alpha_j^{(\kappa)}}{{\beta_j^{(\kappa)}}^2\big[\alpha_j^{(\kappa)}\psi'(\alpha_j^{(\kappa)})-1\big]}\frac{\partial\mathcal{L}_1}{\partial V_{Djj}}\bigg\}, \qquad (26)$$

$$\beta_j^{(\kappa+1)} = (1-\eta^{(\kappa)})\beta_j^{(\kappa)} + \eta^{(\kappa)}\bigg\{\frac{\mathcal{L}_1}{\partial M_{Djj}} + \frac{1}{\beta_j^{(\kappa)}}\bigg[1+\frac{\alpha_j^{(\kappa)}\psi'(\alpha_j^{(\kappa)})}{\alpha_j^{(\kappa)}\psi'(\alpha_j^{(\kappa)})-1}\bigg]\frac{\partial\mathcal{L}_1}{\partial V_{Djj}}\bigg\}, \qquad (27)$$

$$a = \frac{p(p-1)}{4}, \qquad (28)$$

$$b^{(\kappa+1)} = (1-\eta^{(\kappa)})b^{(\kappa)} + \frac{\eta^{(\kappa)}}{4}\operatorname{tr}\Big[\Lambda\langle(LDL^T)\circ(LDL^T)\rangle\Big], \qquad (29)$$

$$d_{jk}^{(\kappa+1)} = (1-\eta^{(\kappa)})d_{jk}^{(\kappa)} + \frac{\eta^{(\kappa)}\langle\omega\rangle}{2}\langle(LDL^T)\circ(LDL^T)\rangle_{jk}, \qquad (30)$$

where $\psi'$ denotes the trigamma function,

$$\begin{aligned}\frac{\partial \mathcal{L}_1}{M_{Ljk}} =& \Big\{ -[nS + (M_L M_D M_L^T)\circ\Lambda]M_L M_D - [M_L(M_D \circ M_D + V_D)]\circ(\Lambda V_L) \\ & - (M_L V_D)\circ[\Lambda(M_L\circ M_L)]\Big\}_{jk}, \qquad (31)\\ \frac{\partial \mathcal{L}_1}{V_{Ljk}} =& \Big\{ -\frac{n}{2}\operatorname{diag}(S)\operatorname{diag}(M_D)^T - \frac{1}{2}\Lambda(M_L\circ M_L + V_L)(M_D\circ M_D + V_D)\Big\}_{jk}, \qquad (32)\\ \frac{\partial \mathcal{L}_1}{M_{Djj}} =& \Big\{ -\frac{1}{2}\operatorname{diag}\left\{M_L^T[nS + (M_L M_D M_L^T)\circ\Lambda]M_L\right\} - \frac{n}{2}V_L^T\operatorname{diag}(S) \\ & - \frac{1}{2}\operatorname{diag}\left[V_L^T\Lambda(V_L + 2M_L\circ M_L)\right]\circ\operatorname{diag}(M_D)\Big\}_{j}, \qquad (33)\\ \frac{\partial \mathcal{L}_1}{V_{Djj}} =& -\frac{1}{4}\operatorname{diag}\left[(M_L\circ M_L + V_L)^T\Lambda(M_L\circ M_L + V_L)\right]_j, \qquad (34)\end{aligned}$$

and

$$\begin{aligned}\langle (LDL^T)\circ(LDL^T)\rangle =& (M_L\circ M_L + V_L)(M_D\circ M_D + V_D)(M_L\circ M_L + V_L)^T \\ & - (M_L\circ M_L)(M_D\circ M_D)(M_L\circ M_L)^T \\ & + (M_L M_D M_L^T)\circ(M_L M_D M_L^T). \qquad (35)\end{aligned}$$

The derivation of the algorithm is provided in Appendix C.

### 3.4. Stochastic Gradients

The computational bottleneck of the update rules in Eq. (24)-(30) lies in the matrix-matrix product, whose computational complexity is $\mathcal{O}(p^3)$. To reduce the computational cost, we resort to stochastic gradients. More precisely, the product of two $p\times p$ matrices $A$ and $B$ can be estimated unbiasedly as:

$$AB \approx \frac{p}{s}\sum_{j\in\mathcal{S}} C^j, \qquad (36)$$

$$C^j_{j,:} = A_{j,:}B, \qquad (37)$$

where $C^j$ denotes a $p\times p$ zero matrix except for row $j$, which equals to $A_{j,:}B$, $\mathcal{S} = \{j_1, j_2, \cdots, j_s\}$ denotes a minibatch of $s$ indices that are uniformly sampled at random from $\{1, 2, \cdots, p\}$, and $A_{j,:}$ denotes row $j$ of matrix $A$. Note that the resulting variance of the stochastic gradients increases linearly with $p$ given fixed $s$. So as to break the dependence of the variance on $p$, we can deal with the normalized ELBO $\tilde{\mathcal{L}} = \mathcal{L}/p$ instead of $\mathcal{L}$ in the KL proximal gradient algorithm (21), that is,

$$\boldsymbol{\mu}^{(\kappa+1)} = \operatorname*{argmin}_{\boldsymbol{\mu}} \ -\boldsymbol{\mu}^T\nabla_{\boldsymbol{\mu}}\tilde{\mathcal{L}}(\boldsymbol{\mu}^{(\kappa)}) + \frac{1}{\rho^{(\kappa)}}\mathbb{KL}[q(\boldsymbol{z}|\boldsymbol{\mu})|q(\boldsymbol{z}|\mu^{(\kappa)})]. \qquad (38)$$

Equivalently, we can replace the step size $\rho$ by $\rho/p$ in (21):

$$\boldsymbol{\mu}^{(\kappa+1)} = \operatorname*{argmin}_{\boldsymbol{\mu}} \ -\boldsymbol{\mu}^T\nabla_{\boldsymbol{\mu}}\mathcal{L}(\boldsymbol{\mu}^{(\kappa)}) + \frac{p}{\rho^{(\kappa)}}\mathbb{KL}[q(\boldsymbol{z}|\boldsymbol{\mu})|q(\boldsymbol{z}|\mu^{(\kappa)})]. \qquad (39)$$

On the other hand, if we fix $s$ as $p$ grows, the computational complexity in each iteration of BISN is only $\mathcal{O}(p^2)$.

### 3.5. Contribution of the LDL Decomposition

It is worth emphasizing that replacing $K$ with $LDL^T$ in the Bayesian formulation is particularly advantageous in BISN. The computational complexity of BISN per iteration cannot be reduced to be $\mathcal{O}(p^2)$ without this reparameterization. As BISN utilizes optimization algorithms to learn the parameters of the variational distribution, we compare BISN with the frequentist methods (Duchi et al., 2008; Scheinberg et al., 2010; Rolfs et al., 2012; Hsieh et al., 2014, 2013; Treister and Turek, 2014) below to demonstrate the merits of using the LDL decomposition. Despite the various optimization algorithms used in the frequentist methods, they all seek to minimize the same objective function:

$$K = \underset{K \succeq 0}{\operatorname{argmin}} \operatorname{tr}(SK) - \log\det K + \lambda \|K\|_1, \tag{40}$$

and the update procedure per iteration in the majority of the frequentist methods can be summarized as: 1) determine the update direction based on the gradients w.r.t. the precision matrix $K$, and 2) update $K$ in that direction with a proper step size such that the value of the objective function (40) is sufficiently decreased and $K$ is positive semi-definite.

First, let us focus on the positive semi-definiteness of $K$. The frequentist methods have to check the positive semi-definiteness in every iteration by performing Cholesky or eigenvalue decomposition. The corresponding computational complexity is $\mathcal{O}(p^3)$. In BISN, however, the variational distribution of $K$ is guaranteed to be positive semi-definite after setting the variational distributions of the diagonal entries in $D$ to be Gamma distributions. BISN successfully circumvents the computationally burdensome operation of checking the positive semi-definiteness.

Second, we turn our attention to the objective function. In the frequentist methods, the computational bottleneck of evaluating the objective function lies in computing $\log\det K$, whose computational cost is $\mathcal{O}(p^3)$. By contrast, after reparameterizing $K$ by $LDL^T$ in BISN, $\log\det K$ reduces to $\sum_j \log D_{jj}$ and the corresponding terms in the update rules are also simplified.

Third, we would like to discuss the gradient. The most complicated operation in the gradients of the frequentist methods is matrix inverse $K^{-1}$, resulting from the term $\log\det K$ in the objective function. The computational complexity of matrix inverse is $\mathcal{O}(p^3)$. Unfortunately, there are no unbiased but computationally cheap estimates of matrix inverse to the best of our knowledge. In other words, stochastic gradients are not applicable in this setting. As opposed to the frequentist methods, owing to the LDL decomposition, the most complicated operation in the BISN update rules (24)-(35) is matrix product, which is already more computationally efficient than matrix inverse. Although the computational complexity of matrix product is still $\mathcal{O}(p^3)$, we can easily find an unbiased stochastic estimate of it with complexity $O(p^2)$ and further derive stochastic gradients as discussed in Section 3.4.

## 4. Variance Reduction via Decaying Recursive Stochastic Gradient (DRSG)

As proven in Khan et al. (2016), if we run $t$ iterations of the above KL proximal stochastic gradient variational Bayes algorithm (21) with a fixed step size $\rho^{(\kappa)}$ for all $\kappa \in \{1, \cdots, t\}$ and define

$$\boldsymbol{g}^{(\kappa)} = \frac{1}{\rho^{(\kappa)}}(\boldsymbol{\mu}^{(\kappa)} - \boldsymbol{\mu}^{(\kappa+1)}), \tag{41}$$

where $\mu$ denotes the mean parameters of the variational distributions, then we have

$$\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \frac{2l[\mathcal{L}^* - \mathcal{L}^0]}{\gamma_*^2 t} + \frac{c_1\sigma^2}{\gamma_*}, \tag{42}$$

where $\kappa$ is uniformly picked at random from $\{0, 1, \cdots, t\}$, the expectation $\mathbb{E}$ is taken w.r.t. all kinds of randomness, $l$ is the Lipschitz constant of $\nabla_{\boldsymbol{\mu}} E_q[\log p(\boldsymbol{x}, \boldsymbol{z})]/p$, $\mathcal{L}^*$ and $\mathcal{L}^0$ are respectively the global maximum and the initial value of the ELBO, $\gamma_*$ and $c_1$ are constants satisfying the following constraints:

$$\gamma_* = \gamma - \frac{1}{2c_1}, \tag{43}$$

$$(\boldsymbol{\mu} - \boldsymbol{\mu}')^T \nabla_\mu \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{\mu})|q(\boldsymbol{z}|\boldsymbol{\mu}^{(\kappa)})] \geq \gamma\|\boldsymbol{\mu} - \boldsymbol{\mu}'\|^2, \tag{44}$$

$$\gamma > 0, \tag{45}$$

$$c_1 > \frac{1}{2\gamma}, \tag{46}$$

where $\sigma^2$ is the variance of the stochastic gradient. In the above convergence analysis, the second term in (42) does not decrease with the number of iterations. To achieve an $\epsilon$-accurate solution (i.e., $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] < \epsilon$), we have to increase the minibatch size $s$ and reduce the fixed step size $\rho$ as a function of $1/\epsilon$ (Ghadimi et al., 2016; Reddi et al., 2016). The resulting number of iterations to reach the $\epsilon$-accuracy is $\mathcal{O}(1/\epsilon^2)$ in theory (Ghadimi et al., 2016; Reddi et al., 2016). Let us define the run time guarantee to be the number of floating point operations (flops) to reach the $\epsilon$-accuracy. For BISN, the resulting run time guarantee under this scenario is $\mathcal{O}(p^2/\epsilon^2)$. However, As pointed out in (Reddi et al., 2016), we typically use a constant minibatch size $s$ that is invariant w.r.t. $\epsilon$ in practice, and consequently, there is no guarantee of convergence. Of course using the diminishing step size $\rho^{(\kappa)}$ that decreases with the number of iterations $\kappa$ would trivially ensure convergence, but the algorithm may terminate before reaching a stationary point.

To mitigate the above issue, we propose a method named KL proximal Decaying Recursive Stochastic Gradient (DRSG) to reduce the variance and to accelerate the convergence. This approach does not require increasing the minibatch size $s$ as a function $1/\epsilon$. Furthermore, as proven in Section 5, the run time guarantee of KL proximal DRSG for BISN is $\mathcal{O}(p^2/\epsilon^{\frac{3}{2}})$. In other words, it takes a fewer number of iterations to reach the $\epsilon$-accuracy. Concretely, we borrow the idea from the recursive stochastic gradient algorithm (i.e., SARAH) proposed in Nguyen et al. (2017). However, we introduce a decaying coefficient $r$ satisfying $r < 1$ to the recursive gradient, and successfully reduce the number of iterations to achieve the

$\epsilon$-accuracy from $\mathcal{O}(1/\epsilon^2)$ in Nguyen et al. (2017) to $\mathcal{O}(1/\epsilon^{\frac{3}{2}})$. In the sequel, we first introduce the original recursive stochastic gradient algorithm. We then elaborate on the proposed KL proximal DRSG algorithm.

The pivotal idea of recursive stochastic gradients is to employ the information of stochastic gradient estimates in previous iterations to improve the estimate in the current iteration. Specifically, suppose that our objective is to find $\boldsymbol{\mu}$ that minimizes the following finite sum problem:

$$f(\boldsymbol{\mu}) = \frac{1}{p}\sum_{j=1}^{p} f_j(\boldsymbol{\mu}), \tag{47}$$

where $f_j(\boldsymbol{\mu})$ is non-convex but $l$-smooth. The exact gradient can be expressed as:

$$\nabla_{\mu} f(\boldsymbol{\mu}) = \frac{1}{p}\sum_{j=1}^{p} \nabla_{\mu} f_j(\boldsymbol{\mu}). \tag{48}$$

The recursive stochastic gradients then proceeds as follows (Nguyen et al., 2017):

$$R^{(0)} = \nabla_{\mu} f(\mu^{(0)}), \tag{49}$$

$$R^{(\kappa)} = \frac{1}{s}\sum_{j\in\mathcal{S}^{(\kappa)}} \nabla_{\mu} f_j(\boldsymbol{\mu}^{(\kappa)}) + \Big[R^{(\kappa-1)} - \frac{1}{s}\sum_{j\in\mathcal{S}^{(\kappa)}} \nabla_{\mu} f_j(\boldsymbol{\mu}^{(\kappa-1)})\Big], \tag{50}$$

$$\boldsymbol{\mu}^{(\kappa+1)} = \boldsymbol{\mu}^{(\kappa)} - \rho^{(\kappa)} R^{(\kappa)}, \tag{51}$$

where $\rho^{(\kappa)}$ is the step size in iteration $\kappa$, and $\mathcal{S}^{(\kappa)}$ is a minibatch of $\{1, \cdots, p\}$ with cardinality $s$. In summary, we compute the recursive stochastic gradient $R^{(\kappa)}$ in each iteration and use it to update the parameter $\boldsymbol{\mu}$. More concretely, in each iteration $\kappa$, we first randomly choose a minibatch $\mathcal{S}^{(\kappa)}$. We then update the recursive gradient $R^{(\kappa)}$ by subtracting the stochastic gradient $1/s\sum_{j\in\mathcal{S}^{(\kappa)}} \nabla_{\mu} f_j(\boldsymbol{\mu}^{(\kappa-1)0})$ w.r.t. $\boldsymbol{\mu}^{(\kappa-1)}$ in the previous iteration $\kappa-1$ and adding the stochastic gradient w.r.t. $\boldsymbol{\mu}^{(\kappa)}$ in the current iteration. Note that $R^{(\kappa)}$ is an unbiased estimate of the exact gradient. It has been proven in Nguyen et al. (2017) that the recursive stochastic gradient algorithm takes $\mathcal{O}(1/\epsilon^2)$ to achieve an $\epsilon$-accurate solution if $s = 1$, $\rho^{(\kappa)} = \rho$ is a constant, and $\rho = \mathcal{O}(1/(l\sqrt{t}))$. Note that the batch size is invariant with $\epsilon$ by means of the recursive gradient.

Next, let us focus on the proposed KL proximal DRSG algorithm. We aim to maximize the normalized ELBO $\tilde{\mathcal{L}}$ (i.e., to minimize $-\tilde{\mathcal{L}}$). To facilitate our analysis, we make a few assumptions:

1. The normalized ELBO $\tilde{\mathcal{L}}$ is a function of the mean parameters $\boldsymbol{\mu}$ of the variational distributions, and it can be decomposed into a “difficult” term $f$ and an ”easy” term $h$ as in Khan et al. (2015, 2016):

$$-\tilde{\mathcal{L}} = f(\boldsymbol{\mu}) + h(\boldsymbol{\mu}). \tag{52}$$

   In BISN, $f(\boldsymbol{\mu})$ denotes the part of $\tilde{\mathcal{L}}$ that requires stochastic approximation (i.e., terms with matrix product), which can be regarded as the mean averaged over $p$ terms, and $h(\boldsymbol{\mu})$ denotes the remaining terms, which are linear functions of $\boldsymbol{\mu}$. In other words, $h(\boldsymbol{\mu}) = \boldsymbol{\mu}^T\nabla_{\boldsymbol{\mu}} h(\boldsymbol{\mu})$.

2. $f(\boldsymbol{\mu}) = 1/p \sum_{j=1}^{p} f_j(\boldsymbol{\mu})$ is $l$-smooth, that is,

$$\|\nabla f(\boldsymbol{\mu}) - \nabla f(\boldsymbol{\mu}')\| \leq l \|\boldsymbol{\mu} - \boldsymbol{\mu}'\|. \tag{53}$$

3. $f_j(\boldsymbol{\mu})$ for any $j \in \{1, \cdots, p\}$ is also $l$-smooth.

4. The variance of $\nabla f_j(\boldsymbol{\mu})$ can be upper bounded as

$$\boldsymbol{v}^{(\kappa)} = \frac{1}{p} \sum_{j=1}^{p} \|\nabla f_j(\boldsymbol{\mu}^{(\kappa)})\|^2 - \|\nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2 \leq \sigma^2, \quad \forall \kappa, \tag{54}$$

   where $\sigma^2$ is a constant.

5. There exists $\gamma > 0$ such that

$$(\boldsymbol{\mu} - \boldsymbol{\mu}') \nabla_{\mu} \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{\mu}) | q\big(\boldsymbol{z}|\mu^{(\kappa)}\big)] \geq \boldsymbol{\gamma} \|\mu - \mu'\|^2. \tag{55}$$

   This assumption is always satisfied as long as all $\boldsymbol{\mu}^{(\kappa)}$ stay within a compact set (Khan et al., 2016).

6. Define $\boldsymbol{g}^{(\kappa)} = \boldsymbol{g}(\boldsymbol{\mu}^{(\kappa)}, R^{(\kappa)}, \rho^{(\kappa)}) = (\boldsymbol{\mu}^{(\kappa)} - \boldsymbol{\mu}^{(\kappa+1)})/\rho^{(\kappa)}$.

The KL proximal DRSG algorithm is then proceeded by iterating the following step:

$$\boldsymbol{\mu}^{(\kappa+1)} = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \; \boldsymbol{\mu}^T R^{(\kappa)} + h(\boldsymbol{\mu}^{(\kappa)}) + \frac{1}{\rho^{(\kappa)}} \mathrm{KL}[q(\boldsymbol{z}|\boldsymbol{\mu}) | q\big(\boldsymbol{z}|\boldsymbol{\mu}^{(\kappa)}\big)], \tag{56}$$

where the decaying recursive gradient is updated as

$$R^{(0)} = \nabla_{\mu} f(\mu^{(0)}), \tag{57}$$

$$R^{(\kappa)} = \frac{1}{s} \sum_{j \in \mathcal{S}^{(\kappa)}} \nabla_{\mu} f_j(\boldsymbol{\mu}^{(\kappa)}) + r^{(\kappa)} \Big[ R^{(\kappa-1)} - \frac{1}{s} \sum_{j \in \mathcal{S}^{(\kappa)}} \nabla_{\mu} f_j(\boldsymbol{\mu}^{(\kappa-1)}) \Big], \tag{58}$$

and $r^{(\kappa)} < 1$ is the decaying coefficient. Note that Eq. (56) is equivalent to (38), since $h(\boldsymbol{\mu})$ is a linear function of $\boldsymbol{\mu}$. As a result, there is no need to derive new update rules for the natural parameters in BISN; we only need to replace the stochastic parts in the update rules (24)-(30) by the corresponding DRSGs. As proven in the next section, the number of iterations the KL proximal DRSG algorithm takes to reach the $\epsilon$-accuracy is $\mathcal{O}(1/\epsilon^{\frac{3}{2}})$.

We note that classical variance reduced stochastic gradient methods can be also applied to BISN, such as ProxSVRG/SAGA (Reddi et al., 2016). These methods lead to a run time guarantee of $\mathcal{O}(p^{\frac{8}{3}}/\epsilon)$. In comparison with ProxSVRG/SAGA, the proposed KL proximal DRSG method scales more gracefully with the dimension $p$ as the run time guarantee is $\mathcal{O}(p^2/\epsilon^{\frac{3}{2}})$. As a summary, we list the run time guarantee of different methods in Table 2.

Table 2: Comparison of the run time guarantee between different algorithms when being applied to BISN. Note that the original stochastic gradient algorithm is convergent only with decaying step size, whereas the other methods converge with a constant step size.

| Methods | Exact Gradient<br>(Khan et al., 2016) | Stochastic Gradient<br>(Khan et al., 2016) | ProxSVRG/SAGA<br>(Reddi et al., 2016) | SARAH<br>(Nguyen et al., 2017) | DRSG |
|---|---|---|---|---|---|
| Run Time Guarantee | $\mathcal{O}(p^3/\epsilon)$ | $\mathcal{O}(p^2/\epsilon^2)$ | $\mathcal{O}(p^{\frac{8}{3}}/\epsilon)$ | $\mathcal{O}(p^2/\epsilon^2)$ | $\mathcal{O}(p^2/\epsilon^{\frac{3}{2}})$ |

## 5. Convergence Analysis and Run Time Guarantee

In this section, we provide the convergence analysis of the proposed algorithm. We further derive the run time guarantee, which only scales quadratically with the dimension $p$ (i.e., linearly with the number of unknown parameters to be estimated). To the best of our knowledge, we are among the first to reduce the time complexity from $\mathcal{O}(p^3)$ to $\mathcal{O}(p^2)$ for the problem of Gaussian graphical model selection.

**Proposition 1** *Suppose the assumptions in Section 4 hold. If we run $t$ iterations of KL proximal DRSG* (56) *with a fixed step size $\rho > 0$, then we have:*

$$-\tilde{\mathcal{L}}^* \leq -\tilde{\mathcal{L}}^0 - \left[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\right]\sum_{\kappa=0}^{t}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1\rho}{2}\sum_{\kappa=1}^{t-1}\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2], \quad (59)$$

*where $c_1$ is a positive constant, and $\tilde{\mathcal{L}}^*$ and $\tilde{\mathcal{L}}^0$ denotes respectively the maximum and initial value of the normalized ELBO $\tilde{\mathcal{L}}$.*

**Proof** See Appendix D. ■

Since our objective is to find the upper bound of $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2]$, we intend to express the third term on the right hand side (RHS) of (59) as a function of $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2]$. That is the objective of Proposition 2 and 3.

**Proposition 2** *Consider $R^{(\kappa)}$ defined in* (58) *in the KL proximal DRSG algorithm, then for any $\kappa > 0$,*

$$\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2] = \sum_{m=1}^{\kappa}\left[\prod_{j=m+1}^{\kappa} {r^{(j)}}^2\Big(\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]\Big)\right]. \quad (60)$$

**Proof** See Appendix E. ■

Given Proposition 2, we then seek the upper bound of the term $E[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]$.

**Proposition 3** *Consider $R^{(m)}$ defined in (58) and $\boldsymbol{v}^{(m)}$ defined in (54), then we can upper bound $E[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]$ as follows:*

$$\begin{aligned}&\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]\\ \leq\ &\frac{1}{s}\frac{p-s}{p-1}\Big(r^{(m)}l^2\rho^2\mathbb{E}[\|\boldsymbol{g}^{(m-1)}\|^2] + (1-r^{(m)})\boldsymbol{v}^{(m)} - r^{(m)}(1-r^{(m)})\boldsymbol{v}^{(m-1)}\Big).\end{aligned} \tag{61}$$

**Proof** See Appendix F. ■

By substituting the results in Proposition 2 and 3 into Proposition 1 and further relaxing the bound, we obtain the following theorem:

**Theorem 1** *Suppose the assumptions in Section 4 hold. If we run $t$ iterations of KL proximal DRSG (56) with constant decaying coefficient $r < 1$, and choose the constant $c_1$ and the step size $\rho$ satisfying:*

$$c_1 \geq \frac{1}{2\gamma}, \tag{62}$$

$$0 < \rho < \frac{\sqrt{a_1^2 + 4a_2a_0} - a_1}{2a_2}, \tag{63}$$

*then we can obtain:*

$$\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \frac{\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0}{t\rho(-a_2\rho^2 - a_1\rho + a_0)} + \frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}\frac{c_1\sigma^2}{-a_2\rho^2 - a_1\rho + a_0}, \tag{64}$$

*where $\kappa$ is uniformly chosen at random from $\{0, \cdots, t-1\}$, the expectation $\mathbb{E}$ is taken w.r.t. all kinds of randomness, and*

$$a_0 = \gamma - \frac{1}{2c_1}, \tag{65}$$

$$a_1 = \frac{l}{2}, \tag{66}$$

$$a_2 = \frac{r}{1-r^2}\frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}l^2. \tag{67}$$

**Proof** See Appendix G. ■

Note that by choosing the decaying coefficient $r$ properly, we can decrease the second term on the RHS of (64) to the desired accuracy $\epsilon$ without increasing the minibatch size $s$, as shown in Theorem 2 below.

**Theorem 2** *Suppose the assumptions in Section 4 hold and we run KL proximal DRSG (56) with minibatch size $s$, decaying coefficient $r$, and step size $\rho$ such that*

$$s = \frac{p}{c_2(p-1)+1}, \tag{68}$$

$$r = \frac{1-c_3\epsilon}{1+c_3\epsilon}, \tag{69}$$

$$\rho = \frac{\sqrt{a_1^2 + 3a_2a_0} - a_1}{3a_2}, \tag{70}$$

*where $c_2 \in (0,1]$ is a fixed constant, $c_3 = a_0/9c_1c_2c_4\sigma^2$, $c_4$ is a fixed constant satisfying $c_4 > \max(1, a_0\epsilon/9c_1c_2\sigma^2)$, $c_1 > 1/2\gamma$, and $a_0$, $a_1$, and $a_2$ are defined in* (65)-(67). *It is sufficient to run the algorithm for $t = \mathcal{O}(1/\epsilon^{\frac{3}{2}})$ iterations in order to obtain the $\epsilon$-accuracy, that is,*

$$\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \epsilon, \tag{71}$$

*where $\kappa$ is uniformly chosen at random from $\{0, \cdots, t-1\}$ and the expectation $\mathbb{E}$ is taken w.r.t. all kinds of randomness.*

**Proof** See Appendix H. ■

When applying KL proximal DRSG to BISN, the computational cost in each iteration is $\mathcal{O}(sp^2)$. Given the specification of $s$ in (68), $\mathcal{O}(sp^2)$ reduces to $\mathcal{O}(p^2)$. Additionally, we notice that we need to compute the full gradient of $f(\boldsymbol{\mu})$ as an initialization of the KL proximal DRSG algorithm (57). In BISN, the computational complexity of evaluating the full gradient for dense mean and variance matrix of the $L$ component $M_L$ and $V_L$ is $\mathcal{O}(p^3)$, cf. Eqs. (24)-(30). However, we manage to reduce the computational cost to be $\mathcal{O}(p^2)$ by initializing the mean matrix $M_L$ to be an identity matrix such that it is easy to compute those matrix products associated with $M_L$. For the remaining matrix products w.r.t. $V_L$ (e.g., $\Lambda V_L$ and $V_L^T(M_D \circ M_D + V_D)V_L$), we initialize the non-zero elements to be the same respectively in $\Lambda$, $V_L$, $M_D$, and $V_D$, thus these products can also be obtained with $\mathcal{O}(p^2)$ operations. As a result, we can obtain the following theorem:

**Theorem 3** *When applying KL proximal DRSG to BISN, the run time guarantee is $\mathcal{O}(p^2/\epsilon^{\frac{3}{2}})$ and the time complexity in terms of $p$ is $\mathcal{O}(p^2)$.*

Note that the proposed KL proximal DRSG algorithm can also be coupled with different adaptive step size schemes to further increase the convergence rate. In practice, we specify an upper bound on the step size $\rho^{(\kappa)}$ and then exploit the adaptive step size scheme proposed in Ranganath et al. (2013) to determine $\rho^{(\kappa)}$.

## 6. Experimental Results

In this section, we benchmark the proposed BISN method with several start-of-the-art methods for automatic graphical model selection, including both tuning-sensitive methods and the tuning-insensitive method, TIGER (Liu and Wang, 2017). Specifically, for tuning-sensitive methods, we select G-ISTA (Rolfs et al., 2012) and BIG&QUIC (Hsieh et al., 2013). Both methods maximize the log-likelihood of the precision matrix $K$ with an $\ell_1$-norm penalty on $K$ such that the resulting $K$ is sparse. G-ISTA is shown to be the fastest frequentist method for Gaussian graphical model selection in the literature (Rolfs et al., 2012; Treister and Turek, 2014) when there is sufficient memory to store the $p \times p$ empirical covariance matrix $S$. BIG&QUIC is applicable to one million variables as long as the memory is large enough to store the sparse precision matrix $K$ and the observed data $\boldsymbol{x}^{\{1:n\}}$. Since we have to determine the value of the penalty parameter in these tuning-sensitive methods, we consider here two methods for regularization selection: BINCO[4] (Li et al., 2013) and

[4] For BINCO, we bootstrap 100 subsample sets from the original observed samples, and we test 6 candidate penalty parameters whose logarithm are $-5, -4, \cdots, 0$.

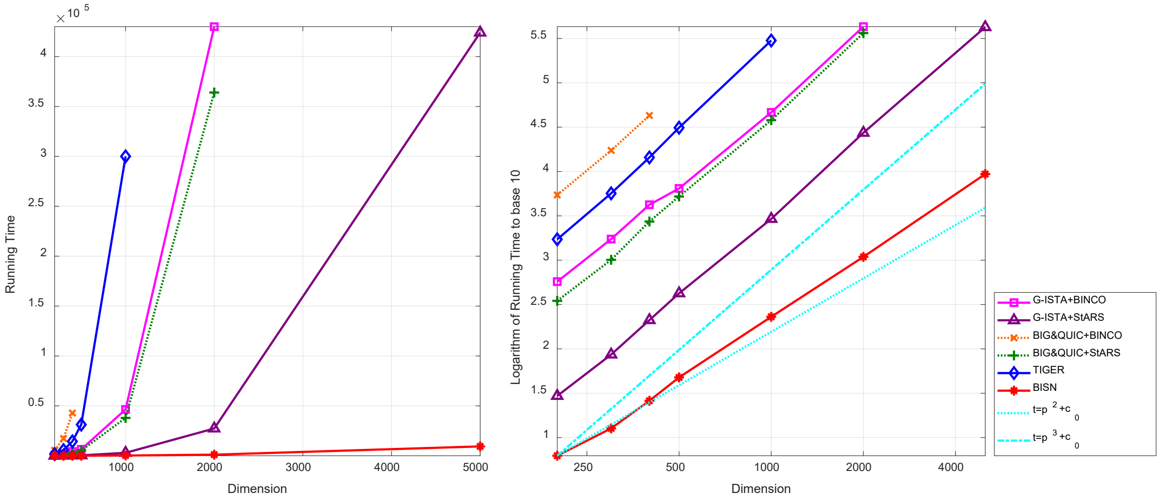
StARS[5] (Liu et al., 2010). The former selects a stable graph directly from subsampled or bootstrapped sample sets across a series of candidate penalty parameters, whereas the latter selects the smallest penalty parameter such that the corresponding graph across subsample sets are suitably stable. These stability-based methods have been shown to be superior to traditional methods such as cross validation, AIC, and BIC in terms of structure learning (Meinshausen and Bühlmann, 2010; Liu et al., 2010; Yu et al., 2012; Li et al., 2013). In particular for BINCO, after obtaining the graph structure, the non-zero elements in $K$ are further estimated by maximizing their likelihood. For BISN, we set the minibatch size $s = p/(0.001(p-1)+1)$ and the decaying coefficient $r = 0.5$. In the sequel, we first present the results of synthetic data, where the ground truth is given. We then apply the proposed BISN approach to a variety of real data sets, including stock, gene, and fMRI data.

### 6.1. Synthetic Data

We generate synthetic data as follows. We first generate a sparse lower triangular matrix $C$ with positive diagonal and $n_e$ non-zero off-diagonal entries. Specifically, the diagonal entries and the non-zero off-diagonal of $C$ follow a uniform distribution respectively on $[1, 1.5]$ and $[-1, -0.5] \cup [0.5, 1]$. Next, we compute the sparse precision matrix as $K = CC^T$. We then randomly permute the rows and columns of $K$ simultaneously such that its Cholesky decomposition is not sparse anymore. We also rescale the rows and columns of $K$ such that the resulting covariance matrix $\Sigma = K^{-1}$ has unit diagonal. Finally, we draw $n$ samples from the Gaussian distribution with zero mean and precision matrix $K$. We compare all algorithms by means of the accuracy of structure estimation, parameter estimation, model fitting, and computational time. More specifically, for accuracy of graph estimation, we consider three criteria, including precision, recall, and $F_1$-score. Precision is defined as the proportion of correctly estimated edges to all the edges in the estimated graph; recall is defined as the proportion of successfully estimated edges to all the edges in the true graph; $F_1$-score is defined as 2·precision·recall/(precision+recall). For parameter estimation, we evaluate the mean squared error (MSE) between the estimated and true precision matrix. Finally, for model fitting, we evaluate the negative log-likelihood (NegLogLLH) of the observed data and the BIC score.

We first investigate the scalability of all methods. Concretely, we set $n = 4p$ and $n_e = 2p$ and consider $p = 200, 300, 400, 500, 1000, 2000, 5000$. The results averaged over 10 trials are summarized in Table 3. We further plot the computational time as a function of $p$ in Figure 1, and the estimates of 10 randomly selected off-diagonal elements in the case of $p = 1000$ in Figure 2. BISN performs the best in terms of graph structure estimation, parameter estimation, and data fitting, with the shortest computational time. Moreover, as demonstrated in Figure 1(b), the slope of the red line (i.e., the logarithm of the BISN running time v.s. the dimension) is similar to that of the cyan dotted line which represents the function $t = p^2 + c$, where $t$ is the running time and $c$ is a constant. In other words, the computational time of BISN scales approximately quadratically with the dimension. On the other hand, the slopes of the other lines are almost the same as that of the cyan dash-dot

[5] For StARS, we follow the implementation in Zhao et al. (2012) and set the number of subsample sets to be 20 and the variability threshold to be 0.1. We begin with the penalty parameter $\lambda = 1$ and decrease its logarithm by 0.1 until the variability of the graph is below the threshold.



(a) (b)

Figure 1: Computational time as a function of dimension for different methods.

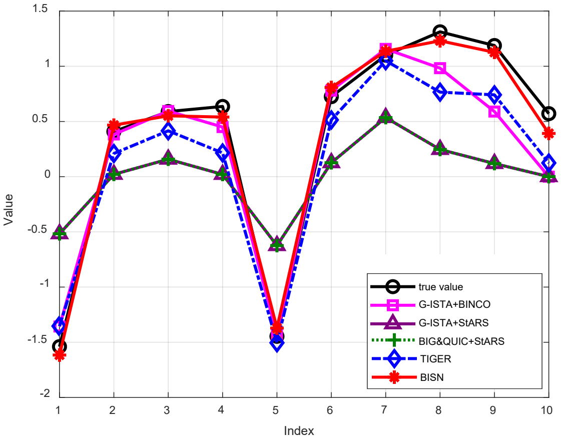


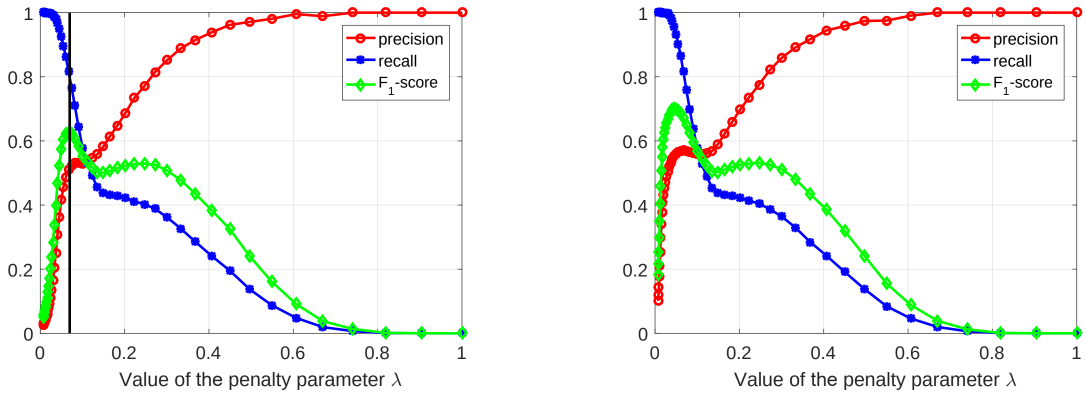
Figure 2: The estimates from the different methods and the true values of 10 randomly selected non-zero off-diagonal entries from the true precision matrix when $p = 1000$.

line (i.e., $t = p^3 + c$), implying that the computational time of the state-of-the-art methods is a cubic function of the dimension.

By contrast, G-ISTA+StARS and BIG&QUIC+StARS perform the worst in terms of structure recovery. This can be explained by Figure 3(a), in which we depict the precision, recall, and $F_1$-score resulting from G-ISTA as a function of the penalty parameter $\lambda$. Although StARS (Liu et al., 2010) successfully finds a penalty parameter $\lambda$ that is close to the one which gives the highest $F_1$-score, the highest $F_1$-score is around 0.63 no matter how we tune the penalty parameter. This observation indicates that the frequentist methods that maximize the penalized likelihood of $K$ including G-ISTA, BIG&QUIC, etc. cannot reliably

Table 3: Accuracy and computational time of the state-of-the-art methods and BISN as $p$ increases. The results are averaged over 10 trials. The corresponding standard deviation is listed in the brackets. The number of parameters in the true graphs (i.e., the node number plus the edge number) averaged over the 10 trials is respectively 1.11e3, 1.63e3, 2.22e3, 2.80e3, 5.68e3, 1.14e4, 2.83e4, for $p = 200, 300, 400, 500, 1000, 2000, 5000$.

| | | $p$ | 200 | 300 | 400 | 500 | 1000 | 2000 | 5000 |
|---|---|---|---|---|---|---|---|---|---|
| Structure Recovery | Precision | G-ISTA+BINCO | 0.91 (1.13e-2) | 0.91 (1.31e-2) | 0.92 (1.74e-2) | 0.91 (8.04e-3) | 0.91 (4.54e-3) | 0.92 (4.27e-3) | |
| | | G-ISTA+StARS | 0.44 (3.00e-2) | 0.45 (1.97e-2) | 0.47 (1.88e-2) | 0.48 (1.40e-2) | 0.51 (1.25e-2) | 0.54 (7.40e-3) | 0.58 (6.88e-3) |
| | | BIG&QUIC+BINCO | 0.91 (1.13e-2) | 0.91 (1.31e-2) | 0.92 (1.74e-2) | | | | |
| | | BIG&QUIC+StARS | 0.44 (3.00e-2) | 0.45 (1.97e-2) | 0.47 (1.88e-2) | 0.48 (1.40e-2) | 0.51 (1.25e-2) | 0.54 (7.40e-3) | |
| | | TIGER | 0.85 (1.58e-2) | 0.87 (7.64e-3) | 0.88 (9.82e-3) | 0.89 (6.49e-3) | 0.90 (3.24e-3) | | |
| | | BISN | **0.97** (1.01e-2) | **0.99** (9.70e-3) | **0.99** (4.68e-3) | **1.00** (3.43e-3) | **1.00** (1.72e-3) | **0.99** (6.25e-3) | **0.97** (2.15e-2) |
| | Recall | G-ISTA+BINCO | 0.60 (1.63e-2) | 0.68 (1.67e-2) | 0.76 (1.12e-1) | 0.83 (1.10e-2) | 0.84 (4.04e-3) | 0.88 (1.54e-3) | |
| | | G-ISTA+StARS | 0.55 (1.32e-2) | 0.60 (1.74e-2) | 0.64 (1.85e-2) | 0.69 (1.68e-2) | 0.81 (1.39e-2) | 0.90 (7.89e-3) | 0.96 (4.43e-3) |
| | | BIG&QUIC+BINCO | 0.60 (1.63e-2) | 0.68 (1.67e-2) | 0.76 (1.12e-1) | | | | |
| | | BIG&QUIC+StARS | 0.55 (1.32e-2) | 0.60 (1.74e-2) | 0.64 (1.85e-2) | 0.69 (1.68e-2) | 0.81 (1.39e-2) | 0.90 (7.89e-3) | |
| | | TIGER | 0.72 (1.43e-2) | 0.80 (1.65e-2) | 0.84 (1.79e-2) | 0.88 (1.38e-2) | 0.96 (5.63e-3) | | |
| | | BISN | **0.95** (1.23e-2) | **0.98** (6.91e-3) | **0.99** (3.07e-3) | **0.99** (6.77e-3) | **1.00** (1.10e-3) | **1.00** (1.75e-3) | **1.00** (2.81e-4) |
| | $F_1$-score | G-ISTA+BINCO | 0.72 (1.29e-2) | 0.77 (1.17e-2) | 0.83 (7.15e-2) | 0.86 (7.62e-3) | 0.87 (2.93e-3) | 0.90 (2.40e-3) | |
| | | G-ISTA+StARS | 0.49 (1.66e-2) | 0.51 (1.07e-2) | 0.54 (1.13e-2) | 0.57 (1.26e-2) | 0.63 (8.83e-3) | 0.68 (6.16e-3) | 0.72 (4.55e-3) |
| | | BIG&QUIC+BINCO | 0.72 (1.29e-2) | 0.77 (1.17e-2) | 0.83 (7.15e-2) | | | | |
| | | BIG&QUIC+StARS | 0.49 (1.66e-2) | 0.51 (1.07e-2) | 0.54 (1.13e-2) | 0.57 (1.26e-2) | 0.63 (8.83e-3) | 0.68 (6.16e-3) | |
| | | TIGER | 0.78 (1.26e-2) | 0.83 (9.92e-3) | 0.86 (1.28e-2) | 0.89 (7.51e-3) | 0.93 (3.41e-3) | | |
| | | BISN | **0.96** (5.79e-3) | **0.98** (4.73e-3) | **0.99** (2.62e-3) | **0.99** (3.11e-3) | **1.00** (7.74e-4) | **0.99** (2.79e-3) | **0.99** (1.11e-2) |
| Model Fitting & Parameters Estimation | MSE | G-ISTA+BINCO | 4.33e-2 (1.04e-2) | 2.69e-2 (5.12e-3) | 1.60e-2 (8.23e-3) | 9.12e-3 (2.39e-3) | 3.61e-3 (6.65e-3) | 1.34e-3 (1.78e-3) | |
| | | G-ISTA+StARS | 1.56e-1 (3.00e-2) | 1.02e-1 (1.47e-2) | 7.86e-2 (8.54e-3) | 5.82e-2 (1.01e-2) | 2.68e-2 (1.82e-3) | 1.23e-2 (5.95e-4) | 4.34e-3 (1.22e-4) |
| | | BIG&QUIC+BINCO | 4.33e-2 (1.04e-2) | 2.69e-2 (5.12e-3) | 1.60e-2 (8.23e-3) | | | | |
| | | BIG&QUIC+StARS | 1.56e-1 (3.00e-2) | 1.02e-1 (1.47e-2) | 7.86e-2 (8.54e-3) | 5.82e-2 (1.01e-2) | 2.68e-2 (1.82e-3) | 1.23e-2 (5.95e-4) | |
| | | TIGER | 3.01e-2 (5.86e-3) | 1.81e-2 (3.21e-3) | 1.25e-2 (1.44e-3) | 7.96e-3 (1.09e-3) | 2.34e-3 (2.59e-4) | | |
| | | BISN | **5.75e-3** (6.35e-3) | **3.54e-3** (1.00e-3) | **2.84e-3** (8.34e-4) | **1.48e-3** (2.93e-4) | **5.66e-4** (1.09e-4) | **2.77e-4** (3.62e-5) | **1.04e-4** (7.31e-6) |
| | NegLogLLH | G-ISTA+BINCO | 3.96e1 (3.04) | 5.68e1 (3.29) | 7.17e1 (9.55) | 8.52e1 (5.00) | 1.72e2 (3.89e1) | 3.59e2 (3.79e1) | |
| | | G-ISTA+StARS | 5.75e1 (2.35) | 8.32e1 (2.18) | 1.09e2 (3.65) | 1.33e2 (3.21) | 2.48e2 (5.37) | 4.63e2 (4.80) | 1.09e3 (1.36e1) |
| | | BIG&QUIC+BINCO | 3.96e1 (3.04) | 5.68e1 (3.29) | 7.17e1 (9.55) | | | | |
| | | BIG&QUIC+StARS | 5.75e1 (2.35) | 8.32e1 (2.18) | 1.09e2 (3.65) | 1.33e2 (3.21) | 2.48e2 (5.37) | 4.63e2 (4.80) | |
| | | TIGER | 7.46e1 (2.45) | 1.03e2 (3.00) | 1.34e2 (5.39) | 1.56e2 (4.19) | 2.59e2 (7.61) | | |
| | | BISN | **3.34e1**(3.23) | **4.851e1** (3.15) | **6.57e1** (4.92) | **8.31e1** (4.54) | **1.63e2** (6.23) | **3.31e2** (6.68) | **8.55e2** (1.32e1) |
| | Prmt No. | G-ISTA+BINCO | 7.32e2 (1.29e1) | 1.22e3 (1.85e1) | 1.83e3 (1.79e2) | 2.55e3 (3.88e1) | 5.24e3 (7.84e1) | 1.09e4 (8.08e1) | |
| | | G-ISTA+StARS | 1.32e3 (8.44e1) | 2.09e3 (8.11e1) | 2.93e3 (1.08e2) | 3.85e3 (8.06e1) | 8.33e3 (1.74e2) | 4.36e4 (3.38e2) | |
| | | BIG&QUIC+BINCO | 7.32e2 (1.29e1) | 1.22e3 (1.85e1) | 1.83e3 (1.79e2) | | | | |
| | | BIG&QUIC+StARS | 1.32e3 (8.44e1) | 2.09e3 (8.11e1) | 2.93e3 (1.08e2) | 3.85e3 (8.06e1) | 8.33e3 (1.74e2) | 4.36e4 (3.38e2) | |
| | | TIGER | 9.55e2 (1.26e1) | 1.54e3 (2.19e1) | 2.16e3 (2.19e1) | 2.79e3 (2.33e1) | 5.94e3 (6.77e1) | | |
| | | BISN | **1.08e3** (1.65e1) | **1.65e3** (3.82e1) | **2.24e3** (3.99e1) | **2.80e3** (4.53e1) | **5.64e3** (7.20e1) | **1.14e4** (1.26e2) | **2.90e4** (5.63e2) |
| | BIC Score | G-ISTA+BINCO | 6.83e4 (4.86e3) | 1.45e5 (7.79e3) | 2.43e5 (2.93e4) | 3.59e5 (1.99e4) | 1.42e6 (3.12e5) | 5.84e6 (6.06e5) | |
| | | G-ISTA+StARS | 1.01e5 (3.26e3) | 2.15e5 (4.72e3) | 3.71e5 (1.11e4) | 5.62e5 (1.24e4) | 2.05e6 (4.16e4) | 7.56e6 (7.60e4) | 4.41e7 (5.41e5) |
| | | BIG&QUIC+BINCO | 6.83e4 (4.86e3) | 1.45e5 (7.79e3) | 2.43e5 (2.93e4) | | | | |
| | | BIG&QUIC+StARS | 1.01e5 (3.26e3) | 2.15e5 (4.72e3) | 3.71e5 (1.11e4) | 5.62e5 (1.24e4) | 2.05e6 (4.16e4) | 7.56e6 (7.60e4) | |
| | | TIGER | 1.26e5 (3.96e3) | 2.60e5 (7.28e3) | 4.45e5 (2.16e3) | 6.46e5 (1.68e4) | 2.15e6 (6.14e4) | | |
| | | BISN | **6.07e4** (5.17e3) | **1.28e5** (7.59e3) | **2.27e5** (1.60e4) | **3.54e5** (1.82e4) | **1.35e6** (5.01e4) | **5.41e6** (1.07e5) | **3.45e7** (5.26e5) |
| Computational Time (s) | | G-ISTA+BINCO | 5.73e2 (1.10e2) | 1.73e3 (4.36e2) | 4.22e3 (1.02e3) | 6.44e3 (1.54e3) | 4.64e4 (8.39e3) | 4.30e5 (3.64e4) | |
| | | G-ISTA+StARS | 2.95e1 (7.23) | 8.63e1 (9.05) | 2.11e2 (2.86e1) | 4.23e2 (8.53e1) | 2.92e3 (1.76e2) | 2.73e4 (2.79e3) | 4.24e5 (7.40e4) |
| | | BIG&QUIC+BINCO | 5.44e3 (5.98e2) | 1.73e4 (2.11e3) | 4.28e4 (5.03e3) | | | | |
| | | BIG&QUIC+StARS | 3.48e2 (6.02e1) | 1.01e3 (8.01e2) | 2.73e3 (2.75e2) | 5.23e3 (7.09e2) | 3.79e4 (1.72e3) | 3.64e5 (3.67e4) | |
| | | TIGER | 1.72e3 (9.41e1) | 5.67e3 (2.67e2) | 1.44e4 (7.85e2) | 3.12e4 (1.87e3) | 3.00e5 (8.05e3) | | |
| | | BISN | **6.24** (9.72e-1) | **1.27e1** (7.32e-1) | **2.60e1** (1.04) | **4.77e1** (6.75) | **2.30e2** (7.16) | **1.09e3** (4.99e1) | **9.36e3** (3.04e2) |
| Memory Cost (MB) | | G-ISTA+BINCO | 5.47e1 (2.34e-1) | 1.19e2 (3.78e-1) | 2.08e2 (8..85e-1) | 3.20e2 (1.07) | 1.20e3 (2.91) | 4.49e3(5.33) | |
| | | G-ISTA+StARS | 3.20 (5.88e-3) | 7.15 (6.99e-3) | 1.27e1 (9.02e-3) | 1.98e1 (2.01e-2) | 7.88e1 (4.02e-2) | 3.14e2 (1.95e-1) | 2.02e3 (3.04e-1) |
| | | BIG&QUIC+BINCO | 5.57e1 (2.92e-1) | 1.21e2 (3.18e-1) | 2.11e2 (5.46e-1) | | | | |
| | | BIG&QUIC+StARS | 7.63 (6.93e-3) | 1.72e1 (1.58e-3) | 3.06e1 (2.41e-3) | 4.79e1 (7.30e-3) | 1.92e2 (9.43e-2) | | |
| | | TIGER | 4.54 (1.08e-3) | 1.02e1 (8.50e-4) | 1.82e1 (1.52e-3) | 2.85e1 (1.77e-3) | 1.14e2 (3.06e-3) | | |
| | | BISN | 6.29 (1.33e-3) | 1.42e1 (1.89e-3) | 2.53e1 (2.28e-3) | 3.94e1 (2.54e-3) | 1.56e2 (1.64e-2) | 6.11e2 (3.98e-2) | 3.27e3 (2.59e-1) |



(a) $n = 4p$.

(b) $n = 40p$.

Figure 3: Precision, recall, and $F_1$-score as a function of the penalty parameter $\lambda$ in the G-ISTA algorithm when $p = 1000$: (a) the data set used in Table 3 in which $n = 4p$; the black line denotes the penalty parameter $\lambda$ determined by StARS; (b) we further increase the sample size to $n = 40p$. BIG&QUIC has the same performance curves as G-ISTA for this dataset (not shown here).

recover the true graph given the limited number of observed samples $n = 4p$. In Figure 3(b), we further increase the sample size by a factor of 10. Although the highest $F_1$-score increases, it is still much lower than 0.99 resulting from the proposed BISN approach. Different from the frequentist methods that impose the same amount of penalty on all elements in $K$, the sparse-promoting penalties on the elements of $K$ in BISN can be different from each other, and they are learned adaptively from the data. As a result, in order to recover the true graph, BISN would require a much smaller sample size than these frequentist methods. Furthermore, due to the $\ell_1$-norm penalty in the objective function, the parameter estimation is biased towards zero as shown in Figure 2, and the likelihood of the data is not maximized. This explains the worst performance of G-ISTA+StARS and BIG&QUIC+StARS in terms of MSE compared with other methods.

In addition, it can be seen from Table 3 that BINCO (i.e., stability selection) helps to improve the performance of G-ISTA and BIG&QUIC. As pointed out in (Meinshausen and Bühlmann, 2010; Liu et al., 2010; Yu et al., 2012; Li et al., 2013), the stability (i.e., the existing frequency among the subsampled or bootstrapped sample sets) of the elements in the precision matrix provides a better distinction between true and false edges than the estimated elements themselves. Moreover, since we re-estimate the non-zero entries in the precision matrix via maximum likelihood after learning the structure, the performance of parameter estimation and model fitting also improves a lot. However, it is prohibitive to apply BINCO to high-dimensional graphical model selection, since BINCO is already quite time-consuming even for low-dimensional problems with a few hundred variables, even if it is coupled with BIG&QUIC.

It should be stressed that BIG&QUIC is not applicable to one million variables when coupled with regularization selection methods such as StARS and BINCO. All regularization

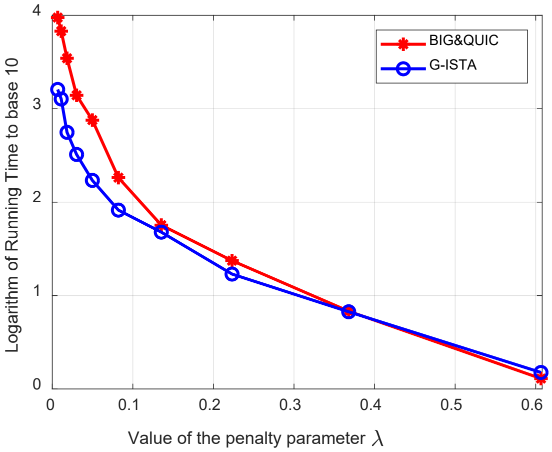


Figure 4: The computational time of BIG&QUIC and G-ISTA as a function of the penalty parameter $\lambda$ when $p = 2000$.

selection methods require testing small candidates of penalty parameters to guard against false negatives in the estimated edge set of the resulting graphical model. When running BIG&QUIC with small penalty parameters, the resulting precision matrix in most of the iterations would be dense, and so it is impossible to store the precision matrix as BIG&QUIC does in the memory when $p$ is large. Moreover, as mentioned in Section 1, the time complexity of the computational bottleneck of BIG&QUIC is $\mathcal{O}(pm)$, where $m$ is the number of non-zero entries in the precision matrix. When the penalty parameter is small, $m$ can easily grow quadratically with $p$. The resulting time complexity of BIG&QUIC is $\mathcal{O}(p^3)$. Figure 4 shows the running time of BIG&QUIC and G-ISTA as a function of the penalty parameter. We can see that the running time grows dramatically as $\lambda$ decreases, since the number of elements in $K$ increases. When coupling with the regularization selection methods, tests on small $\lambda$ would dominate the overall computational time. This explains the cubic increasing trend of BIG&QUIC w.r.t. $p$ in Figure 1(b). Additionally, it can be observed that BIG&QUIC is much slower than G-ISTA when $\lambda$ is small. Consequently, when coupling with StARS and BINCO, BIG&QUIC is slower than G-ISTA as shown in Table 3.

Now let us focus on the tuning-insensitive method TIGER. It produces the second best results for structure estimation. The parameter estimation is also more accurate than that of G-ISTA and BIG&QUIC. However, the time complexity of TIGER is $\mathcal{O}(\min(n, p)p^2)$. When $n$ is a linear function of $p$, the time complexity can be simplified as $\mathcal{O}(p^3)$. In comparison with BISN, the computational time of TIGER is at least two orders of magnitude larger, as shown in Figure 1. Furthermore, as demonstrated in Table 3, although the number of parameters given by TIGER is larger than that given by BINCO, the likelihood resulting from TIGER is smaller than that of BINCO. This can be explained by the fact that TIGER maximizes the pseudo likelihood instead of the true likelihood. Indeed, TIGER performs the worst in terms of NegLogLLH and BIC. By comparing all methods from the perspective of data fitting, we can tell that BISN fits the data well in an automated fashion. However,

Table 4: The graph recovery performance of the state-of-the-art methods and BISN on synthetic data averaged over 10 trials when the sample size $n$ decreases for $p = 1000$. The corresponding standard deviation is listed in the brackets.

| Sample Size $n$ | | | $n = 2p$ | $n = p$ | $n = p/2$ | $n = p/4$ | $n = p/8$ |
|---|---|---|---|---|---|---|---|
| Structure Recovery | Precision | BINCO | 0.92 (3.44e-3) | 0.94 (4.57e-3) | **0.96** (2.38e-3) | **0.98** (3.19e-3) | **0.98** (5.10e-3) |
| | | StARS | 0.48 (1.07e-2) | 0.36 (2.13e-2) | 0.23 (3.10e-3) | 0.09 (5.98e-3) | 0.05 (5.03e-4) |
| | | TIGER | 0.88 (3.81e-3) | 0.84 (7.28e-3) | 0.77 (8.36e-3) | 0.68 (7.87e-3) | 0.59 (1.18e-2) |
| | | BISN | **0.99** (8.50e-3) | **0.96** (3.73e-3) | 0.91 (4.73e-2) | 0.84 (3.31e-2) | 0.86 (2.48e-2) |
| | Recall | BINCO | 0.74 (6.52e-3) | 0.63 (1.21e-2) | 0.50 (1.09e-2) | 0.39 (9.42e-3) | 0.23 (7.32e-3) |
| | | StARS | 0.70 (1.48e-2) | 0.64 (2.78e-2) | 0.55 (1.08e-2) | **0.55** (1.92e-2) | **0.52** (9.01e-3) |
| | | TIGER | 0.87 (1.11e-2) | 0.70 (1.11e-2) | 0.54 (8.43e-3) | 0.44 (9.08e-3) | 0.37 (8.99e-3) |
| | | BISN | **0.99** (5.31e-3) | **0.89** (9.12e-3) | **0.57** (3.10e-2) | 0.40 (1.01e-2) | 0.28 (1.49e-2) |
| | $F_1$-score | BINCO | 0.82 (3.70e-3) | 0.76 (8.41e-3) | 0.66 (9.33e-3) | **0.56** (9.93e-3) | 0.37 (9.48e-3) |
| | | StARS | 0.57 (7.34e-3) | 0.46 (8.92e-3) | 0.33 (2.90e-3) | 0.16 (7.54e-3) | 0.09 (9.60e-4) |
| | | TIGER | 0.87 (6.77e-3) | 0.77 (8.44e-3) | 0.64 (7.90e-3) | 0.53 (8.75e-3) | **0.45** (9.42e-3) |
| | | BISN | **0.99** (2.37e-3) | **0.92** (4.74e-3) | **0.70** (1.57e-2) | 0.54 (7.83e-3) | 0.43 (1.51e-2) |
| Running Time (s) | | BINCO | 3.74e4 (3.47e3) | 5.24e4 (1.14e4) | 6.10e4 (6.11e3) | 7.59e4 (1.01e4) | 8.47e4 (2.52e4) |
| | | StARS | 1.67e3 (1.44e2) | 1.57e3 (1.74e2) | 1.24e3 (9.74e1) | 1.29e3 (1.37e2) | 1.58e3 (1.69e2) |
| | | TIGER | 3.13e5 (6.31e4) | 8.48e4 (1.82e4) | 5.14e4 (1.24e3) | 1.92e4 (1.18e3) | 8.01e3 (3.52e2) |
| | | BISN | **2.06e2** (1.84e1) | **3.03e2** (6.74) | **4.29e2** (9.06) | **4.79e2** (1.33e1) | **5.95e2** (5.22) |

for TIGER, BIG&QUIC, and G-ISTA, it is recommended to refit the estimated graph to the data after applying these algorithms.

We also summarize the memory cost of all algorithms at the bottom of Table 3. The theoretical space complexity of all these methods is $\mathcal{O}(p^2)$ and we can observe that the actual memory cost is approximately a quadratic function of the dimension $p$. Moreover, the memory cost of BISN is comparable to that of the state-of-the-art methods.

Next, we discuss the performance of the methods when the sample size decreases. In Table 4, we show the results for graph structure recovery. Note that the results from G-ISTA and BIG&QUIC are identical, therefore, we only present the results of G-ISTA. We can find that the performance of all methods deteriorates as the sample size decreases, as expected. Moreover, BISN achieves the best performance in terms of recall and $F_1$-score when $n \geq p/2$. Its performance is still comparable to the benchmark methods when $n < p/2$. Under this scenario all the methods fail to reliably recover the true graph. On the other hand, BISN takes the least amount of time to obtain the graphical models. Its computational time increases with the decrease of the sample size, probably because there is less evidence in data as sample size decreases and it becomes more difficult for BISN to separate the true edges from the false ones. The same phenomenon also occurs for BINCO. As opposed to BISN and BINCO, the running time of TIGER decreases as the sample size becomes smaller, since its time complexity is $\mathcal{O}(\min(n,p)p^2)$. However, its computational time is still at least one order of magnitude larger than that of BISN.

### 6.2. Stock Data

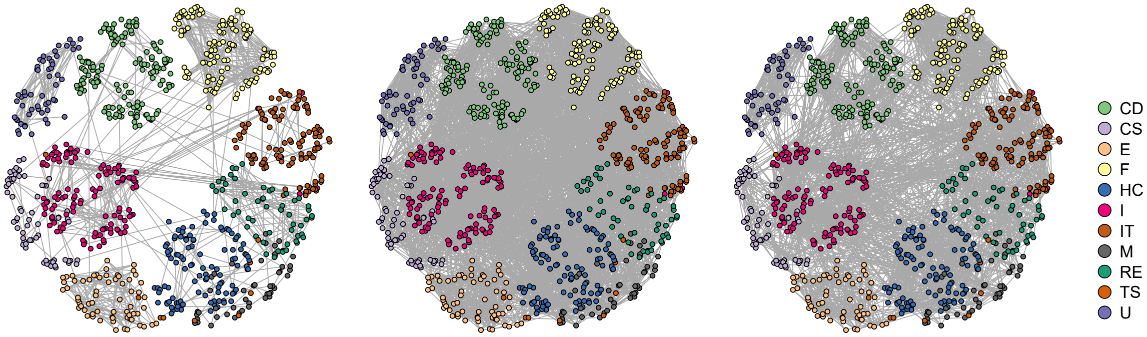
In this section, we analyze the daily stock returns data of the S&P500 companies during the 2008 financial crisis (from 2007 to 2011). We only consider 453 stocks, since the

Table 5: Quantitative comparison of BINCO, StARS, TIGER, and BISN for stock data

| Period | $n$ | $p$ | Methods | No. of Edges | Precision | | | | Running |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | BINCO | StARS | TIGER | BISN | Time (s) |
| Pre-Crisis<br>2006-2007 | 501 | 453 | BINCO | 635 | 1 | 1.00 | 0.92 | 0.44 | 6.74e4 |
| | | | StARS | 8487 | 0.07 | 1 | 0.43 | 0.04 | 1.71e3 |
| | | | TIGER | 4130 | 0.14 | 0.89 | 1 | 0.08 | 2.87e4 |
| | | | BISN | 322 | 0.87 | 0.99 | 0.99 | 1 | 3.21e2 |
| Crisis<br>2008-2009 | 505 | 453 | BINCO | 1015 | 1 | 1.00 | 0.84 | 0.75 | 7.61e4 |
| | | | StARS | 11595 | 0.09 | 1 | 0.29 | 0.14 | 2.20e3 |
| | | | TIGER | 3755 | 0.23 | 0.90 | 1 | 0.46 | 3.93e4 |
| | | | BISN | 2054 | 0.37 | 0.83 | 0.84 | 1 | 3.68e2 |
| Post-Crisis<br>2010-2011 | 504 | 453 | BINCO | 784 | 1 | 1.00 | 0.71 | 0.56 | 8.73e4 |
| | | | StARS | 10479 | 0.07 | 1 | 0.22 | 0.09 | 2.18e3 |
| | | | TIGER | 3849 | 0.14 | 0.61 | 1 | 0.32 | 4.58e4 |
| | | | BISN | 1306 | 0.34 | 0.78 | 0.93 | 1 | 3.63e2 |

data for the remaining stocks are missing in the first few years. The 2008 financial crisis is known to be the most severe financial crisis after the Great Depression of the 1930s. Research on financial networks for system risk modeling has surged in the aftermath of this financial crisis, since such networks can be exploited to analyze the interactions between financial institutions, to detect channels of risk contagion that can impair the stability of the entire system, and to further establish which institutions are more contagious or subject to contagion (Billio et al., 2012; Ahelegbey and Giudici, 2014; Barigozzi and Brownlees, 2019). Gaussian graphical models have been applied to infer financial networks in the fields of both machine learning (Choi et al., 2009; Chandrasekaran et al., 2012; Fan et al., 2016; Tarzanagh and Michailidis, 2018; Yang and Peng, 2019; Yu et al., 2019) and finance (Cont et al., 2010; Ahelegbey and Giudici, 2014; Ahelegbey et al., 2016; Hashem and Giudici, 2016; Cerchiello et al., 2017; Bianchi et al., 2019).

In order to check how the financial network changes during the financial crisis, we partition the data into three parts: pre-crisis (2006-2007), crisis (2008-2009), and post-crisis (2010-2011), according to the Federal Reserve Bank of St. Louis' Financial Crisis Timeline. We then apply the four methods BINCO, StARS, TIGER, and BISN to infer financial networks for all the three parts of data. Since the ground truth network structure is not available, we "cross validate" the results of the four methods. The results are listed in Table 5. For entry $(i, j)$ in the columns of precision, we compute the precision by regarding the graph $\mathcal{G}_i$ resulting from method $i$ as the estimated graph and the graph $\mathcal{G}_j$ resulting from method $j$ as the true graph. According to the definition, the precision can be calculated as the ratio between the number of common edges in $\mathcal{G}_i$ and $\mathcal{G}_j$ and the number of edges in $\mathcal{G}_i$. In other words, it can be interpreted as the proportion of the edges in the graph estimated by method $i$ that can also be detected by method $j$. As an example, we can find that for the pre-crisis data, the number of edges in the estimated graphs increases in the order of BISN, BINCO, TIGER, and StARS. Moreover, we can observe that 87% of edges in the BISN graph are also detected by BINCO and 99% of edges in the BISN graph are identified by TIGER and StARS. Although these methods yield graphs with different sparsity, the denser graph typically contains most of the edges in the sparser graph, indicating that these



(a) Pre-Crisis (322 edges)

(b) Crisis (2054 edges)

(c) Post-Crisis (1306 edges)

Figure 5: Financial networks resulting from BISN before, during, and after the 2008 financial crisis.

results are consistent with each other. This conclusion also holds for the crisis and post-crisis data in Table 5. Again, we underline that BISN achieves comparable performance to the state-of-the-art methods with the least amount of computational time.

Now let us focus on the financial networks resulting from BISN as shown in Fig. 5. The 453 stocks can be categorized into 11 sectors according to the Global Industry Classification Standard (GICS), namely, Consumer Discretionary (CD), Consumer Staples (CS), Energy (E), Financials (F), Health Care (HC), Industrials (I), Information Technology (IT) Materials (M), Real Estate (RE), Telecommunication Services (TS), and Utilities (U). By comparing Fig. 5(a) to Fig. 5(b), we can see that the financial network becomes much denser during the financial crisis. After the financial crisis (see Fig. 5(c), the network is sparser but is still denser than the one before the financial crisis. Similar phenomena are observed in Yang and Peng (2019) and Bianchi et al. (2019) where time-varying graphical models are applied to analyze the stock returns data of 283 stocks and S&P 100 respectively. Due to risk contagion, all companies are exposed to the economy turndown and market unrest during the crisis period, leading to similar stock price movement and business response to the system risk (Bullard et al., 2009). This explains the increased number of connections between the stocks. In the post-crisis period, interactions between stocks decreases but stocks still have more connections than in the pre-crisis period, suggesting the significant evolution of financial structure due to the crisis (Yang and Peng, 2019).

We further compute the total number of edges within each sector and between different sectors as shown in the last column of Table 6. It can be observed that the inner-sector connections dominate the total number of connections. In other words, stocks from the same sector are clustered together by BISN, especially when the network is sparse (cf. Fig. 5(a)). Indeed, companies in the same GICS sector are supposed to have more connections. However, the ratios between the number of the inner-sector edges and the total number of edges before, during, and after the financial crisis are respectively 0.94, 0.57, and 0.73. The proportion of the inner-sector edges decreases during the financial crisis, due to the greatly increased number of the inter-sector edges. Such patterns are also found in Yang and Peng (2019). It

Table 6: Number edges within the sector and connected from other sectors (i.e., inner and inter-sector edges) for each of the 11 sectors in the BISN graphs.

| Sector | | CD | CS | E | F | HC | I | IT | M | RE | TS | U | All Sectors |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| No. of Nodes | | 72 | 32 | 32 | 59 | 54 | 62 | 58 | 23 | 28 | 6 | 27 | 453 |
| Pre-Crisis 2006-2007 | No. of Inner-Sector Edges | 14 | 5 | 57 | 77 | 11 | 38 | 12 | 7 | 47 | 1 | 35 | 304 |
| | No. of Inter-Sector Edges | 5 | 5 | 2 | 5 | 5 | 5 | 2 | 4 | 1 | 1 | 1 | 18 |
| | Total No. of Edges | 19 | 10 | 59 | 82 | 16 | 43 | 14 | 11 | 48 | 2 | 36 | 322 |
| Crisis 2008-2009 | No. of Inner-Sector Edges | 138 | 48 | 122 | 203 | 117 | 153 | 124 | 44 | 102 | 7 | 106 | 1164 |
| | No. of Inter-Sector Edges | 239 | 124 | 86 | 289 | 221 | 288 | 241 | 141 | 72 | 25 | 54 | 890 |
| | Total No. of Edges | 377 | 172 | 208 | 492 | 338 | 441 | 365 | 185 | 174 | 32 | 160 | 2054 |
| Post-Crisis 2010-2011 | No. of Inner-Sector Edges | 110 | 34 | 88 | 176 | 107 | 133 | 102 | 27 | 83 | 3 | 86 | 949 |
| | No. of Inter-Sector Edges | 84 | 58 | 50 | 72 | 82 | 156 | 112 | 5 | 11 | 19 | 16 | 357 |
| | Total No. of Edges | 194 | 92 | 138 | 248 | 189 | 289 | 214 | 81 | 94 | 22 | 102 | 1306 |

Table 7: Increased percentage of the inter-sector edges for the 11 sectors.

| Sector | IT | RE | F | I | U | CD | HC | E | M | TS | CS |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Increased Percentage of Inter-sector Edges (%) | 119.5 | 71 | 56.8 | 56.6 | 53 | 46.8 | 43.2 | 42 | 34.25 | 24 | 23.8 |
| No. of Nodes | 58 | 28 | 59 | 62 | 62 | 72 | 54 | 32 | 23 | 6 | 32 |

seems that the financial crisis has a larger influence on the inter-sector connections. On the other hand, we are also interested in the number of the inter-sector connections, since they provide us a measure of how vulnerable or contagious one sector is in the entire financial system. According to the theory of system risk, financial institutions with more connections are more sensitive to the financial crisis, or conversely, their failure is more likely to cause the breakdown of the entire system (Billio et al., 2012; Ahelegbey and Giudici, 2014; Barigozzi and Brownlees, 2019). To this end, we compute the number of edges between one sector and the other sectors excluding this one for each of the 11 sectors in Table 6. We further calculate the increased percentage of the inter-sector edges for each sector during the financial crisis by comparing the seventh row with the fourth row in Table 6, and then sort all sectors in the descending order of the increased percentage in Table 7. As demonstrated in the table, the increased percentage is typically larger for sectors with more stocks. There are two exceptions though, i.e., RE (Real Estate) and CS (Consumer Staples). The increased percentage of the inter-sector edges for RE is large, whereas the number of nodes in the RE sector is small, indicating that RE is more risk contagious in the system during the financial crisis. In fact, RE is known as the trigger of the 2008 financial crisis (Williams, 2010). On the other hand, although there is a relatively large number of stocks in the CS sector, its increased percentage is small. Note that the companies in the CS sector typically produce or distribute goods that people buy out of necessity regardless of the economic conditions (Asinas, 2018). Thus, this sector is more robust to the financial crisis.

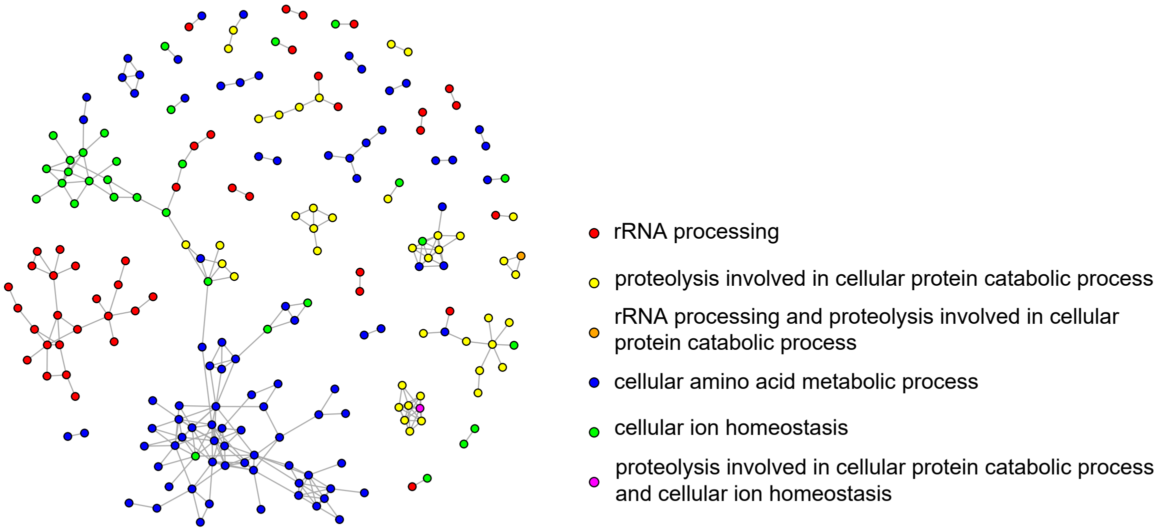

● rRNA processing
● proteolysis involved in cellular protein catabolic process
● rRNA processing and proteolysis involved in cellular protein catabolic process
● cellular amino acid metabolic process
● cellular ion homeostasis
● proteolysis involved in cellular protein catabolic process and cellular ion homeostasis

Figure 6: Gene regulatory network among genes associated with four biological processes: rRNA processing, proteolysis involved in cellular protein catabolic process, cellular amino acid metabolic process, and cellular ion homeostasis.

### 6.3. Gene Data

In this section, we exploit BISN to learn gene regulatory networks form the Rosetta Inpharmatics Compendium of gene expression profiles (Hughes et al., 2000). The data set contains the 300 expression profiles of the yeast Saccharomyces cerevisiae for 6316 genes. As mentioned in Section 1, reliable estimation of gene regulatory networks plays an indispensable role in systematically understanding the molecular mechanism, providing meaningful insights into the mechanism of diseases that occur when cellular processes are dysregulated, and further finding the possible therapeutic targets for the diseases (Su et al., 2018; Jia and Liang, 2018; Zhao and Duan, 2019). Due to the high dimensionality and complexity of the gene data, inference of gene regulatory networks typically resort to statistical methods. The Gaussian graphical model has proven itself as a useful tool in this field (Banerjee et al., 2008; Fitch and Jones, 2009; Rolfs et al., 2012; Hsieh et al., 2014; Chun et al., 2015; Fan et al., 2016; Tarzanagh and Michailidis, 2018; Deng et al., 2018; Zhao and Duan, 2019).

We notice that the three benchmark methods, BINCO, StARS, and TIGER, can barely scale up to 5000 thousand variables. Therefore, we only present the gene regulatory network yielded by BISN. Also, 1.61% data are missing at random, and so we infer the variational distribution of these missing data along with that of the precision matrix. The resulting network only has 4306 edges, which is quite sparse, since the sample size $n$ is much smaller than the dimension $p$ in this data set. According to the gene ontology database (http://www.yeastgenome.org), different genes are involved in different biological processes. Note that some genes have more than one functional category. Here we choose four biological processes with a small number of shared genes between each other, that is, rRNA processing, proteolysis involved in cellular protein catabolic process, cellular amino acid metabolic process, and cellular ion homeostasis. The subgraph corresponding to the genes involved in these four processes is depicted in Fig. 6. We can see that genes with the same annotations are clustered together in an automatic fashion. In addition, the network also shows how genes with different annotations are connected. Such edges represent crosstalks between genes. Crosstalks are known to happen among the genes of the yeast Saccharomyces cerevisiae, and are essential to understanding how a cell integrate internal and external stimuli and adjust cellular metabolism, growth and proliferation (Simpson-Lavy et al., 2015; Shashkova et al., 2015).

### 6.4. Functional Magnetic Resonance Imaging (fMRI) data

In this section, we apply BISN to infer functional brain networks based on fMRI data. Recent studies in neural science have shown that functional brain networks typically undergo changes during different cognitive activities (Liang et al., 2016) and development (Cao et al., 2016), as well as in neurological and mental disorders, such as epileptic seizures (Evangelisti et al., 2018), Alzheimer's disease (Schumacher et al., 2018), autism spectrum disorder (Keown et al., 2017), and hyperactivity disorder (van den Heuvel et al., 2017). Learning and understanding the brain networks and their changes can shed light upon the biological mechanisms underlying human cognition, as well as health and disease (Karwowski et al., 2019). These networks can also help to differentiate between different cognitive tasks and between patients and healthy people. Gaussian graphical models are of widespread utility for inferring functional brain networks (Dauwels et al., 2012; Xu and Lindquist, 2015; Ortiz et al., 2015; Belilovsky et al., 2016; Yu and Dauwels, 2016, 2018; Zhang et al., 2018a, 2019), due to their simplicity and scalability.

Here we consider the fMRI-based mind-state classification problem described in Mitchell et al. (2004). The data set consists of 40 experiments for each of the 6 subjects in half of which the subject is given a sentence and in the other half a picture. In each experiment, there are 16 fMRI images recorded when the subject is looking at the sentence or the picture, each with around 5000 voxels. These voxels can be divided into 24 anatomical regions of interest (ROIs). They are calcarine sulcus (CALC), dorsolateral prefrontal cortex - left & right (LDLPFC, RDLPFC), frontal eye fields left & right (LFEF, RFEF), inferior parietal lobule left & right (LIPL, RIPL), intraparietal sulcus left & right (LIPS, RIPS), opercularis left & right (LOPER, ROPER), posterior precentral sulcus left & right (LPPREC, RPPREC), supramarginal gyrus left & right (LSGA, RSGA), superior parietal lobule left & right (LSPL, RSPL), temporal lobe left & right (LT, RT), triangularis left & right (LTRIA, RTRIA), supplementary motor areas (SMA), inferior temporal lobule left & right (LIT, RIT). The objective is to differentiate whether a subject is looking at the sentence or the picture given the fMRI images.

We first combine the fMRI images respectively for the sentence and the picture stimulus for each subject, and correspondingly apply BISN to infer the functional brain network for all observed voxels. The results are summarized in Table 8. We can find that for all subjects and for both cognitive tasks, voxels from the same ROIs have more connections than those from different ROIs, since the neurons within each ROI are supposed to work together more closely. We then count the number of different edges inside each ROI and between every

Table 8: Number of inner and inter-region edges for the 6 subjects when the subject is looking at a sentence and a picture.

| Subject ID | 04799 | 04820 | 04847 | 05675 | 05680 | 05710 |
|---|---|---|---|---|---|---|
| No. of Voxels | 4949 | 5015 | 4698 | 5135 | 5062 | 4634 |
| No. of Inner-region Edges (Sentence) | 5746 | 4088 | 4550 | 4550 | 3622 | 4229 |
| No. of Inter-region Edges (Sentence) | 1634 | 660 | 1940 | 671 | 399 | 836 |
| No. of Inner-region Edges (Picture) | 6048 | 4639 | 5221 | 4829 | 4089 | 4029 |
| No. of Inter-region Edges (Picture) | 1755 | 736 | 2113 | 736 | 471 | 823 |

Table 9: 10 Regions with the largest number of different edges for each of the 6 subjects. (LDLPFC, RLDPFC) denotes the connectivity between LDLPFC and RLDPFC.

| Subject ID | 10 Regions Sorted in Descending Order of the No. of Different Edges |
|---|---|
| 04799 | LDLPFC, RDLPFC, LIT, RT, LT, CALC, RIPL, RIT, LIPL, (LDLPFC, RLDPFC) |
| 04820 | LT, LDLPFC, RT, CALC, RDLPFC, LSPL, RSPL, RTRIA, LIT, SMA |
| 04847 | LDLPFC, SMA, RDLPFC, CALC, LT, LSPL, LIT, RT, RIT, LIPS |
| 05675 | LT, LDLPFC, CALC, RT, RDLPFC, LSPL, LIPL, RIT, RIPL, RSPL |
| 05680 | LDLPFC, LT, CALC, RDLPFC, RT, LSPL, LIPL, ROPER, LOPER, RIPS |
| 05710 | LDLPFC, LT, RDLPFC, RT, RIT, LIPL, RIPL, CALC, (LDLPFC, RLDPFC), LIT |

pair of ROIs, and list the 10 regions with the largest number of different edges for each subject in Table 9. It can be observed that the commonly chosen regions for all subjects are CALC, LDLPFC, LT, RDLPFC, and RT. In Do and Yang (2014), a Gaussian Naive Bayes (GNB) classifier is trained based on the most active voxels from each ROI, and 7 regions that produce the highest classification accuracy are selected. They are CALC, LDLPFC, LIPL, LIPS, LOPER, LT, and LTRIA. Three out of the five regions selected by BISN are also selected in Do and Yang (2014). Interestingly, besides the left part of the dorsolateral prefrontal cortex and the temporal lobe (i.e., LDLPFC and LT), BISN also includes the right part of these two regions, RDLPFC and RT. In neural science, there exists evidence showing that RDLPFC is involved in visual working memory (Wang et al., 2018), while RL contributes to visual signal processing (Doyon and Milner, 1991; Milner, 2003). This may explain why the interactions in these two regions yielded by BISN are quite different for the two cognitive tasks.

Next, we employ BISN to learn a brain network for each of the 40 experiments and for each of the five selected regions individually, and use the network structure to train a random forest (RF) classifier in order to distinguish between the sentence and the picture stimulus. In other words, the input to the classifier is the zero pattern of the BISN precision matrices. We apply leave-one-out cross validation to test the performance of the classifier based on graph structure and show the resulting classification accuracy in the second row in Table 10. Gaussian graphical models have been applied to classification for three subjects in

Table 10: Classification accuracy resulting from different methods.

| Subject ID | 04799 | 04820 | 04847 | 05675 | 05680 | 05710 |
|---|---|---|---|---|---|---|
| RF Classifier based on brain connectivity inferred by BISN | 92.5% | 95% | 100% | 92.5% | 97.5% | 90% |
| Classifier based on likelihood (Rish and Grabarnik, 2014) | N.A. | 95% | 95% | N.A. | 95% | N.A. |
| SVM Classifier based on voxel values (Rish and Grabarnik, 2014) | N.A. | 90% | 97.5% | N.A. | 87.5% | N.A. |
| GNB Classifier based on voxel values (Do and Yang, 2014) | 92.5% | 97.5% | 100% | 98.75% | 95% | 95% |

this data set in Rish and Grabarnik (2014). Up to 300 voxels from the 7 regions selected in Do and Yang (2014) that have the highest discriminative ability are chosen. The graphical models for both stimuli are then estimated from the training data by solving the penalized maximum likelihood problem (40) using the frequentist method SINCO (Scheinberg and Rish, 2010). The penalty parameter is selected manually. The testing data is classified into the class with a larger likelihood. As a benchmark, the support vector machine (SVM) is also applied for classification in Rish and Grabarnik (2014). The results from the two classifiers in Rish and Grabarnik (2014) and the GNB classifier in Do and Yang (2014) are listed in the bottom rows in Table 10. We can tell from the table that the accuracy given by the proposed method is comparable to the three benchmark methods. Different from the other three methods, we do not select voxels that are the most active or most discriminative from the selected regions as in Do and Yang (2014) and Rish and Grabarnik (2014). Instead, we use all voxels and then BISN learns sparse networks between them. The high classification accuracy suggests that the brain network resulting from BISN also provides an effective tool for determining the mind state.

## 7. Conclusion and Future Work

We introduced BISN for Gaussian graphical model selection, which is tuning-free and has a low time complexity that is quadratic in dimension. Numerical results show that BISN achieves comparable or better performance than the state-of-the-art methods, within a computational time that is several orders of magnitude smaller for large-scale problems. Moreover, BISN can be extended to handle missing data and latent variables in a straightforward manner.

Since BISN copes with the LDL decomposition of the precision matrix, the $L$ matrix can be dense even if the true precision matrix is sparse. Moreover, BISN is a Bayesian method, and therefore, we need to store the mean and the variance of every element in $L$, regardless of whether the true value is zero or not. Indeed, given 32 GB of memory, BISN can tackle around 15,000 variables, whereas BIG&QUIC can deal with one million variables if the penalty parameter is known and the graph is very sparse (Hsieh et al., 2013). In other words, if the prior information or expert knowledge is available of the penalty parameter in the tuning-sensitive methods (e.g., G-ISTA and BIG&QUIC), these methods can be faster and more memory efficient than BISN, since they only store a sparse precision matrix in the memory and take advantage of the sparsity to simplify the learning process. On the other hand, when such information is unknown, which is often the case in practice, BISN provides an effective and efficient tool to learn the structure of the graphical model from data. In future work, we intend to make use of the sparsity of the estimated precision matrix to further reduce the time and space complexity of BISN.

In addition, BISN can only deal with Gaussian distributed variables so far. In future work, we plan to extend BISN for non-Gaussian data by coupling it with copulas (Liu et al., 2009; Yu et al., 2012; Dauwels et al., 2013).

We also emphasize that the KL proximal DRSG algorithm proposed in this paper can be applied to general finite sum problems in which the objective function can be decomposed as the sum of one smooth term that can be nonconvex and one convex term that can be nonsmooth. Furthermore, the KL divergence can be replaced by the more general Bregman divergence in the proof. Such finite sum problems arise frequently in the field of machine learning, cf. (Reddi et al., 2016). KL proximal DRSG offers a computationally attractive alternative to solve these problems when the number of terms in the finite sum is large. In future work, it is interesting to prove whether the convergence rate of DRSG can be further improved when applied to (strongly) convex and smooth problems.

## Acknowledgments

We are grateful for the constructive comments from Prof. Peter Spirtes and three anonymous reviewers. We would also like to acknowledge the support for this project from MOE (Singapore) project 2017-T2-2-126 and the NAM Advanced Biomedical Imaging Program (FY2016) between Nanyang Technological University, Singapore and Medical University of Vienna, Austria. The MATLAB C++ MEX code of BISN is available at https://github.com/fhlyhv/BISN. The major part of the code is implemented using the Armadillo C++ template library (Sanderson and Curtin, 2016)

## Appendix A. Derivation of the Jacobian Matrix and the Absolute Value of Its Determinant

The Jacobian matrix $J$ contains the partial derivative of all lower triangular entries in $K$ with regard to $L_{jk}$ and $D_{jj}$ for $j = 1, \cdots, p$ and $k < j$, which can be written as:

$$J = \begin{bmatrix} \frac{\partial K_{11}}{\partial D_{11}} & \frac{\partial K_{21}}{\partial D_{11}} & \frac{\partial K_{22}}{\partial D_{11}} & \frac{\partial K_{31}}{\partial D_{11}} & \cdots & \frac{\partial K_{p-1,p}}{\partial D_{11}} & \frac{\partial K_{pp}}{\partial D_{11}} \\ \frac{\partial K_{11}}{\partial L_{21}} & \frac{\partial K_{21}}{\partial L_{21}} & \frac{\partial K_{22}}{\partial L_{21}} & \frac{\partial K_{31}}{\partial L_{21}} & \cdots & \frac{\partial K_{p-1,p}}{\partial L_{21}} & \frac{\partial K_{pp}}{\partial L_{21}} \\ \frac{\partial K_{11}}{\partial D_{22}} & \frac{\partial K_{21}}{\partial D_{22}} & \frac{\partial K_{22}}{\partial D_{22}} & \frac{\partial K_{31}}{\partial D_{22}} & \cdots & \frac{\partial K_{p-1,p}}{\partial D_{22}} & \frac{\partial K_{pp}}{\partial D_{22}} \\ \frac{\partial K_{11}}{\partial L_{31}} & \frac{\partial K_{21}}{\partial L_{31}} & \frac{\partial K_{22}}{\partial L_{31}} & \frac{\partial K_{31}}{\partial L_{31}} & \cdots & \frac{\partial K_{p-1,p}}{\partial L_{31}} & \frac{\partial K_{pp}}{\partial L_{31}} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial K_{11}}{\partial L_{p-1,p}} & \frac{\partial K_{21}}{\partial L_{p-1,p}} & \frac{\partial K_{22}}{\partial L_{p-1,p}} & \frac{\partial K_{31}}{\partial L_{p-1,p}} & \cdots & \frac{\partial K_{p-1,p}}{\partial L_{p-1,p}} & \frac{\partial K_{pp}}{\partial L_{p-1,p}} \\ \frac{\partial K_{11}}{\partial D_{pp}} & \frac{\partial K_{21}}{\partial D_{pp}} & \frac{\partial K_{22}}{\partial D_{pp}} & \frac{\partial K_{31}}{\partial D_{pp}} & \cdots & \frac{\partial K_{p-1,p}}{\partial D_{pp}} & \frac{\partial K_{pp}}{\partial D_{pp}} \end{bmatrix}. \tag{72}$$

In the above expression, the indices of both the denominators and nominators in the partial derivatives follow the column-major order of non-zero entries in a lower triangular matrix. Note that $L_{jj} = 1$ for all $j$ are constant and $D_{jj}$ is the argument we focus on.

Given that $K = LDL^T$, we can obtain that for the lower triangular off-diagonal entries in $J$:

$$\frac{\partial K_{jk}}{\partial L_{ab}} = 0 \quad \forall a \neq j \text{ or } b > k, \tag{73}$$

$$\frac{\partial K_{jk}}{\partial D_{aa}} = 0 \quad \forall a > k. \tag{74}$$

As a result, $J$ is an upper triangular matrix, and therefore, its determinant equals the product of its diagonal entries. For the diagonal entries of $J$, we have:

$$\frac{\partial K_{jk}}{\partial L_{jk}} = D_{kk}, \tag{75}$$

$$\frac{\partial K_{jj}}{\partial D_{jj}} = 1. \tag{76}$$

Taken together, the absolute value of the determinant is:

$$|\det(J)| = \prod_{j=1}^{p} D_{jj}^{p-j}. \tag{77}$$

Note that the Jacobian matrix can also be formulated by permuting columns and rows of $J$ in (72) simultaneously, but permutation does not change the absolute value of the determinant.

## Appendix B. Derivation of the Closed-Form Expression of $\mathcal{L}_1$

As mentioned in the main body of the paper, the original Bayesian model can be factorized as (13):

$$p(\boldsymbol{x}^{\{1:n\}}, L, D, \boldsymbol{\lambda}, \omega) = p(\boldsymbol{x}^{\{1:n\}}|L, D)p(L, D|\boldsymbol{\lambda}, \omega)p(\boldsymbol{\lambda})p(\omega) \tag{78}$$

$$= \prod_{i=1}^{n} p(\boldsymbol{x}^{\{i\}}|L, D)|\det(J)| \prod_{j=1}^{p} \prod_{k=j+1}^{p} \Big[p(L_{j,:}DL_{k,:}^T|\lambda_{jk}, \omega)p(\lambda_{jk})\Big] \cdot p(\omega), \tag{79}$$

where

$$p(\boldsymbol{x}^{\{i\}}|L, D) \propto \exp\Big(-\frac{1}{2}\boldsymbol{x}^{\{i\}^T} LDL^T \boldsymbol{x}^{\{i\}}\Big), \tag{80}$$

$$p(L_{j,:}DL_{k,:}^T|\lambda_{jk}, \omega) \propto \sqrt{\omega\lambda_{jk}} \exp\Big(-\frac{1}{2}\omega\lambda_{jk}\big(p(L_{j,:}DL_{k,:}^T)^2\Big), \tag{81}$$

$$p(\lambda_{jk}) = \frac{1}{\pi}\lambda_{jk}^{-\frac{1}{2}}\big(\lambda_{jk} + 1\big)^{-1}, \quad \forall \lambda_{jk} > 0, \tag{82}$$

$$p(\omega) \propto \frac{1}{w}. \tag{83}$$

The variational distributions are specified as in Eq. (16)-(19).

The expectation $\mathcal{L}_1$ can be decomposed as:

$$\begin{aligned}\mathcal{L}_1 =& \mathbb{E}_q\big[\log p(\boldsymbol{x}^{\{1:n\}}, L, D, \boldsymbol{\lambda}, \omega)\big]\\ =& \mathbb{E}_q\big[\log p(\boldsymbol{x}^{\{1:n\}}|L, D)\big] + \mathbb{E}_q\big[\log p(L, D|\boldsymbol{z}, \lambda_0, \omega)\big] + \mathbb{E}_q\big[\log p(\boldsymbol{\lambda})\big] + \mathbb{E}_q\big[\log p(\omega)\big].\end{aligned} \tag{84}$$

We then turn our attention to each term in $\mathcal{L}_1$ (84).

$$\mathbb{E}_q\Bigg[\log p(\boldsymbol{x}^{\{1:n\}}|L, D)\Bigg] = \frac{n}{2}\sum_{i=1}^{n}\langle\log D_{jj}\rangle - \frac{1}{2}\langle\sum_{i=1}^{n}\boldsymbol{x}^{\{i\}^T}LDL^T\boldsymbol{x}^{\{i\}}\rangle + c \tag{85}$$

$$= \frac{n}{2}\sum_{i=1}^{n}\langle\log D_{jj}\rangle - \frac{n}{2}\operatorname{tr}(M_L M_D M_L^T S) - \frac{n}{2}\operatorname{diag}(S)^T V_L M_D \mathbf{1} + c, \tag{86}$$

where $c$ summarizes all irrelevant constants and

$$\langle\log D_{jj}\rangle = \psi(\alpha_j) - \log(\beta_j), \tag{87}$$

$$M_{Ljk} = \langle L_{jk}\rangle = \frac{h_{jk}}{\zeta_{jk}}, \tag{88}$$

$$V_{Ljk} = \langle L_{jk}^2\rangle - \langle L_{jk}\rangle^2 = \frac{1}{\zeta_{jk}}, \tag{89}$$

$$M_{Djj} = \langle D_{jj}\rangle = \frac{\alpha_j}{\beta_j}, \tag{90}$$

$$V_{Djj} = \langle D_{jj}^2\rangle - \langle D_{jj}\rangle^2 = \frac{\alpha_j}{\beta_j^2}. \tag{91}$$

Next, for $p(L_{j,:}DL_{k,:}^T|\lambda_{jk}, \omega)$,

$$\begin{aligned}\mathbb{E}_q[\log p(L, D|\boldsymbol{z}, \lambda_0, \omega)] =& \sum_{j=1}^{p}(p-j)\langle\log D_{jj}\rangle + \frac{p(p-1)}{4}\langle\log\omega\rangle\\ &+ \frac{1}{2}\sum_{j=1}^{p}\sum_{k=j+1}^{p}\Big[\langle\log\lambda_{jk}\rangle - \langle\omega\rangle\langle\lambda_{jk}\rangle\langle(LDL^T)\circ(LDL^T)\rangle_{jk}\Big] + c,\end{aligned} \tag{92}$$

where $c$ summarizes all irrelevant constants, and

$$\langle\lambda_{jk}\rangle = \frac{1}{d_{jk}\exp(d_{jk})E_1(d_{jk})} - 1, \tag{93}$$

$$\langle\omega\rangle = \frac{a}{b}. \tag{94}$$

We next focus on the term $\langle(LDL^T)\circ(LDL^T)\rangle_{jk}$. According to the properties of second-order moments,

$$\langle(LDL^T)\circ(LDL^T)\rangle_{jk} = \langle LDL\rangle_{jk}^2 + \mathbb{V}[LDL]_{jk} = [M_L M_D M_L]_{jk}^2 + \mathbb{V}[LDL]_{jk}, \tag{95}$$

where $\mathbb{V}[LDL]_{jk}$ represents the variance of $[LDL]_{jk}$. Note that $[LDL^T]_{jk} = \sum_{i=1}^p D_{ii}L_{ji}L_{ik}$ and we have assumed that $L_{jk}$ and $D_{jj}$ are independent for all $j$ and $k$ in the variational distribution. Thus, the off-diagonal elements in the product $LDL^T$ are given by the sum of the product of independence variables. Moreover, for two independent variables $X$ and $Y$, we have

$$\mathbb{V}[XY] = \langle X\rangle^2\mathbb{V}[Y] + \mathbb{V}[X]\langle Y\rangle^2 + \mathbb{V}[X]\mathbb{V}[Y]. \tag{96}$$

It follows from the above equality that:

$$\begin{aligned}\mathbb{V}[LDL]_{jk} &= \mathbb{V}[\sum_{j=1}^p D_{jj}L_{:,j}L_{:,j}^T]_{jk} = \sum_{j=1}^p \mathbb{V}[D_{jj}L_{:,j}L_{:,j}^T]_{jk}\\ &= \sum_{j=1}^p \Big\{\langle D_{jj}\rangle^2\mathbb{V}[L_{:,j}L_{:,j}^T]_{jk} + \mathbb{V}[D_{jj}]\langle L_{:,j}L_{:,j}^T\rangle_{jk}^2 + \mathbb{V}[D_{jj}]\mathbb{V}[L_{:,j}L_{:,j}^T]_{jk}\Big\},\end{aligned} \tag{97}$$

where $\mathbb{V}[L_{:,j}L_{:,j}^T]_{jk}$ can be further expanded as:

$$\begin{aligned}\mathbb{V}[L_{:,j}L_{:,j}^T]_{jk} = &\Big\{(\langle L_{:,j}\rangle \circ \langle L_{:,j}\rangle)\mathbb{V}[L_{:,j}]^T + \mathbb{V}[L_{:,j}](\langle L_{:,j}\rangle \circ \langle L_{:,j}\rangle)^T\\ &+ \mathbb{V}[L_{:,j}]\mathbb{V}[L_{:,j}]^T\Big\}_{jk}.\end{aligned} \tag{98}$$

As a result,

$$\begin{aligned}\langle(LDL^T)\circ(LDL^T)\rangle_{jk} = &\Big[(M_LM_DM_L^T)\circ(M_LM_DM_L^T) + (M_L\circ M_L)(M_D\circ M_D + V_D)V_L^T\\ &+ V_L(M_D\circ M_D + V_D)(M_L\circ M_L)^T + V_L(M_D\circ M_D+V_D)V_L^T\\ &+ (M_L\circ M_L)V_D(M_L\circ M_L)^T\Big]_{jk}.\end{aligned} \tag{99}$$

The expectation $\mathbb{E}_q[\log p(L_{j,:}DL_{k,:}^T|\lambda_{jk},\omega)]$ can be expressed as:

$$\begin{aligned}&\mathbb{E}_q[\log p(L,D|\boldsymbol{z},\omega,\lambda_0)]\\ =&\sum_{j=1}^p(p-j)\langle\log D_{jj}\rangle + \frac{p(p-1)}{4}\langle\log\omega\rangle + \frac{1}{2}\sum_{j=1}^p\sum_{k=j+1}^p\langle\log\lambda_{jk}\rangle\\ &-\frac{1}{4}\operatorname{tr}\Big\{\Lambda\big[(M_LM_DM_L^T)\circ(M_LM_DM_L^T) + (M_L\circ M_L)(M_D\circ M_D+V_D)V_L^T\\ &+V_L(M_D\circ M_D+V_D)(M_L\circ M_L)^T + V_L(M_D\circ M_D+V_D)V_L^T\\ &+(M_L\circ M_L)V_D(M_L\circ M_L)^T\big]\Big\} + c,\end{aligned} \tag{100}$$

where $\Lambda_{jk} = \langle\omega\rangle\langle\lambda_{jk}\rangle$. Note that the summation in the last term in (92) can be equivalently written as the trace in the above expression due to the fact that $\Lambda_{jj} = 0$ for all $j$.

The remaining expectations can be evaluated as:

$$\mathbb{E}_q[\log p(\boldsymbol{\lambda})] = -\frac{1}{2}\sum_{j=1}^p\sum_{k=j+1}^p\langle\log\lambda_{jk}\rangle - \sum_{j=1}^p\sum_{k=j+1}^p\langle\log(\lambda_{jk}+1)\rangle + c, \tag{101}$$

$$\mathbb{E}_q[\log p(\omega)] = -\langle\log\omega\rangle. \tag{102}$$

Taken together, we can obtain $\mathcal{L}_1$ in (23).

## Appendix C. Derivation of the Update Rules

As the variational distributions in BISN are in the minimal exponential family, the corresponding natural parameters $\boldsymbol{\theta}$ can be updated as follows according to the framework of the KL proximal gradient method [20]:

$$\boldsymbol{\theta}^{(\kappa+1)} = (1-\eta^{(\kappa)})\boldsymbol{\theta}^{(\kappa)} + \eta^{(\kappa)} \nabla_{\boldsymbol{\mu}} \mathcal{L}_1 |_{\boldsymbol{\mu}=\boldsymbol{\mu}^{(\kappa)}}, \tag{103}$$

where $\eta^{(\kappa)}$ is the step size, and $\boldsymbol{\mu}$ denotes the mean parameters of the variational distributions.

In order to obtain the update rules, we first need to calculate the gradient $\nabla_{\boldsymbol{\mu}}\mathcal{L}_1$ of $\mathcal{L}_1$ with respect to the mean parameters $\boldsymbol{\mu}$. In the proposed model, the mean parameters are $\langle L_{jk}\rangle$ and $\langle L_{jk}^2\rangle$ for $q(L_{jk})$, $\langle \log D_{jj}\rangle$ and $\langle D_{jj}\rangle$ for $q(D_{jj})$, $\langle z_{jk}\rangle$ for $q(z_{jk})$, $\langle \log \pi_{jk}\rangle$ and $\langle \log(1-\pi_{jk})\rangle$ for $q(\pi_{jk})$, $\langle \log \omega\rangle$ and $\langle \log(1-\omega)\rangle$ for $q(\omega)$, and $\langle \log \lambda_0\rangle$ and $\langle \lambda_0\rangle$ for $q(\lambda_0)$. The corresponding gradient $\nabla_{\boldsymbol{\mu}}\mathcal{L}_1$ for these mean parameters can be derived as:

$$\begin{bmatrix} \dfrac{\partial \mathcal{L}_1}{\partial \langle L_{jk}\rangle} \\ \dfrac{\partial \mathcal{L}_1}{\partial \langle L_{jk}^2\rangle} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \langle L_{jk}\rangle}{\partial M_{Ljk}} & \dfrac{\partial \langle L_{jk}^2\rangle}{\partial M_{Ljk}} \\ \dfrac{\partial \langle L_{jk}\rangle}{\partial V_{Ljk}} & \dfrac{\partial \langle L_{jk}^2\rangle}{\partial V_{Ljk}} \end{bmatrix}^{-1} \begin{bmatrix} \dfrac{\partial \mathcal{L}_1}{\partial M_{Ljk}} \\ \dfrac{\partial \mathcal{L}_1}{\partial V_{Ljk}} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \mathcal{L}_1}{\partial M_{Ljk}} - 2M_{Ljk}\dfrac{\partial \mathcal{L}_1}{\partial V_{Ljk}} \\ \dfrac{\partial \mathcal{L}_1}{\partial V_{Ljk}} \end{bmatrix}, \tag{104}$$

$$\begin{aligned} \begin{bmatrix} \dfrac{\partial \mathcal{L}_1}{\partial \langle D_{jj}\rangle} \\ \dfrac{\partial \mathcal{L}_1}{\partial \langle \log D_{jj}\rangle} \end{bmatrix} &= \begin{bmatrix} \dfrac{\partial \langle D_{jj}\rangle}{\partial \alpha_j} & \dfrac{\partial \langle \log D_{jj}\rangle}{\partial \alpha_j} \\ \dfrac{\partial \langle D_{jj}\rangle}{\partial \beta_j} & \dfrac{\partial \langle \log D_{jj}\rangle}{\partial \beta_j} \end{bmatrix}^{-1} \begin{bmatrix} \dfrac{\partial M_{Djj}}{\partial \alpha_j} & \dfrac{\partial V_{Djj}}{\partial \alpha_j} \\ \dfrac{\partial M_{Djj}}{\partial \beta_j} & \dfrac{\partial V_{Djj}}{\partial \beta_j} \end{bmatrix} \begin{bmatrix} \dfrac{\partial \mathcal{L}_1}{\partial M_{Djj}} \\ \dfrac{\partial \mathcal{L}_1}{\partial V_{Djj}} \end{bmatrix} \\ &= \begin{bmatrix} \dfrac{\partial \mathcal{L}_1}{\partial M_{Djj}} + \dfrac{1}{\beta_j}\Big(1 + \dfrac{\alpha_j \psi'(\alpha_j)}{\alpha_j\psi'(\alpha_j)-1}\Big)\dfrac{\partial \mathcal{L}_1}{\partial V_{Djj}} \\ \dfrac{n}{2} + p - j - \dfrac{\alpha_j}{\beta_j^2(\alpha_j\psi'(\alpha_j)-1)}\dfrac{\partial \mathcal{L}_1}{\partial V_{Djj}} \end{bmatrix}, \end{aligned} \tag{105}$$

$$\frac{\partial \mathcal{L}_1}{\partial \langle \log \omega\rangle} = \frac{p(p-1)}{4} - 1, \tag{106}$$

$$\frac{\partial \mathcal{L}_1}{\partial \langle \omega\rangle} = -\frac{1}{2}\sum_{j=1}^{p}\sum_{k=j+1}^{p} \langle \lambda_{jk}\rangle \langle (LDL^T)\circ(LDL^T)\rangle_{jk}, \tag{107}$$

$$\frac{\partial \mathcal{L}_1}{\partial \langle \lambda_{jk}\rangle} = -\frac{1}{2}\langle \omega\rangle \langle (LDL^T)\circ(LDL^T)\rangle_{jk}, \tag{108}$$

where

$$\begin{aligned} \frac{\partial \mathcal{L}_1}{M_{Ljk}} =& \big\{ -[nS + (M_L M_D M_L^T)\circ\Lambda]M_L M_D - [M_L(M_D\circ M_D + V_D)]\circ(\Lambda V_L) \\ & - (M_L V_D)\circ[\Lambda(M_L\circ M_L)]\big\}_{jk}, \end{aligned} \tag{109}$$

$$\frac{\partial \mathcal{L}_1}{V_{Ljk}} = \Big\{ -\frac{n}{2}\operatorname{diag}(S)\operatorname{diag}(M_D)^T - \frac{1}{2}\Lambda(M_L\circ M_L + V_L)(M_D\circ M_D + V_D)\Big\}_{jk}, \tag{110}$$

$$\begin{aligned} \frac{\partial \mathcal{L}_1}{M_{Djj}} =& \Big\{ -\frac{1}{2}\operatorname{diag}\left\{M_L^T[nS + (M_L M_D M_L^T)\circ\Lambda]M_L\right\} - \frac{n}{2}V_L^T\operatorname{diag}(S) \\ & - \frac{1}{2}\operatorname{diag}\left[V_L^T\Lambda(V_L + 2M_L\circ M_L)\right]\circ\operatorname{diag}(M_D)\Big\}_j, \end{aligned} \tag{111}$$

$$\frac{\partial \mathcal{L}_1}{V_{Djj}} = -\frac{1}{4}\,\mathrm{diag}\left[(M_L \circ M_L + V_L)^T \Lambda (M_L \circ M_L + V_L)\right]_j, \tag{112}$$

and

$$\begin{aligned}
&\langle (LDL^T) \circ (LDL^T) \rangle \\
=& (M_L M_D M_L^T) \circ (M_L M_D M_L^T) + (M_L \circ M_L)(M_D \circ M_D + V_D) V_L^T \\
&+ V_L (M_D \circ M_D + V_D)(M_L \circ M_L)^T + V_L (M_D \circ M_D + V_D) V_L^T + (M_L \circ M_L) V_D (M_L \circ M_L)^T \\
=& (M_L M_D M_L^T) \circ (M_L M_D M_L^T) + (M_L \circ M_L + V_L)(M_D \circ M_D + V_D)(M_L \circ M_L + V_L)^T \\
&- (M_L \circ M_L)(M_D \circ M_D)(M_L \circ M_L)^T.
\end{aligned} \tag{113}$$

Given the gradients, the updated rules for all natural parameters can then be derived as:

$$h_{jk}^{(\kappa+1)} = (1-\eta^{(\kappa)}) h_{jk}^{(\kappa)} + \eta^{(\kappa)} \frac{\partial \mathcal{L}_1}{\partial \langle L_{jk} \rangle}, \tag{114}$$

$$\zeta_{jk}^{(\kappa+1)} = (1-\eta^{(\kappa)}) \zeta_{jk}^{(\kappa)} - 2\eta^{(\kappa)} \frac{\partial \mathcal{L}_1}{\partial \langle L_{jk}^2 \rangle}, \tag{115}$$

$$\alpha_j^{(\kappa+1)} = (1-\eta^{(\kappa)}) \alpha_j^{(\kappa)} + \eta^{(\kappa)} \Big( \frac{\partial \mathcal{L}_1}{\partial \langle \log D_{jj} \rangle} + 1 \Big), \tag{116}$$

$$\beta_j^{(\kappa+1)} = (1-\eta^{(\kappa)}) \beta_j^{(\kappa)} - \eta^{(\kappa)} \frac{\partial \mathcal{L}_1}{\partial \langle D_{jj} \rangle}, \tag{117}$$

$$\epsilon_j^{(\kappa+1)} = (1-\eta^{(\kappa)}) \epsilon_j^{(\kappa)} + \eta^{(\kappa)} \frac{\partial \mathcal{L}_1}{\partial \langle z_{jk} \rangle}, \tag{118}$$

$$a^{(\kappa+1)} = (1-\eta^{(\kappa)}) a_0^{(\kappa)} + \eta^{(\kappa)} \Big( \frac{\partial \mathcal{L}_1}{\partial \langle \log \omega \rangle} + 1 \Big), \tag{119}$$

$$b^{(\kappa+1)} = (1-\eta^{(\kappa)}) b_0^{(\kappa)} - \eta^{(\kappa)} \frac{\partial \mathcal{L}_1}{\partial \langle \omega \rangle}, \tag{120}$$

$$d_{jk}^{(\kappa+1)} = (1-\eta^{(\kappa)}) d_{jk}^{(\kappa)} - \eta^{(\kappa)} \frac{\partial \mathcal{L}_1}{\partial \langle \lambda_{jk} \rangle}. \tag{121}$$

Substitute (104)-(108) into (114)-(121) and we can obtain the update rules in Eqs. (24)-(30) in the main body.

## Appendix D. Proof of Proposition 1

Before proving Proposition 1, we would like to introduce a lemma:

**Lemma 1** *(Khan et al., 2016) Suppose all assumptions in Section 4 are satisfied. Then the following holds for $\mu^{(\kappa)}$ in its domain, any real-valued column vector $R$, $\rho > 0$, and $g^{(\kappa)} = g(\mu^{(\kappa)}, R, \rho)$:*

$$R^T \boldsymbol{g}^{(\kappa)} \geq \gamma \|\boldsymbol{g}^{(\kappa)}\|^2 + \frac{1}{\rho}[h(\boldsymbol{\mu}^{(\kappa+1)} - h(\boldsymbol{\mu}^{(\kappa)})], \tag{122}$$

*where $\gamma$ is defined in* (55).

Next, let us return to the main thread. Since $f(\boldsymbol{\mu})$ is $l$-smooth, we have:

$$f(\boldsymbol{\mu}^{(\kappa+1)}) \leq f(\boldsymbol{\mu}^{(\kappa)}) + \nabla f(\boldsymbol{\mu}^{(\kappa)})^T(\boldsymbol{\mu}^{(\kappa+1)} - \boldsymbol{\mu}^{(\kappa)}) + \frac{l}{2}\|\boldsymbol{\mu}^{(\kappa+1)} - \boldsymbol{\mu}^{(\kappa)}\|^2. \tag{123}$$

It follows from the definition $\boldsymbol{g}^{(\kappa)} = (\boldsymbol{\mu}^{(\kappa)} - \boldsymbol{\mu}^{(\kappa+1)})/\rho$ that

$$f(\boldsymbol{\mu}^{(\kappa+1)}) \leq f(\boldsymbol{\mu}^{(\kappa)}) - \rho\nabla f(\boldsymbol{\mu}^{(\kappa)})^T\boldsymbol{g}^{(\kappa)} + \frac{l\rho^2}{2}\|\boldsymbol{g}^{(\kappa)}\|^2. \tag{124}$$

Note that we assume that the step size is a constant $\rho$ in Proposition 1. We further expand the second term on the right hand side (RHS) and obtain:

$$f(\boldsymbol{\mu}^{(\kappa+1)}) \leq f(\boldsymbol{\mu}^{(\kappa)}) - \rho {R^{(\kappa)}}^T\boldsymbol{g}^{(\kappa)} + \rho[R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})]^T\boldsymbol{g}^{(\kappa)} + \frac{l\rho^2}{2}\|\boldsymbol{g}^{(\kappa)}\|^2. \tag{125}$$

Substituting Lemma 1 into the above inequality yields:

$$\begin{aligned} f(\boldsymbol{\mu}^{(\kappa+1)}) \leq{} & f(\boldsymbol{\mu}^{(\kappa)}) - [\gamma\rho\|\boldsymbol{g}^{(\kappa)}\|^2 + h(\boldsymbol{\mu}^{(\kappa+1)}) - h(\boldsymbol{\mu}^{(\kappa)})] + \rho[R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})]^T\boldsymbol{g}^{(\kappa)} \\ & + \frac{l\rho^2}{2}\|\boldsymbol{g}^{(\kappa)}\|^2. \end{aligned} \tag{126}$$

Recall that $-\tilde{\mathcal{L}}(\boldsymbol{\mu}) = f(\boldsymbol{\mu}) + h(\boldsymbol{\mu})$, and therefore,

$$-\tilde{\mathcal{L}}(\boldsymbol{\mu}^{(\kappa+1)}) \leq -\tilde{\mathcal{L}}(\boldsymbol{\mu}^{(\kappa)}) - \gamma\rho\|\boldsymbol{g}^{(\kappa)}\|^2 + \rho[R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})]^T\boldsymbol{g}^{(\kappa)} + \frac{l\rho^2}{2}\|\boldsymbol{g}^{(\kappa)}\|^2. \tag{127}$$

By applying Cauchy-Schwarz and Young's inequality to the product $\rho[R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})]^T\boldsymbol{g}^{(\kappa)}$, for any $c_1 > 0$, we have:

$$-\tilde{\mathcal{L}}(\boldsymbol{\mu}^{(\kappa+1)}) \leq -\tilde{\mathcal{L}}(\boldsymbol{\mu}^{(\kappa)}) - \gamma\rho\|\boldsymbol{g}^{(\kappa)}\|^2 + \frac{\rho}{2c_1}\|\boldsymbol{g}^{(\kappa)}\|^2 + \frac{\rho c_1}{2}\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2 + \frac{l\rho^2}{2}\|\boldsymbol{g}^{(\kappa)}\|^2. \tag{128}$$

Summing both sides of the above inequality from $\kappa = 0$ to $\kappa = t$ gives:

$$-\tilde{\mathcal{L}}(\boldsymbol{\mu}^{(t)}) \leq -\tilde{\mathcal{L}}^0 - \left[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\right]\sum_{\kappa=0}^{t-1}\|\boldsymbol{g}^{(\kappa)}\|^2 + \frac{\rho c_1}{2}\sum_{\kappa=0}^{t-1}\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2, \tag{129}$$

where $\tilde{\mathcal{L}}^0 = \tilde{\mathcal{L}}(\boldsymbol{\mu}^{(0)})$. Since $\tilde{\mathcal{L}}^* \geq \tilde{\mathcal{L}}(\boldsymbol{\mu}^{(t)})$, by taking expectation over the distribution of the random subsets $\mathcal{S}^{(\kappa)}$ on both sides of the above inequality and noticing that $\|R^{(0)} - \nabla f(\boldsymbol{\mu}^{(0)})\|^2 = 0$, we can obtain the results in Proposition 1.

## Appendix E. Proof of Proposition 2

Next, let us turn our attention to $\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2]$, which can be expanded as:

$$\begin{aligned} & \mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2] \\ ={} & \mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}(R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)})) + r^{(\kappa)}(R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)}))\|^2] \end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)} - (\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})) + r^{(\kappa)}(R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)}))\|^2] \\
&= \mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}\|^2] + \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2] \\
&\quad + {r^{(\kappa)}}^2\mathbb{E}[\|R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)}\|^2] - 2\mathbb{E}[(R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)})^T(\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)}))] \\
&\quad + 2r^{(\kappa)}\mathbb{E}[R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}]^T\mathbb{E}[R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)})] \\
&\quad - 2r^{(\kappa)}\mathbb{E}[\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})]^T\mathbb{E}[R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)})]. \qquad (130)
\end{aligned}
$$

According to the definition of $R^{(\kappa)}$ in (58),

$$
\begin{aligned}
\mathbb{E}[(R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)})^T(\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)}))] &= \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2], \\
\mathbb{E}[R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)})] &= \mathbf{0}.
\end{aligned}
$$

As a result, $\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2]$ can be written as:

$$
\begin{aligned}
\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2] =& \mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)}\|^2] \\
&+ {r^{(\kappa)}}^2\mathbb{E}[\|R^{(\kappa-1)} - \nabla f(\boldsymbol{\mu}^{(\kappa-1)}\|^2]. \qquad (131)
\end{aligned}
$$

By applying the above equation recursively w.r.t. $\kappa$, we can obtain:

$$
\begin{aligned}
\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2] = \sum_{m=1}^{\kappa}\Bigg[\prod_{j=m+1}^{\kappa}{r^{(j)}}^2\Big(&\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) \\
&- r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]\Big)\Bigg] + \prod_{j=1}^{\kappa}{r^{(j)}}^2\mathbb{E}[\|R^{(0)} - \nabla f(\boldsymbol{\mu}^{(0)})\|^2].
\end{aligned}
\qquad (132)
$$

Since we compute the exact gradient in the first step, $\mathbb{E}[\|R^{(0)} - \nabla f(\boldsymbol{\mu}^{(0)})\|^2] = 0$ and we can obtain the results in Proposition 2.

## Appendix F. Proof of Proposition 3

In this appendix, we start with a lemma on the variance of $R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}$.

**Lemma 2** *Given the definition of the recursive gradient $R^{(\kappa)}$ in* (58)*, we can obtain the following equality:*

$$
\begin{aligned}
&\mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2] \\
&= \frac{1}{s}\frac{p-s}{p-1}\left[\frac{1}{p}\sum_{j=1}^{p}\|\nabla f_j(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f_j(\boldsymbol{\mu}^{(\kappa-1)})\|^2 - \|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2\right].
\end{aligned}
\qquad (133)
$$

**Proof** Define

$$
\xi_j = \nabla f_j(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f_j(\boldsymbol{\mu}^{(\kappa-1)}), \qquad (134)
$$

and we can express $\mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2]$ as a function of $\xi_j$:

$$\begin{aligned}
&\mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2] \\
=&\mathbb{E}\left[\left\|\frac{1}{s}\sum_{j\in\mathcal{S}^{(\kappa)}} \xi\right\|^2\right] - \frac{1}{p}\sum_{j=1}^{p}\xi_j && (135)\\
=&\mathbb{E}\left[\left\|\frac{1}{s}\sum_{j\in\mathcal{S}^{(\kappa)}} \xi\right\|^2\right] - E[\xi] && (136)\\
=&\mathbb{E}\left[\left\|\frac{1}{s}\sum_{j\in\mathcal{S}^{(\kappa)}} \big(\xi_j - \mathbb{E}[\xi]\big)\right\|^2\right], && (137)
\end{aligned}$$

where $\mathcal{S}^{(\kappa)}$ is a random subset of $\{1,\cdots,p\}$ with cardinality $s$ as defined in (58). On the other hand, (137) can be rewritten as:

$$\frac{1}{s}\sum_{j\in\mathcal{S}^{(\kappa)}} \big(\xi_j - \mathbb{E}[\xi]\big) = \frac{1}{s}\sum_{j=1}^{p} w_j\big(\xi_j - \mathbb{E}[\xi]\big), \tag{138}$$

where $w_j = 1$ only if $j \in \mathcal{S}^{(\kappa)}$ and $w_j = 0$ otherwise. It is easy to see that

$$\mathbb{E}[w_j^2] = \mathbb{E}[w_j] = \frac{s}{p}, \tag{139}$$

$$\mathbb{E}[w_j w_k] = \frac{s(s-1)}{p(p-1)}, \quad \forall j \neq k. \tag{140}$$

As such, we can express the LHS of (133) as:

$$\begin{aligned}
&\mathbb{E}[\|R^{(\kappa)} - r^{(\kappa)}R^{(\kappa-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(\kappa)}) - r^{(\kappa)}\nabla f(\boldsymbol{\mu}^{(\kappa-1)})\|^2] \\
=&\mathbb{E}\left[\left\|\frac{1}{s}\sum_{j=1}^{p} w_j\big(\xi_j - \mathbb{E}[\xi]\big)\right\|^2\right] \\
=&\frac{1}{s^2}\left(\sum_{j=1}^{p}\mathbb{E}[w_j^2]\big\|\xi_j - \mathbb{E}[\xi]\big\|^2 + \sum_{j=1}^{p}\sum_{k\neq j}\mathbb{E}[w_j w_k]\big(\xi_j - \mathbb{E}[\xi]\big)^T\big(\xi_k - \mathbb{E}[\xi]\big)\right) && (141)\\
=&\frac{1}{s^2}\left(\frac{s}{p}\sum_{j=1}^{p}\big\|\xi_j - \mathbb{E}[\xi]\big\|^2 + \frac{s(s-1)}{p(p-1)}\sum_{j=1}^{p}\sum_{k\neq j}\big(\xi_j - \mathbb{E}[\xi]\big)^T\big(\xi_k - \mathbb{E}[\xi]\big)\right) && (142)\\
=&\frac{1}{s^2}\left[\left(\frac{s}{p} - \frac{s(s-1)}{p(p-1)}\right)\sum_{j=1}^{p}\big\|\xi_j - \mathbb{E}[\xi]\big\|^2 + \frac{s(s-1)}{p(p-1)}\left\|\sum_{j=1}^{p}\big(\xi_j - \mathbb{E}[\xi]\big)\right\|^2\right] && (143)\\
=&\frac{1}{s^2}\left(\frac{s}{p} - \frac{s(s-1)}{p(p-1)}\right)\sum_{j=1}^{p}\big\|\xi_j - \mathbb{E}[\xi]\big\|^2 && (144)\\
=&\frac{1}{s}\frac{p-s}{p-1}\frac{1}{p}\sum_{j=1}^{p}\big\|\xi_j - \mathbb{E}[\xi]\big\|^2 && (145)
\end{aligned}$$

$$= \frac{1}{s}\frac{p-s}{p-1}\left(\frac{1}{p}\sum_{j=1}^{p}\left\|\xi_j\right\|^2 - E[\xi]^2\right). \tag{146}$$

■

Based on Lemma 2, we can further express $\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]$ as:

$$\begin{aligned}
&\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2] \\
=& \frac{1}{s}\frac{p-s}{p-1}\left[\frac{1}{p}\sum_{j=1}^{p}\|\nabla f_j(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f_j(\boldsymbol{\mu}^{(m-1)})\|^2 - \|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2\right] \\
=& \frac{1}{s}\frac{p-s}{p-1}\left[\frac{1}{p}\sum_{j=1}^{p}\Big(r^{(m)}\|\nabla f_j(\boldsymbol{\mu}^{(m)}) - \nabla f_j(\boldsymbol{\mu}^{(m-1)})\|^2 + (1-r^{(m)})\|\nabla f_j(\boldsymbol{\mu}^{(m)})\|^2\right. \\
&- r^{(m)}(1-r^{(m)})\|\nabla f_j(\boldsymbol{\mu}^{(m-1)})\|^2\Big) - \Big(r^{(m)}\|\nabla f(\boldsymbol{\mu}^{(m)}) - \nabla f(\boldsymbol{\mu}^{(m-1)})\|^2 \\
&\left.+ (1-r^{(m)})\|\nabla f(\boldsymbol{\mu}^{(m)})\|^2 - r^{(m)}(1-r^{(m)})\|\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2\Big)\right].
\end{aligned} \tag{147}$$

Let

$$\boldsymbol{v}^{(m)} = \frac{1}{p}\sum_{j=1}^{p}\|\nabla f_j(\boldsymbol{\mu}^{(m)})\|^2 - \|\nabla f(\boldsymbol{\mu}^{(m)})\|^2, \tag{148}$$

and ignore the term $r^{(m)}\|\nabla f(\boldsymbol{\mu}^{(m)}) - \nabla f(\boldsymbol{\mu}^{(m-1)})\|^2$, we can find the upper bound of $\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2]$, that is,

$$\begin{aligned}
&\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2] \\
\leq& \frac{1}{s}\frac{p-s}{p-1}\left(r^{(m)}\frac{1}{p}\sum_{j=1}^{p}\|\nabla f_j(\boldsymbol{\mu}^{(m)}) - \nabla f_j(\boldsymbol{\mu}^{(m-1)})\|^2 + (1-r^{(m)})\boldsymbol{v}^{(m)} - r^{(m)}(1-r^{(m)})\boldsymbol{v}^{(m-1)}\right).
\end{aligned}$$

Recall that we assume $f_j(\boldsymbol{\mu}^{(m)})$ is $l$-smooth and so

$$\|\nabla f_j(\boldsymbol{\mu}^{(m)}) - \nabla f_j(\boldsymbol{\mu}^{(m-1)})\|^2 \leq l^2\|\boldsymbol{\mu}^{(m)} - \boldsymbol{\mu}^{(m-1)}\|^2 = l^2\rho^2\|\boldsymbol{g}^{(m-1)}\|^2. \tag{149}$$

As a result, we can obtain

$$\begin{aligned}
&\mathbb{E}[\|R^{(m)} - r^{(m)}R^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r^{(m)}\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2] \\
\leq& \frac{1}{s}\frac{p-s}{p-1}\Big(r^{(m)}l^2\rho^2\mathbb{E}[\|\boldsymbol{g}^{(m-1)}\|^2] + (1-r^{(m)})\boldsymbol{v}^{(m)} - r^{(m)}(1-r^{(m)})\boldsymbol{v}^{(m-1)}\Big).
\end{aligned} \tag{150}$$

This closes the proof.

## Appendix G. Proof of Theorem 1

Put together Proposition 2 and 3 and assume that the decaying coefficient $r^{(\kappa)} = r$ is a constant, we can have:

$$\begin{aligned}
&\mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2] \\
&= \sum_{m=1}^{\kappa} \bigg( \prod_{j=m+1}^{\kappa} r^2 \bigg) \bigg( \mathbb{E}[\|R^{(m)} - rR^{(m-1)}\|^2] - \mathbb{E}[\|\nabla f(\boldsymbol{\mu}^{(m)}) - r\nabla f(\boldsymbol{\mu}^{(m-1)})\|^2] \bigg) \\
&\leq \frac{1}{s}\frac{p-s}{p-1} r l^2 \rho^2 \sum_{m=0}^{\kappa-1} r^{2(\kappa-m-1)} \mathbb{E}[\|\boldsymbol{g}^{(m)}\|] + \frac{1}{s}\frac{p-s}{p-1} \sum_{m=1}^{\kappa} r^{2(\kappa-m)} \big[(1-r)\boldsymbol{v}^{(m)} - r(1-r)\boldsymbol{v}^{(m-1)}\big].
\end{aligned} \tag{151}$$

Substituting (151) into Proposition 1 yields:

$$\begin{aligned}
&-\tilde{\mathcal{L}}^* \leq -\tilde{\mathcal{L}}^0 - \bigg[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\bigg] \sum_{\kappa=0}^{t-1} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1\rho}{2} \sum_{\kappa=1}^{t-1} \mathbb{E}[\|R^{(\kappa)} - \nabla f(\boldsymbol{\mu}^{(\kappa)})\|^2] \\
&\leq -\tilde{\mathcal{L}}^0 - \bigg[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\bigg] \sum_{\kappa=0}^{t-1} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1\rho}{2} \sum_{\kappa=1}^{t-1} \Bigg\{ \frac{1}{s}\frac{p-s}{p-1} r l^2 \rho^2 \sum_{m=0}^{\kappa-1} r^{2(\kappa-m-1)} \mathbb{E}[\|\boldsymbol{g}^{(m)}\|] \\
&\quad + \frac{1}{s}\frac{p-s}{p-1} \sum_{m=1}^{\kappa} r^{2(\kappa-m)} \big[(1-r)\boldsymbol{v}^{(m)} - r(1-r)\boldsymbol{v}^{(m-1)}\big] \Bigg\} \qquad (152) \\
&= -\tilde{\mathcal{L}}^0 - \bigg[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\bigg] \sum_{\kappa=0}^{t-1} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} r l^2 \rho^3 \sum_{\kappa=1}^{t-1} \sum_{m=0}^{\kappa-1} r^{2(\kappa-m-1)} \mathbb{E}[\|\boldsymbol{g}^{(m)}\|] \\
&\quad + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} \rho \sum_{\kappa=1}^{t-1} \sum_{m=1}^{\kappa} r^{2(\kappa-m)} \big[(1-r)\boldsymbol{v}^{(m)} - r(1-r)\boldsymbol{v}^{(m-1)}\big] \\
&= -\tilde{\mathcal{L}}^0 - \bigg[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\bigg] \sum_{\kappa=0}^{t-1} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} r l^2 \rho^3 \sum_{\kappa=0}^{t-2} \frac{1 - r^{2(t-\kappa-1)}}{1-r^2} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|] \\
&\quad + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} \rho \sum_{\kappa=1}^{t-1} \frac{1 - r^{2(t-\kappa)}}{1-r^2} \big[(1-r)\boldsymbol{v}^{(\kappa)} - r(1-r)\boldsymbol{v}^{(\kappa-1)}\big] \\
&= -\tilde{\mathcal{L}}^0 - \bigg[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\bigg] \sum_{\kappa=0}^{t-1} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} r l^2 \rho^3 \sum_{\kappa=0}^{t-2} \frac{1 - r^{2(t-\kappa-1)}}{1-r^2} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|] \\
&\quad + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} \rho \sum_{\kappa=1}^{t-1} \frac{(1-r)(1 - r^{2(t-\kappa)})}{1-r^2} \boldsymbol{v}^{(\kappa)} - \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} \rho \sum_{\kappa=0}^{t-2} \frac{r(1-r)(1 - r^{2(t-\kappa-1)})}{1-r^2} \boldsymbol{v}^{(\kappa)} \\
&= -\tilde{\mathcal{L}}^0 - \bigg[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\bigg] \sum_{\kappa=0}^{t-1} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} r l^2 \rho^3 \sum_{\kappa=0}^{t-2} \frac{1 - r^{2(t-\kappa-1)}}{1-r^2} \mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|] \\
&\quad + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} \rho (1-r) \boldsymbol{v}^{(t-1)} + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1} \rho \frac{1-r}{1+r} \sum_{\kappa=1}^{t-2} \big(1 + r^{2(t-\kappa)-1}\big) \boldsymbol{v}^{(\kappa)}
\end{aligned}$$

$$
\begin{aligned}
&- \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}\rho\frac{r(1-r^{2(t-1)})}{1+r}\boldsymbol{v}^0 && (153)\\
&\leq -\tilde{\mathcal{L}}^0 - \left[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\right]\sum_{\kappa=0}^{t-1}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}rl^2\rho^3\sum_{\kappa=0}^{t-2}\frac{1}{1-r^2}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|] \\
&\quad + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}\rho(1-r)\boldsymbol{v}^{(t-1)} + c_1\frac{1}{s}\frac{p-s}{p-1}\rho\frac{1-r}{1+r}\sum_{\kappa=1}^{t-2}\boldsymbol{v}^{(\kappa)} && (154)\\
&\leq -\tilde{\mathcal{L}}^0 - \left[\Big(\gamma - \frac{1}{2c_1}\Big)\rho - \frac{l}{2}\rho^2\right]\sum_{\kappa=0}^{t-1}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] + \frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}\frac{r}{1-r^2}l^2\rho^3\sum_{\kappa=0}^{t-1}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|] \\
&\quad + c_1\frac{1}{s}\frac{p-s}{p-1}\rho\frac{1-r}{1+r}t\sigma^2, && (155)
\end{aligned}
$$

where (154) holds since $1 - r^{2(t-\kappa-1)} < 1$ and we ignore the last term on RHS of (153) that is non-positive, and (155) holds since we use the assumption that $\boldsymbol{v}^{(\kappa)}$ is upper bound by $\sigma^2$ for all $\kappa$ (54). It follows that

$$
\begin{aligned}
&\rho\left[-\frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}\frac{r}{1-r^2}l^2\rho^2 - \frac{l}{2}\rho + \Big(\gamma - \frac{1}{2c_1}\Big)\right]\sum_{\kappa=0}^{t-1}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \\
&\leq \tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0 + c_1\frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}t\sigma^2\rho. && (156)
\end{aligned}
$$

Let

$$a_0 = \gamma - \frac{1}{2c_1} > 0, \tag{157}$$

$$a_1 = \frac{l}{2} > 0, \tag{158}$$

$$a_2 = \frac{r}{1-r^2}\frac{c_1}{2}\frac{1}{s}\frac{p-s}{p-1}l^2 > 0, \tag{159}$$

we can equivalently write (156) as:

$$\rho(-a_2\rho^2 - a_1\rho + a_0)\sum_{\kappa=0}^{t-1}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0 + c_1\frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}\rho t\sigma^2. \tag{160}$$

Note that $a_0 > 0$ since it is assumed that $c_1 > 1/2\gamma$. In order to find the upper bound on $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2]$, we need to ensure that

$$\rho(-a_2\rho^2 - a_1\rho + a_0) > 0. \tag{161}$$

To this end, $\rho$ should be chosen such that

$$0 < \rho < \frac{\sqrt{a_1^2 + 4a_2a_0} - a_1}{2a_2}. \tag{162}$$

Finally, $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2]$ can be upper bounded as:

$$\frac{1}{t}\sum_{\kappa=0}^{t-1}\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \frac{\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0}{t\rho(-a_2\rho^2 - a_1\rho + a_0)} + \frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}\frac{c_1\sigma^2}{-a_2\rho^2 - a_1\rho + a_0}. \tag{163}$$

Suppose that $\kappa$ is uniformly chosen at random from $\{0, \cdots, t-1\}$, the above inequality can be equivalently expressed as:

$$\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \frac{\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0}{t\rho(-a_2\rho^2 - a_1\rho + a_0)} + \frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}\frac{c_1\sigma^2}{-a_2\rho^2 - a_1\rho + a_0}, \tag{164}$$

where the $\mathbb{E}$ is taken w.r.t. all kinds of randomness.

## Appendix H. Proof of Theorem 2

Given the specific value of $\rho$, that is,

$$\rho = \frac{\sqrt{a_1^2 + 3a_2a_0} - a_1}{3a_2}, \tag{165}$$

we can lower bound $(-a_2\rho^2 - a_1\rho + a_0)$ as:

$$\begin{aligned}
(-a_2\rho^2 - a_1\rho + a_0) &= -a_2\left(\rho + \frac{a_1 - \sqrt{a_1^2 + 4a_2a_0}}{2a_2}\right)\left(\rho + \frac{a_1 + \sqrt{a_1^2 + 4a_2a_0}}{2a_2}\right) \\
&= -a_2\left(\frac{\sqrt{a_1^2 + 3a_2a_0} - a_1}{3a_2} + \frac{a_1 - \sqrt{a_1^2 + 4a_2a_0}}{2a_2}\right)\left(\frac{\sqrt{a_1^2 + 3a_2a_0} - a_1}{3a_2} + \frac{a_1 + \sqrt{a_1^2 + 4a_2a_0}}{2a_2}\right) \\
&= a_2\left(\frac{-a_1 - 2\sqrt{a_1^2 + 3a_2a_0} + 3\sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right)\left(\frac{a_1 + 2\sqrt{a_1^2 + 3a_2a_0} + 3\sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right) \\
&\geq a_2\left(\frac{-a_1 - 2\sqrt{a_1^2 + 4a_2a_0} + 3\sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right)\left(\frac{a_1 + 2\sqrt{a_1^2 + 3a_2a_0} + 3\sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right) \\
&= a_2\left(\frac{-a_1 + \sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right)\left(\frac{a_1 + 2\sqrt{a_1^2 + 3a_2a_0} + 3\sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right) \\
&\geq a_2\left(\frac{-a_1 + \sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right)\left(\frac{a_1 + \sqrt{a_1^2 + 4a_2a_0}}{6a_2}\right) \\
&= \frac{a_0}{9}.
\end{aligned} \tag{166}$$

As a result, we can further relax the upper bound of $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2]$ as:

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] &\leq \frac{\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0}{t\rho(-a_2\rho^2 - a_1\rho + a_0)} + \frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}\frac{c_1\sigma^2}{-a_2\rho^2 - a_1\rho + a_0} &(167)\\
&\leq \frac{9(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)}{ta_0\rho} + \frac{1}{s}\frac{p-s}{p-1}\frac{1-r}{1+r}\frac{9c_1\sigma^2}{a_0}. &(168)
\end{aligned}$$

Recall that

$$s = \frac{p}{c_2(p-1) + 1}, \tag{169}$$

where $c_2 \in (0,1]$ is a fixed constant, and therefore, the ratio

$$\frac{1}{s}\frac{p-s}{p-1} = c_2 \tag{170}$$

is fixed. The upper bound in (168) can be equivalently written as:

$$\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \frac{9(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)}{ta_0\rho} + \frac{1-r}{1+r}\frac{9c_1c_2\sigma^2}{a_0}. \tag{171}$$

So as to obtain $\mathbb{E}[\|\boldsymbol{g}^{(\kappa)}\|^2] \leq \epsilon$, we set

$$\frac{9(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)}{ta_0\rho} + \frac{1-r}{1+r}\frac{9c_1c_2\sigma^2}{a_0} = \epsilon, \tag{172}$$

Given the specific values of $r$, that is,

$$r = \frac{1-c_3\epsilon}{1+c_3\epsilon}, \tag{173}$$

where

$$c_3 = \frac{a_0}{9c_1c_2c_4\sigma^2}, \tag{174}$$

and $c_4$ is an arbitrary but fixed constant satisfying

$$c_4 > \max\Big(1, \frac{a_0\epsilon}{9c_1c_2\sigma^2}\Big), \tag{175}$$

we can have

$$\frac{1-r}{1+r}\frac{9c_1c_2\sigma^2}{a_0} = \frac{\epsilon}{c_4}. \tag{176}$$

Therefore, to achieve the $\epsilon$-accuracy, the following equality must hold:

$$\frac{9(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)}{ta_0\rho} = \Big(1 - \frac{1}{c_4}\Big)\epsilon, \tag{177}$$

where $(1 - 1/c_4) > 0$ given the definition of $c_4$. We then replace $\rho$ in the above equation by its specific value in (165):

$$\frac{27a_2(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)}{ta_0(\sqrt{a_1^2 + 3a_2a_0} - a_1)} = \Big(1 - \frac{1}{c_4}\Big)\epsilon, \tag{178}$$

$$\frac{9(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)(\sqrt{a_1^2 + 3a_2a_0} + a_1)}{ta_0^2} = \Big(1 - \frac{1}{c_4}\Big)\epsilon, \tag{179}$$

Let

$$c_5 = \frac{9(\tilde{\mathcal{L}}^* - \tilde{\mathcal{L}}^0)}{a_0^2\big(1 - \frac{1}{c_4}\big)}, \tag{180}$$

we can have

$$t = \frac{c_5(\sqrt{a_1^2 + 3a_2a_0} + a_1)}{\epsilon}. \tag{181}$$

Substituting (173), (170), and (67) into (181) gives

$$t = \frac{c_5\left(\sqrt{a_1^2 + \frac{3}{2}a_0c_1c_2l^2\frac{1-c_3^2\epsilon^2}{4c_3\epsilon}} + a_1\right)}{\epsilon}. \tag{182}$$

On the other hand, $c_3\epsilon < 1$ according to the definition of $c_3$ and $c_4$, hence,

$$\begin{align} t &\leq \frac{c_5\left(\sqrt{a_1^2 + \frac{3}{2}a_0c_1c_2l^2\frac{1}{4c_3\epsilon}} + a_1\right)}{\epsilon} \tag{183} \\ &= \frac{c_5\left(\sqrt{4a_1^2c_3\epsilon + \frac{3}{2}a_0c_1c_2l^2} + 2a_1\sqrt{c_3\epsilon}\right)}{2\sqrt{c_3}\epsilon^{\frac{3}{2}}} \tag{184} \\ &\leq \frac{c_5\left(\sqrt{4a_1^2 + \frac{3}{2}a_0c_1c_2l^2} + 2a_1\right)}{2\sqrt{c_3}\epsilon^{\frac{3}{2}}} = \mathcal{O}\left(\frac{1}{\epsilon^{\frac{3}{2}}}\right). \tag{185} \end{align}$$

Note that the constants $c_1$, $c_2$, $c_3$, $c_5$, $a_0$, and $a_1$ are independent of $p$ and $\epsilon$.

## References

Daniel Felix Ahelegbey and Paolo Giudici. Hierarchical graphical models with application to systemic risk. *University Ca'Foscari of Venice, Dept. of Economics Research Paper Series No*, 1, 2014.

Daniel Felix Ahelegbey, Monica Billio, and Roberto Casarin. Bayesian graphical models for structural vector autoregressive processes. *Journal of Applied Econometrics*, 31(2): 357–386, 2016.

Mark Adrian S Asinas. Stock market betas for cyclical and defensive sectors: A practitioners perspective. *Philippine Management Review*, 25, 2018.

Onureena Banerjee, Laurent El Ghaoui, and Alexandre dAspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine learning research*, 9(3):485–516, 2008.

Matteo Barigozzi and Christian Brownlees. Nets: Network estimation for time series. *Journal of Applied Econometrics*, 34(3):347–364, 2019.

Matthew J Beal, Zoubin Ghahramani, et al. Variational bayesian learning of directed graphical models with hidden variables. *Bayesian Analysis*, 1(4):793–831, 2006.

Eugene Belilovsky, Gaël Varoquaux, and Matthew B Blaschko. Testing for differences in gaussian graphical models: applications to brain connectivity. In *Advances in Neural Information Processing Systems*, pages 595–603, 2016.

Victor Bernal, Rainer Bischoff, Victor Guryev, Marco Grzegorczyk, and Peter Horvatovich. Exact hypothesis testing for shrinkage based gaussian graphical models. *Bioinformatics*, 2019.

Anindya Bhadra, Jyotishka Datta, Nicholas G Polson, and Brandon Willard. The horseshoe+ estimator of ultra-sparse signals. *Bayesian Analysis*, 12(4):1105–1131, 2017.

Daniele Bianchi, Monica Billio, Roberto Casarin, and Massimo Guidolin. Modeling systemic risk with markov switching graphical sur models. *Journal of Econometrics*, 210(1):58–74, 2019.

Monica Billio, Mila Getmansky, Andrew W Lo, and Loriana Pelizzon. Econometric measures of connectedness and systemic risk in the finance and insurance sectors. *Journal of financial economics*, 104(3):535–559, 2012.

Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

Matthias Bollhöfer, Aryan Eftekhari, Simon Scheidegger, and Olaf Schenk. Large-scale sparse inverse covariance matrix estimation. *SIAM Journal on Scientific Computing*, 41 (1):A380–A401, 2019.

James Bullard, Christopher J Neely, and David C Wheelock. Systemic risk and the financial crisis: a primer. *Federal Reserve Bank of St. Louis Review*, 91(September/October 2009), 2009.

Miao Cao, Hao Huang, Yun Peng, Qi Dong, and Yong He. Toward developmental connectomics of the human brain. *Frontiers in neuroanatomy*, 10:25, 2016.

Carlos M Carvalho, Nicholas G Polson, and James G Scott. Handling sparsity via the horseshoe. In *Artificial Intelligence and Statistics*, pages 73–80, 2009.

Carlos M Carvalho, Nicholas G Polson, and James G Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

Paola Cerchiello, Paolo Giudici, and Giancarlo Nicola. Twitter data models for bank risk contagion. *Neurocomputing*, 264:50–56, 2017.

Venkat Chandrasekaran, Pablo A Parrilo, Alan S Willsky, et al. Latent variable graphical model selection via convex optimization. *The Annals of Statistics*, 40(4):1935–1967, 2012.

Myung Jin Choi, Venkat Chandrasekaran, and Alan S Willsky. Gaussian multiresolution models: Exploiting sparse markov and covariance structure. *IEEE Transactions on Signal Processing*, 58(3):1012–1024, 2009.

Hyonho Chun, Xianghua Zhang, and Hongyu Zhao. Gene regulation network inference with joint sparse gaussian graphical models. *Journal of Computational and Graphical Statistics*, 24(4):954–974, 2015.

Rama Cont, Amal Moussa, et al. Network structure and systemic risk in banking systems. *Edson Bastos e, Network Structure and Systemic Risk in Banking Systems (December 1, 2010)*, 2010.

Justin Dauwels, Hang Yu, Xueou Wang, Francois Vialatte, Charles Latchoumane, Jaeseung Jeong, and Andrzej Cichocki. Inferring brain networks through graphical models with hidden variables. In *Machine Learning and Interpretation in Neuroimaging*, pages 194–201. Springer, 2012.

Justin Dauwels, Hang Yu, Shiyan Xu, and Xueou Wang. Copula gaussian graphical model for discrete data. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6283–6287. IEEE, 2013.

Wenping Deng, Kui Zhang, Sanzhen Liu, Patrick X Zhao, Shizhong Xu, and Hairong Wei. Jrmgrn: joint reconstruction of multiple gene regulatory networks with common hub genes using data from multiple tissues or conditions. *Bioinformatics*, 34(20):3470–3478, 2018.

Luu-Ngoc Do and Hyung-Jeong Yang. A robust feature selection method for classification of cognitive states with fmri data. In *Advances in computer science and its applications*, pages 71–76. Springer, 2014.

Adrian Dobra, Alex Lenkoski, and Abel Rodriguez. Bayesian inference for general gaussian graphical models with application to multivariate lattice data. *Journal of the American Statistical Association*, 106(496):1418–1433, 2011.

Julien Doyon and Brenda Milner. Right temporal-lobe contribution to global visual processing. *Neuropsychologia*, 29(5):343–360, 1991.

John Duchi, Stephen Gould, and Daphne Koller. Projected subgradient methods for learning sparse gaussians. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 153–160. AUAI Press, 2008.

Stefania Evangelisti, Claudia Testa, Lorenzo Ferri, Laura Ludovica Gramegna, David Neil Manners, Giovanni Rizzo, Daniel Remondini, Gastone Castellani, Ilaria Naldi, Francesca Bisulli, et al. Brain functional connectivity in sleep-related hypermotor epilepsy. *NeuroImage: Clinical*, 17:873–881, 2018.

Jianqing Fan, Yuan Liao, and Han Liu. Approaches to high-dimensional covariance and precision matrix estimations. *Financial Signal Processing and Machine Learning*, pages 100–134, 2016.

Alireza Farasat, Alexander Nikolaev, Sargur N Srihari, and Rachael Hageman Blair. Probabilistic graphical models in modern social network analysis. *Social Network Analysis and Mining*, 5(1):62, 2015.

A Marie Fitch and M Beatrix Jones. Shortest path analysis using partial correlations for classifying gene functions from gene expression data. *Bioinformatics*, 25(1):42–47, 2009.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305, 2016.

Shatha Qamhieh Hashem and Paolo Giudici. Systemic risk of conventional and islamic banks: comparison with graphical network models. *Applied Mathematics*, 7(17):2079–96, 2016.

Matthew D Hoffman, David M Blei, Chong Wang, and John William Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Cho-Jui Hsieh, Mátyás A Sustik, Inderjit S Dhillon, Pradeep K Ravikumar, and Russell Poldrack. Big & quic: Sparse inverse covariance estimation for a million variables. In *Advances in Neural Information Processing Systems*, pages 3165–3173, 2013.

Cho-Jui Hsieh, Mátyás A Sustik, Inderjit S Dhillon, and Pradeep Ravikumar. Quic: quadratic approximation for sparse inverse covariance estimation. *Journal of Machine Learning Research*, 15(1):2911–2947, 2014.

Jianhua Z Huang, Naiping Liu, Mohsen Pourahmadi, and Linxu Liu. Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, 93(1):85–98, 2006.

Timothy R Hughes, Matthew J Marton, Allan R Jones, Christopher J Roberts, Roland Stoughton, Christopher D Armour, Holly A Bennett, Ernest Coffey, Hongyue Dai, Yudong D He, et al. Functional discovery via a compendium of expression profiles. *Cell*, 102(1):109–126, 2000.

Bochao Jia and Faming Liang. Learning gene regulatory networks with high-dimensional heterogeneous data. In *New Frontiers of Biostatistics and Bioinformatics*, pages 305–327. Springer, 2018.

Waldemar Karwowski, Farzad Vasheghani Farahani, and Nichole Lighthall. Application of graph theory for identifying connectivity patterns in human brain networks: A systematic review. *Frontiers in Neuroscience*, 13:585, 2019.

Christopher L Keown, Michael C Datko, Colleen P Chen, Jose Omar Maximo, Afrooz Jahedi, and Ralph-Axel Müller. Network organization is globally atypical in autism: a graph theory study of intrinsic functional connectivity. *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, 2(1):66–75, 2017.

Mohammad E Khan, Pierre Baqué, François Fleuret, and Pascal Fua. Kullback-leibler proximal variational inference. In *Advances in Neural Information Processing Systems*, pages 3402–3410, 2015.

Mohammad Emtiyaz Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *Proceedings of the Twentieth Conference on Artificial Intelligence and Statistics*, 2017.

Mohammad Emtiyaz Khan, Reza Babanezhad, Wu Lin, Mark Schmidt, and Masashi Sugiyama. Faster stochastic variational inference using proximal-gradient methods with general divergence functions. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 319–328. AUAI Press, 2016.

Alp Kucukelbir, Rajesh Ranganath, Andrew Gelman, and David Blei. Automatic variational inference in stan. In *Advances in neural information processing systems*, pages 568–576, 2015.

Ginette Lafit, Francisco J Nogales, Marcelo Ruiz, and Ruben H Zamar. A stepwise approach for high-dimensional gaussian graphical models. *arXiv preprint arXiv:1808.06016*, 2018.

Gwenaël GR Leday and Sylvia Richardson. Fast bayesian inference in large gaussian graphical models. *Biometrics*, 75(4):1288–1298, 2019.

William J Lentz. Generating bessel functions in mie scattering calculations using continued fractions. *Applied Optics*, 15(3):668–671, 1976.

Jinzhou Li and Marloes H Maathuis. Nodewise knockoffs: False discovery rate control for gaussian graphical models. *arXiv preprint arXiv:1908.11611*, 2019.

Shuang Li, Li Hsu, Jie Peng, and Pei Wang. Bootstrap inference for network construction with an application to a breast cancer microarray study. *The annals of applied statistics*, 7(1):391, 2013.

Yunfan Li, Bruce A Craig, and Anindya Bhadra. The graphical horseshoe estimator for inverse covariance matrices. *Journal of Computational and Graphical Statistics*, pages 1–24, 2019.

Xia Liang, Qihong Zou, Yong He, and Yihong Yang. Topologically reorganized connectivity architecture of default-mode, executive-control, and salience networks across working memory task loads. *Cerebral cortex*, 26(4):1501–1511, 2016.

Han Liu and Lie Wang. Tiger: A tuning-insensitive approach for optimally estimating gaussian graphical models. *Electronic Journal of Statistics*, 11(1):241–294, 2017.

Han Liu, John Lafferty, and Larry Wasserman. The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *Journal of Machine Learning Research*, 10(Oct):2295–2328, 2009.

Han Liu, Kathryn Roeder, and Larry Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. In *Advances in neural information processing systems*, pages 1432–1440, 2010.

Qiang Liu and Alexander T Ihler. Variational algorithms for marginal map. *Journal of Machine Learning Research*, 14(1):3165–3200, 2013.

Benjamin M Marlin and Kevin P Murphy. Sparse gaussian graphical models with unknown block structure. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 705–712. ACM, 2009.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *The annals of statistics*, pages 1436–1462, 2006.

Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

Brenda Milner. Visual recognition and recall after right temporal-lobe excision in man. *Epilepsy & Behavior*, 4(6):799–812, 2003.

Tom M Mitchell, Rebecca Hutchinson, Radu S Niculescu, Francisco Pereira, Xuerui Wang, Marcel Just, and Sharlene Newman. Learning to decode cognitive states from brain images. *Machine learning*, 57(1):145–175, 2004.

Abdolreza Mohammadi and Ernst C Wit. Bayesian structure learning in sparse gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.

Sarah E Neville, John T Ormerod, and MP Wand. Mean field variational bayes for continuous sparse signal shrinkage: pitfalls and remedies. *Electronic Journal of Statistics*, 8(1):1113–1151, 2014.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Stochastic recursive gradient algorithm for nonconvex optimization. *arXiv preprint arXiv:1705.07261*, 2017.

Andrés Ortiz, Jorge Munilla, Ignacio Álvarez-Illán, Juan M Górriz, Javier Ramírez, Alzheimer's Disease Neuroimaging Initiative, et al. Exploratory graphical models of functional and structural connectivity patterns for alzheimer's disease diagnosis. *Frontiers in computational neuroscience*, 9:132, 2015.

Juho Piironen, Aki Vehtari, et al. Sparsity information and regularization in the horseshoe and other shrinkage priors. *Electronic Journal of Statistics*, 11(2):5018–5051, 2017.

Rajesh Ranganath, Chong Wang, Blei David, and Eric Xing. An adaptive learning rate for stochastic variational inference. In *International Conference on Machine Learning*, pages 298–306, 2013.

Sashank J Reddi, Suvrit Sra, Barnabas Poczos, and Alexander J Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In *Advances in Neural Information Processing Systems*, pages 1145–1153, 2016.

Irina Rish and Genady Grabarnik. *Sparse modeling: theory, algorithms, and applications*. CRC press, 2014.

Benjamin Rolfs, Bala Rajaratnam, Dominique Guillot, Ian Wong, and Arian Maleki. Iterative thresholding algorithm for sparse inverse covariance estimation. In *Advances in Neural Information Processing Systems*, pages 1574–1582, 2012.

Conrad Sanderson and Ryan Curtin. Armadillo: a template-based c++ library for linear algebra. *Journal of Open Source Software*, 1(2):26, 2016.

Katya Scheinberg and Irina Rish. Learning sparse gaussian markov networks using a greedy coordinate ascent approach. *Machine Learning and Knowledge Discovery in Databases*, pages 196–212, 2010.

Katya Scheinberg, Shiqian Ma, and Donald Goldfarb. Sparse inverse covariance selection via alternating linearization methods. In *Advances in neural information processing systems*, pages 2101–2109, 2010.

Julia Schumacher, Luis R Peraza, Michael Firbank, Alan J Thomas, Marcus Kaiser, Peter Gallagher, John T O'Brien, Andrew M Blamire, and John-Paul Taylor. Functional connectivity in dementia with lewy bodies: A within-and between-network analysis. *Human brain mapping*, 39(3):1118–1129, 2018.

Sviatlana Shashkova, Niek Welkenhuysen, and Stefan Hohmann. Molecular communication: crosstalk between the snf1 and other signaling pathways. *FEMS yeast research*, 15(4), 2015.

Kobi J Simpson-Lavy, Alex Bronstein, Martin Kupiec, and Mark Johnston. Cross-talk between carbon metabolism and the dna damage response in s. cerevisiae. *Cell reports*, 12 (11):1865–1875, 2015.

Michael Smith and Robert Kohn. Parsimonious covariance matrix estimation for longitudinal data. *Journal of the American Statistical Association*, 97(460):1141–1153, 2002.

Lingtao Su, Xiangyu Meng, Qingshan Ma, Tian Bai, and Guixia Liu. Lprp: a gene–gene interaction network construction algorithm and its application in breast cancer data analysis. *Interdisciplinary Sciences: Computational Life Sciences*, 10(1):131–142, 2018.

Davoud Ataee Tarzanagh and George Michailidis. Estimation of graphical models through structured norm minimization. *The Journal of Machine Learning Research*, 18(1):7692–7739, 2018.

Sekhar Tatikonda et al. Learning unfaithful $k$-separable gaussian graphical models. *Journal of Machine Learning Research*, 20(109):1–30, 2019.

Eran Treister and Javier S Turek. A block-coordinate descent approach for large-scale sparse inverse covariance estimation. In *Advances in neural information processing systems*, pages 927–935, 2014.

Eran Treister, Javier S Turek, and Irad Yavneh. A multilevel framework for sparse optimization with application to inverse covariance estimation and logistic regression. *SIAM Journal on Scientific Computing*, 38(5):S566–S592, 2016.

Martijn P van den Heuvel, Siemon C de Lange, Andrew Zalesky, Caio Seguin, BT Thomas Yeo, and Ruben Schmidt. Proportional thresholding in resting-state fmri functional connectivity networks and consequences for patient-control connectome studies: Issues and recommendations. *Neuroimage*, 152:437–449, 2017.

Hao Wang. Bayesian graphical lasso models and efficient posterior computation. *Bayesian Analysis*, 7(4):867–886, 2012.

Hao Wang. Scaling it up: Stochastic search structure learning in graphical models. *Bayesian Analysis*, 10(2):351–377, 2015.

Min Wang, Ping Yang, Chaoyang Wan, Zhenlan Jin, Junjun Zhang, and Ling Li. Evaluating the role of the dorsolateral prefrontal cortex and posterior parietal cortex in memory-guided attention with repetitive transcranial magnetic stimulation. *Frontiers in human neuroscience*, 12, 2018.

Donald R Williams, Juho Piironen, Aki Vehtari, and Philippe Rast. Bayesian estimation of gaussian graphical models with predictive covariance selection. *arXiv preprint arXiv:1801.05725*, 2018.

Mark Williams. *Uncontrolled risk: lessons of Lehman brothers and how systemic risk can still bring down the world financial system*. McGraw Hill Professional, 2010.

Yuting Xu and Martin A Lindquist. Dynamic connectivity detection: an algorithm for determining functional connectivity change points in fmri data. *Frontiers in neuroscience*, 9:285, 2015.

Jilei Yang and Jie Peng. Estimating time-varying graphical models. *Journal of Computational and Graphical Statistics*, pages 1–12, 2019.

Hang Yu and Justin Dauwels. Variational bayes learning of graphical models with hidden variables. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2015.

Hang Yu and Justin Dauwels. Variational bayes learning of time-varying graphical models. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016.

Hang Yu and Justin Dauwels. Modeling functional networks via piecewise-stationary graphical models. In *Signal Processing and Machine Learning for Biomedical Big Data*, pages 193–208. CRC Press, 2018.

Hang Yu, Justin Dauwels, and Xueou Wang. Copula gaussian graphical models with hidden variables. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2177–2180. IEEE, 2012.

Hang Yu, Luyin Xin, and Justin Dauwels. Variational wishart approximation for graphical model selection: Monoscale and multiscale models. *IEEE Transactions on Signal Processing*, 67(24):6468–6482, 2019.

Aiying Zhang, Jian Fang, Faming Liang, Vince D Calhoun, and Yuping Wang. Aberrant brain connectivity in schizophrenia detected via a fast gaussian graphical model. *IEEE journal of biomedical and health informatics*, 2018a.

Aiying Zhang, Biao Cai, Wenxing Hu, Bochao Jia, Faming Liang, Tony W Wilson, Julia M Stephen, Vince D Calhoun, and Yu-Ping Wang. Joint bayesian-incorporating estimation of multiple gaussian graphical models to study brain connectivity development in adolescence. *IEEE transactions on medical imaging*, 2019.

Richard Zhang, Salar Fattahi, and Somayeh Sojoudi. Large-scale sparse inverse covariance estimation via thresholding and max-det matrix completion. In *International Conference on Machine Learning*, pages 5761–5770, 2018b.

Haitao Zhao and Zhong-Hui Duan. Cancer genetic network inference using gaussian graphical models. *Bioinformatics and biology insights*, 13:1177932219839402, 2019.

Tuo Zhao, Han Liu, Kathryn Roeder, John Lafferty, and Larry Wasserman. The huge package for high-dimensional undirected graph estimation in r. *Journal of Machine Learning Research*, 13(Apr):1059–1062, 2012.