# TUDelft

Delft University of Technology

# Efficient Variational Bayes Learning of Graphical Models with Smooth Structural Changes

Yu, Hang; Wu, Songwei; Dauwels, Justin

**Important note**
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

# Efficient Variational Bayes Learning of Graphical Models With Smooth Structural Changes

Hang Yu ⬤, *Member, IEEE*, Songwei Wu, and Justin Dauwels, *Senior Member, IEEE*

**Abstract**—Estimating a sequence of dynamic undirected graphical models, in which adjacent graphs share similar structures, is of paramount importance in various social, financial, biological, and engineering systems, since the evolution of such networks can be utilized for example to spot trends, detect anomalies, predict vulnerability, and evaluate the impact of interventions. Existing methods for learning dynamic graphical models require the tuning parameters that control the graph sparsity and the temporal smoothness to be selected via brute-force grid search. Furthermore, these methods are computationally burdensome with time complexity $\mathcal{O}(NP^3)$ for $P$ variables and $N$ time points. As a remedy, we propose a low-complexity tuning-free Bayesian approach, named BASS. Specifically, we impose temporally dependent spike and slab priors on the graphs such that they are sparse and varying smoothly across time. An efficient variational inference algorithm based on natural gradients is then derived to learn the graph structures from the data in an automatic manner. Owing to the pseudo-likelihood and the mean-field approximation, the time complexity of BASS is only $\mathcal{O}(NP^2)$. To cope with the local maxima problem of variational inference, we resort to simulated annealing and propose a method based on bootstrapping of the observations to generate the annealing noise. We provide numerical evidence that BASS outperforms existing methods on synthetic data in terms of structure estimation, while being more efficient especially when the dimension $P$ becomes high. We further apply the approach to the stock return data of 78 banks from 2005 to 2013 and find that the number of edges in the financial network as a function of time contains three peaks, in coincidence with the 2008 global financial crisis and the two subsequent European debt crisis. On the other hand, by identifying the frequency-domain resemblance to the time-varying graphical models, we show that BASS can be extended to learning frequency-varying inverse spectral density matrices, and further yields graphical models for multivariate stationary time series. As an illustration, we analyze scalp EEG signals of patients at the early stages of Alzheimer's disease (AD) and show that the brain networks extracted by BASS can better distinguish between the patients and the healthy controls.

**Index Terms**—Graphical models, structural changes, variational inference, simulated annealing, inverse spectral density matrices

✦

## 1 INTRODUCTION

THE recent decades have witnessed a rapid development of graphical models, since they provide a refined language to describe complicated systems and further facilitate the derivation of efficient inference algorithms [2]. While an extensive literature revolves around learning static graphical models that are time-invariant (see [3], [4], [5], [6], [7], [8], [9], [10], [11] and references therein), the change of interdependencies with a covariate (e.g., time or space) is often the rule rather than the exception for real-world data, such as friendships between individuals in a social community, communications between genes in a cell, equity trading between companies, and

computer network traffic. Furthermore, such dynamic graphical models can be leveraged to spot trends, detect anomalies, classify events, evaluate the impact of interventions, and predict future behaviors of the systems. For instance, estimating time-varying functional brain networks during epileptic seizures can show how the dysrhythmia of the brain propagates, and analyzing the network evolution can help to detect epilepsy and assess the treatment of epilepsy [12]. We therefore focus on learning dynamic graphical models in this study.

In the case where all variables follow a joint Gaussian distribution, the graphical model structure is directly defined by the precision matrix (i.e., the inverse covariance matrix). Specifically, a zero element corresponds to the absence of an edge in the graphical model or, equivalently, the conditional independence between two variables. Therefore, our objective is to learn a time-varying precision matrix. Existing works on learning the time-varying precision matrices can be categorized into three groups. The first one [13], [14], [15], [16] considers the temporal dependence by smoothing the empirical covariance matrix across time using kernels. Given the temporally dependent covariance matrix, the sparse precision matrix is then estimated individually at each time point. The estimation problem can be solved by maximizing the likelihood with an $\ell_1$-norm penalty on the precision matrix. However, unexpected variability may arise between two adjacent networks since each network is estimated independently [17]. To mitigate this issue, the second group of dynamic network models [17],

- Hang Yu was with the Nanyang Technological Unversity, Singapore. He is now with Ant Group, Hangzhou 310000, China. E-mail: HYU1@e.ntu.edu.sg.
- Songwei Wu is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798. E-mail: wuso002@e.ntu.edu.sg.
- Justin Dauwels is with Electrical Engineering, Mathematics & Computer Science, Delft University of Technology, 2628, CD, Delft, The Netherlands. E-mail: J.H.G.Dauwels@tudelft.nl.

[18], [19], [20], [21], [22], [23], [24] further captures the temporal dependence by enforcing $\ell_1$, $\ell_2$, or Frobenius norm constraints on the difference between two consecutive precision matrices. As an alternative, instead of imposing separate constraints for the sparsity and the smoothness across time on the precision matrices, the third group [25] employs the local group lasso penalty to promote sparsity and smoothness together.

Unfortunately, the dynamic graphical models inferred by all three categories of methods are sensitive to the tuning parameters, including the kernel bandwidth and the penalty parameters that control the sparsity and smoothness. Classical brute-force grid search approaches for selecting these parameters are cross validation (CV), Akaike information criterion (AIC) and Bayesian information criterion (BIC) [14], [15], [17], [18], [22], [23], [24], [25]. However, heavy computational burdens come along with these methods; the learning algorithm needs to be run once for every combination of all possible values of the tuning parameters in a predefined candidate set before the one associated with the largest score is chosen. Moreover, it has been demonstrated in [26] that these parameter selection approaches yield unsatisfactory results for graphical model selection, especially when the number of variables is large. Apart from the large number of runs for parameter selection, the computational cost in each run is also large. The time complexity of the current three groups of methods is $\mathcal{O}(NP^3)$, where $P$ denotes the dimension (i.e., number of variables) and $N$ denotes the sample size. In practice, these methods are fraught with difficulties of daunting computational cost when tackling problems with more than 100 variables.

To address these problems, we propose a novel approach named BASS (BAyesian learning of graphical models with Smooth Structural changes) to learn the time-varying graphical models that is free of tuning while having a low time complexity of $\mathcal{O}(NP^2)$. In particular, we focus on Gaussian graphical models, and consequently, our objective is to infer the time-varying precision matrix. To this end, we impose a temporally dependent spike and slab prior [27], [28] on the off-diagonal entries of the precision matrix at each time point. Specifically, each off-diagonal entry of the precision matrix can be factorized as the product of a Bernoulli and a Gaussian distributed variable; the former is coupled over time via a binary Markov chain while the latter a Gauss-Markov chain (i.e., a thin-membrane model [29]). To facilitate the derivation of the variational inference algorithm, we replace the exact likelihood of the precision matrix at each time instant by the pseudo-likelihood that consists of the conditional distributions of one variable conditioned on the remaining variables. We then develop an efficient variational inference algorithm based on natural gradients to learn the variational distribution of the time-varying precision matrix. Due to the use of the pseudo-likelihood and the mean-field approximation in the variational inference, the time complexity of BASS is only $\mathcal{O}(NP^2)$. To cope with the problem of local maxima during the variational inference, we resort to simulated annealing [30] and propose a method based on bootstrapping to generate the annealing noise. Numerical results show that when compared with the three groups of frequentist methods, BASS achieves better performance in terms of structure estimation with significantly

less amount of computational time. We further apply BASS to construct financial networks from the stock return data of 78 banks worldwide during the 2008 Great recession. We find that the network becomes denser during the crisis, with clear peaks during the Great financial crisis and each wave of the subsequent European debt crisis.

Interestingly, BASS can be extended to inferring graphical models for multiple *stationary* time series in the frequency domain in a straightforward manner. Before explaining this approach, we briefly review the relevant literature on graphical models for stationary time series below. In [31], it is shown that for jointly Gaussian time series, the conditional independencies between time series are encoded by the common zeros in the inverse spectral density matrices at all frequencies. Given this insight, hypothesis tests are then performed in [31], [32], [33], [34] to test the conditional independence between every pair of time series. However, such methods are limited to problems with low dimensions and the true graphical model cannot be very sparse. On the other hand, Bach and Jordan [36] further show that by leveraging the Whittle approximation [35] the Fourier transform of the time series at a certain frequency can be regarded as samples drawn from the complex Gaussian distribution whose covariance matrix is the spectral density matrix at the same frequency. As a result, an appealing approach is to first estimate the smoothed spectral density matrix given the Fourier transform of the time series and then to infer the sparse inverse spectral density matrix by maximizing the $\ell_1$ norm penalized likelihood [37]. However, this approach requires extensive tuning. A variant of this approach for autoregressive processes is proposed in [38]. Apart from the frequentist methods, Bayesian methods have also been proposed in [39]. Unfortunately, this method can only learn decomposable graphs from the data. It is also quite time-consuming since Monte-Carlo Markov Chain is used to learn the Bayesian model. Note that the time complexity of all aforementioned methods is at least $\mathcal{O}(NP^3)$ for $P$-variate time series with length $N$. In this paper, in analogy to estimating the time-varying inverse covariance matrix, we learn the frequency-varying inverse spectral density matrix using BASS based on the Fourier transform of the time series, and then define the graphical model for the multivariate time series by identifying the common zero pattern of all inverse spectral density matrices. Different from the aforementioned methods, BASS is tuning free and scales gracefully with the dimension with time complexity $\mathcal{O}(NP^2)$. We compare BASS with the frequentist method GMS (graphical model selection) proposed in [37] on synthetic data. Similar to the results in the time domain, BASS can better recover the true graphs while being more efficient. We further apply BASS to the scalp EEG signals of patients at early stages of Alzheimer's disease (AD), and build a classifier based on the estimated graphical models to differentiate between the AD patients and age-matched control subjects. The classification accuracy resulting from BASS is higher than that from GMS.

This paper is structured as follows. We present our Bayesian model for time-varying graphical models in Section 2 and derive the natural gradient variational inference algorithm in Section 3. We then extend the proposed model to the frequency domain to infer graphical models for

stationary time series in Section 4. In Section 5, we show the numerical results for both synthetic and real data. Finally, we close this paper with conclusions in Section 6.

## 2 TIME-VARYING GRAPHICAL MODELS

We are concerned with undirected graphical models $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in this paper, where $\mathcal{V}$ denotes a set of vertices relating to variables and $\mathcal{E}$ denotes the edge set that encodes the conditional dependencies between the variables. 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: $p(x_j, x_k | x_{-jk}) = p(x_j | x_{-jk}) p(x_k | x_{-jk})$, where $-jk$ denotes all the nodes in $\mathcal{V}$ except $j$ and $k$. When all variables $x = [x_1, \ldots, x_P]'$ are jointly Gaussian distributed, the resulting graphical model is referred to as a Gaussian graphical model. Let $\mathcal{N}(\mu, \Sigma)$ denote a Gaussian distribution with mean $\mu$ and covariance $\Sigma$. The distribution can be equivalently parameterized as $\mathcal{N}(K^{-1}h, K^{-1})$, where $K = \Sigma^{-1}$ is the precision matrix (the inverse covariance) and $h = K\mu$ is the potential vector. The density function is

$$p(x) \propto \det(K)^{\frac{1}{2}} \exp\left( -\frac{1}{2} x' K x + h' x \right), \tag{1}$$

where $x'$ denotes the transpose of $x$. Under this scenario, the conditional dependencies are characterized by the precision matrix, that is, $x_j$ and $x_k$ are conditionally independent if and only if $K_{jk} = 0$. As a result, for Gaussian graphical models, we target at inferring a sparse precision matrix.

For time-varying graphical models, we assume that the observation $x^{(t)}$ at time $t$ is drawn from the graphical model with precision matrix $K^{(t)}$ for $t = 1, \ldots, N$, and $K^{(t)}$ changes smoothly with $t$. Without loss of generality, we further assume that $\mu^{(t)} = 0$ in our model, and so $h^{(t)} = 0$. The likelihood of $K^{(t)}$ can then be expressed as

$$p(x^{(t)} | K^{(t)}) \propto \det(K^{(t)})^{\frac{1}{2}} \exp\left( -\frac{1}{2} x^{(t)'} K^{(t)} x^{(t)} \right). \tag{2}$$

To facilitate the derivation of the variational inference algorithm, we propose to use the pseudo-likelihood instead of the exact likelihood (2) in the Bayesian formulation. More specifically, the pseudo-likelihood is derived from the conditional distributions of one variable $x_j$ conditioned on the remaining variables $x_{-j}$:

$$
\begin{aligned}
&p(x_j^{(t)} | x_{-j}^{(t)}, K_{jj}^{(t)}, K_{j,-j}^{(t)}) \\
&\propto \sqrt{K_{jj}^{(t)}} \exp\left[ -\frac{1}{2} K_{jj}^{(t)} \left( x_j^{(t)} + K_{jj}^{(t)-1} K_{j,-j}^{(t)} x_{-j}^{(t)} \right)^2 \right], \\
&\propto \sqrt{K_{jj}^{(t)}} \exp\left[ -\frac{1}{2} K_{jj}^{(t)} x_j^{(t)2} - x_j^{(t)} K_{j,-j}^{(t)} x_{-j}^{(t)} \right. \\
&\qquad \left. -\frac{1}{2} K_{jj}^{(t)-1} \left( K_{j,-j}^{(t)} x_{-j}^{(t)} \right)^2 \right],
\end{aligned}
\tag{3}
$$

where $K_{j,-j}$ denotes row $j$ in $K$ excluding $K_{jj}$, and $-K_{jj}^{(t)-1} K_{j,-j}^{(t)} x_{-j}^{(t)}$ and $K_{jj}^{(t)-1}$ are respectively the mean and the variance of the conditional distribution $p(x_j | x_{-j})$. Here, we regard (3) as a Gaussian distribution of $x_j$ whose mean and variance are parameterized by $K_{jj}^{(t)}$ and $K_{j,-j}^{(t)}$. In other words, it is a likelihood function of $K_{jj}^{(t)}$ and $K_{j,-j}^{(t)}$. This

pseudo-likelihood of $K$ has been frequently explored in the literature [3], [4], [5], [6], [7], [18], [25], for Gaussian graphical model selection. It simplifies the determinant term in (2) that typically leads to heavy computational burden of $\mathcal{O}(P^3)$, and so improves the computational efficiency. Indeed, the time complexity of the proposed method BASS is only $\mathcal{O}(P^2)$ w.r.t. (with regard to) $P$, owing to the pseudo-likelihood. Furthermore, the pseudo-likelihood typically results in more accurate and robust results when learning the graph structure [6], [7], [25]. Different from previous works [3], [4], [5], [6], [7], [25] that allow $K_{jk}$ to be different from $K_{kj}$, we assume that $K_{jk} = K_{kj}$ in our paper. Additionally, we infer the distribution of the diagonal elements $K_{jj}$ explicitly from the observed data $x$ rather than setting $K_{jj} = 1$ as in [3], [4], [5], [6], [7], [25]. For compactness in notation, we denote the distribution in (3) as $p(x_j^{(t)} | K_{jj}^{(t)}, K_{j,-j}^{(t)})$ in the sequel.

Next, we impose priors on both $K_{jj}^{(t)}$ and $K_{j,-j}^{(t)}$ in order to construct a full Bayesian model. We first focus on the off-diagonal elements $K_{jk}^{(t)}$. To guarantee that the off-diagonal parts of the precision matrices $K^{(t)}$ are sparse while varying smoothly across time, we resort to the temporally dependent spike and slab prior [27], [28]. Concretely, a spike and slab prior on $K_{jk}^{(t)}$ can be defined as [40]

$$K_{jk}^{(t)} \sim \pi_{jk}^{(t)} \mathcal{N}(\mu_{jk}^{(t)}, v_{jk}^{(t)}) + \left( 1 - \pi_{jk}^{(t)} \right) \delta_0, \tag{4}$$

where $\mathcal{N}(\mu_{jk}^{(t)}, v_{jk}^{(t)})$ is a Gaussian distribution with mean $\mu_{jk}^{(t)}$ and variance $v_{jk}^{(t)}$, $\delta_0$ is a Kronecker delta function, and $\pi_{jk}^{(t)} \in [0, 1]$ determines the probability of $K_{jk}^{(t)} = 0$ (i.e., the spike probability). By decreasing $\pi_{jk}^{(t)}$ to 0, this prior would shrink $K_{jk}^{(t)}$ to 0, thus encouraging sparsity in $K^{(t)}$. The above expression can also be equivalently written as [27]

$$K_{jk}^{(t)} = s_{jk}^{(t)} J_{jk}^{(t)} \tag{5}$$

$$J_{jk}^{(t)} \sim \mathcal{N}\left( \mu_{jk}^{(t)}, v_{jk}^{(t)} \right) \tag{6}$$

$$s_{jk}^{(t)} \sim \mathrm{Ber}\left( \pi_{jk}^{(t)} \right), \tag{7}$$

where $\mathrm{Ber}(\pi_{jk}^{(t)})$ is a Bernoulli distribution with success probability $\pi_{jk}^{(t)}$. To obtain $K^{(t)}$ that changes smoothly with $t$, we impose smoothness priors on both $s_{jk}^{(t)}$ and $J_{jk}^{(t)}$. For $s_{jk}^{(t)}$, we assume that it is drawn a binary Markov chain defined by the initial state and the transition probabilities

$$p\left( s_{jk}^{(1:N)} \right) = p(s_{jk}^{(1)}) \prod_{t=2}^{N} p\left( s_{jk}^{(t)} | s_{jk}^{(t-1)} \right), \tag{8}$$

where

$$p\left( s_{jk}^{(1)} \right) = \pi_1^{\delta(s_{jk}^{(1)}=1)} \left( 1 - \pi_1 \right)^{\delta(s_{jk}^{(1)}=0)}, \tag{9}$$

$$
\begin{aligned}
p\left( s_{jk}^{(t)} | s_{jk}^{(t-1)} \right) &= A_{00}^{\delta(s_{jk}^{(t-1)}=0, s_{jk}^{(t)}=0)} \left( 1 - A_{00} \right)^{\delta(s_{jk}^{(t-1)}=0, s_{jk}^{(t)}=1)} \\
&\cdot A_{11}^{\delta(s_{jk}^{(t-1)}=1, s_{jk}^{(t)}=1)} \left( 1 - A_{11} \right)^{\delta(s_{jk}^{(t-1)}=1, s_{jk}^{(t)}=0)},
\end{aligned}
\tag{10}
$$

and $\delta(\cdot)$ denotes the indicator function that yields 1 when the condition in the bracket is satisfied and 0 otherwise. We
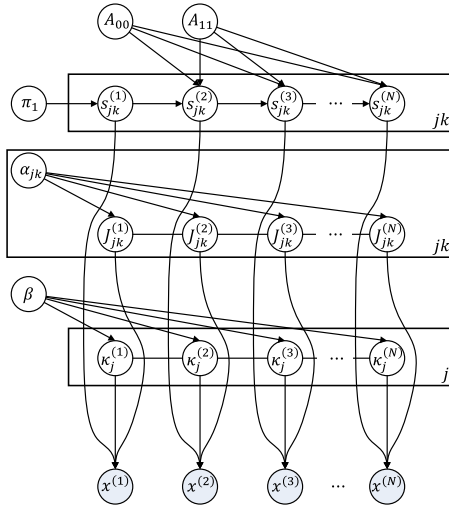
Fig. 1. Graph representation of BASS.

further assume $\pi_1$, $A_{00}$, and $A_{11}$ follow uniform distributions $\mathrm{Be}(1,1)$, where $\mathrm{Be}(1,1)$ denotes a Beta distribution with shape parameters one.

On the other hand, we assume that $J_{jk}^{(t)}$ forms a Gauss-Markov chain, in particular, a thin-membrane model [29]. The resulting prior on $J_{jk}^{(t)}$ can be expressed as

$$
p\left(J_{jk}^{(1:N)}\right) \propto \alpha_{jk}^{\frac{N-1}{2}}\exp\left[-\frac{\alpha_{jk}}{2}\sum_{t=2}^{N}\left(J_{jk}^{(t)}-J_{jk}^{(t-1)}\right)^2\right],
$$
$$
\propto \alpha_{jk}^{\frac{N-1}{2}}\exp\left(-\frac{\alpha_{jk}}{2}J_{jk}^{(1:N)\prime}K_{\mathrm{TM}}J_{jk}^{(1:N)}\right), \qquad (11)
$$

where $\alpha_{jk}$ is the smoothness parameter and $\alpha_{jk}K_{\mathrm{TM}}$ is the precision matrix of this Gaussian graphical model. We further impose a non-informative Jeffrey's prior on $\alpha_{jk}$, that is, $p(\alpha_{jk}) \propto 1/\alpha_{jk}$. The difference between $J_{jk}^{(t-1)}$ and $J_{jk}^{(t)}$ at every two consecutive time points $t-1$ and $t$ is controlled by the smoothness parameter $\alpha_{jk}$, suggesting that $\alpha_{jk}$ determines the smoothness of $J_{jk}^{(t)}$ across $t$. We also notice that $K_{\mathrm{TM}}$ is the graph Laplacian matrix corresponding to the Markov chain: the diagonal entry $[K_{\mathrm{TM}}]_{jj}$ equals the number of neighbors of node $j$, while the off-diagonal entry $[K_{\mathrm{TM}}]_{jk}$ equals $-1$ if node $j$ and $k$ are adjacent and 0 otherwise. As a result, $K_{\mathrm{TM}}$ is a tri-diagonal matrix in our case. Furthermore, it follows from the properties of the Laplacian matrix that $K_{\mathrm{TM}}\mathbf{1} = 0$, where $\mathbf{1}$ denotes a vector of all ones. In other words, the thin-membrane model is invariant to the addition of $c\mathbf{1}$, where $c$ is an arbitrary constant, and it allows the deviation from any overall mean level without having to specify the overall mean level itself. Such desirable properties make the thin-membrane model a popular smoothness prior in practice.

For diagonal entries in the time-varying precision matrix, since they can only take positive values, we reparameterize $K_{jj}^{(t)}$ as $K_{jj}^{(t)} = \exp(\kappa_j^{(t)})$. To promote the smooth variation of $\kappa_j^{(t)}$ across $t$, we assume that $\kappa_j^{(1:N)}$ follows a thin-membrane model with smoothness parameter $\beta$. We also impose the Jeffrey's prior on $\beta$.

Altogether, the proposed Bayesian model is summarized as a graphical model in Fig. 1. The joint distribution of all variables can be factorized as

$$
p(\boldsymbol{x}^{(1:N)}, \boldsymbol{s}^{(1:N)}, J^{(1:N)}, \boldsymbol{\kappa}^{(1:N)}, \pi_1, A_{00}, A_{11}, \boldsymbol{\alpha}, \beta)
$$
$$
= \prod_{j=1}^{P}\prod_{t=1}^{N}p(x_j^{(t)}|\kappa_j^{(t)}, J_{j,-j}^{(t)}, \boldsymbol{s}_{j,-j}^{(t)})
$$
$$
\cdot \prod_{j=1}^{P}\prod_{k=j+1}^{P}\left[p(\boldsymbol{s}_{jk}^{(1:N)}|\pi_1, A_{00}, A_{11})p(J_{jk}^{(1:N)}|\alpha_{jk})p(\alpha_{jk})\right]
$$
$$
\cdot \prod_{j=1}^{P}p(\boldsymbol{\kappa}_j^{(1:N)}|\beta)p(\pi_1)p(A_{00})p(A_{11})p(\beta). \qquad (12)
$$

## 3 VARIATIONAL INFERENCE

In this section, we develop a variational inference algorithm to learn the above Bayesian model. We first derive the low-complexity variational inference algorithm for BASS. Since the variational inference algorithm is often sensitive to local maxima, we further present how to utilize simulated annealing to help the algorithm escape from local maxima. We refer the readers to the summary of variational inference and natural gradients in Section A in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2022.3140886, before diving into the calculations in the sequel.

### 3.1 Variational Inference for BASS

Our objective is to approximate the intractable posterior distribution $p(\boldsymbol{s}^{(1:N)}, J^{(1:N)}, \boldsymbol{\kappa}^{(1:N)}, \pi_1, A_{00}, A_{11}, \boldsymbol{\alpha}, \beta|\boldsymbol{x}^{(1:N)})$ by a tractable variational distribution, that is, $q(\boldsymbol{s}^{(1:N)}, J^{(1:N)}, \boldsymbol{\kappa}^{(1:N)}, \pi_1, A_{00}, A_{11}, \boldsymbol{\alpha}, \beta)$. Specifically, we apply the mean-field approximation and factorize the variational distribution as

$$
q(\boldsymbol{s}^{(1:N)}, J^{(1:N)}, \boldsymbol{\kappa}^{(1:N)}, \pi_1, A_{00}, A_{11}, \boldsymbol{\alpha}, \beta)
$$
$$
= \prod_{j=1}^{P}\prod_{k=j+1}^{P}\left[q\left(\boldsymbol{s}_{jk}^{(1:N)}\right)q\left(J_{jk}^{(1:N)}\right)q(\alpha_{jk})\right]\prod_{j=1}^{P}q\left(\boldsymbol{\kappa}_j^{(1:N)}\right)q(\pi_1)
$$
$$
q(A_{00})q(A_{11})q(\beta). \qquad (13)
$$

For ease of exposition, we present the proposed algorithm from the perspective of message passing [41]. In a hierarchical Bayesian model, the update for each node only requires information (i.e., messages) from the nodes in its Markov blanket, including this node's parents, children, and co-parents of its children.

Given the Bayesian model in Fig. 1, the update rule of $q(\boldsymbol{s}_{jk}^{(1:N)})$ can be derived by passing messages in the directions of the red dashed arrows in Fig. 2a, resulting in

$$
q\left(\boldsymbol{s}_{jk}^{(1:N)}\right) \propto \exp\left[\sum_{t=1}^{N}\varphi_{\mathcal{V}}^s\left(s_{jk}^{(t)}\right) + \sum_{t=2}^{N}\varphi_{\mathcal{E}}^s\left(s_{jk}^{(t)}, s_{jk}^{(t-1)}\right)\right],
$$
$$
\qquad (14)
$$

where the node potential $\varphi_{\mathcal{V}}^s$ and edge potential $\varphi_{\mathcal{E}}^s$ are defined in (S.14)[1] and (S.15) in Table S.1 in the supplementary material, that are functions of those nodes in the Markov blanket of $\boldsymbol{s}_{jk}^{(1:N)}$. Moreover, it is apparent that $q(\boldsymbol{s}_{jk}^{(1:N)})$ can be factorized as a binary Markov Chain. As such, the marginal

---

1. S stands for equations, tables, and figures in the supplementary material.
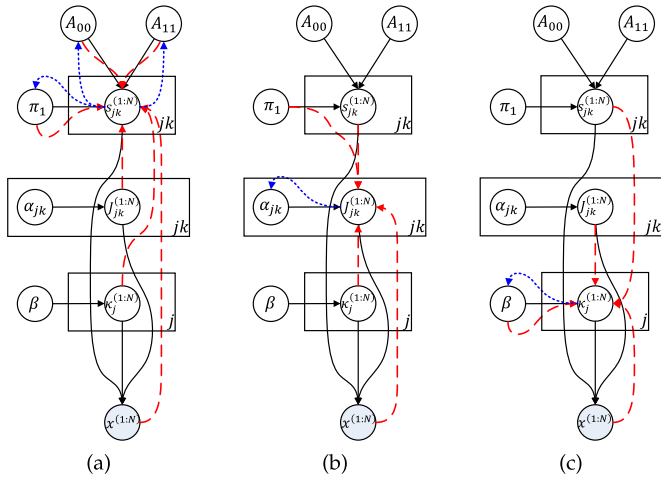
Fig. 2. Message passing scheme for updating (a) $q(s_{jk}^{(1:N)})$ (the red dashed arrows), $q(\pi_1)$, $q(A_{00})$, and $q(A_{11})$ (the blue dotted arrows), (b) $q(J_{jk}^{(1:N)})$ (the red dashed arrows) and $q(\alpha_{jk})$ (the blue dotted arrow), and (c) $q(\kappa_j^{(1:N)})$ (the red dashed arrows) and $q(\beta)$ (the blue dotted arrow).

and pairwise densities, $q(s_{jk}^{(t)})$ and $q(s_{jk}^{(t)}, s_{jk}^{(t-1)})$, can then be computed via message passing in the binary Markov chain (i.e., the forward-backward algorithm) with time complexity $\mathcal{O}(N)$. Given $q(s_{jk}^{(t)})$ and $q(s_{jk}^{(t)}, s_{jk}^{(t-1)})$, the variational distributions of the initial state $\pi_1$ and the transition probabilities $A_{00}$ and $A_{11}$ can be derived by receiving messages from $s_{jk}^{(t)}$ (see blue dotted arrows in Fig. 2a)

$$q(\pi_1) = \text{Be}(a, b), \tag{15}$$

$$q(A_{00}) = \text{Be}(c_0, d_0), \tag{16}$$

$$q(A_{11}) = \text{Be}(c_1, d_1). \tag{17}$$

where the shape parameters $a$, $b$, $c_0$, $d_0$, $c_1$, and $d_1$ of the Beta distributions can be updated as in (S.22)-(S.27).

Similarly, as denoted by the red dashed arrows in Fig. 2b, $q(J_{jk}^{(1:N)})$ can be updated by collecting messages from nodes in its Markov blanket, that is

$$q\left(J_{jk}^{(1:N)}\right) \propto \exp\left[\sum_{t=1}^{N} \varphi_{\mathcal{V}}^{J}\left(J_{jk}^{(t)}\right) + \sum_{t=2}^{N} \varphi_{\mathcal{E}}^{J}\left(J_{jk}^{(t)}, J_{jk}^{(t-1)}\right)\right], \tag{18}$$

where the node potentials $\varphi_{\mathcal{V}}^{J}(J_{jk}^{(t)})$ and edge potentials $\varphi_{\mathcal{E}}^{J}(J_{jk}^{(t)}, J_{jk}^{(t-1)})$ are defined in (S.16) and (S.17) in the supplementary material. The above expression corresponds to a Gauss-Markov chain, and therefore, the mean and variance for each $J_{jk}^{(t)}$ and the pairwise covariance of $J_{jk}^{(t)}$ and $J_{jk}^{(t-1)}$ can be obtained via message passing (a.k.a. belief propagation) with complexity $\mathcal{O}(N)$. We can then update the variational distribution of $\alpha_{jk}$ by receiving messages from $J_{jk}^{(1:N)}$ (see blue dotted arrows in Fig. 2b) as

$$q(\alpha_{jk}) = \text{Ga}\left(\frac{N-1}{2}, \frac{\sum_{t=2}^{N}\langle(J_{jk}^{(t)} - J_{jk}^{(t-1)})^2\rangle}{2}\right), \tag{19}$$

where $\text{Ga}(a, b)$ denotes a Gamma distribution with shape parameter $a$ and rate parameter $b$.

Finally, let us turn our attention to $\kappa^{(1:N)}$. The message updates for $\kappa_j^{(1:N)}$ are indicated by the red dashed arrows in Fig. 2c. Note that the prior and likelihood are not conjugate in this case. As mentioned in the Section A in the supplementary material, we need to specify the functional form of

$q(\kappa_j^{(1:N)})$, compute the natural gradients, and then update the natural parameters of $q(\kappa_j^{(1:N)})$ following (21). Here, we choose $q(\kappa_j^{(1:N)})$ to be Gaussian. Owing to the thin-membrane priors on $\kappa_j^{(1:N)}$, $q(\kappa_j^{(1:N)})$ is also associated with a Gauss-Markov chain, in the same fashion as $q(J_{jk}^{(1:N)})$. Therefore, we can parameterize $q(\kappa_j^{(1:N)})$ as

$$q\left(\kappa_j^{(1:N)}\right) \propto \exp\left[\sum_{t=1}^{N} \varphi_{\mathcal{V}}^{\kappa}\left(\kappa_j^{(t)}\right) + \sum_{t=2}^{N} \varphi_{\mathcal{E}}^{\kappa}\left(\kappa_j^{(t)}, \kappa_j^{(t-1)}\right)\right]$$

$$\propto \exp\left[\sum_{t=1}^{N}\left(-\frac{\Omega_{t,t}}{2}\kappa_j^{(t)^2} + h_t \kappa_j^{(t)}\right) - \sum_{t=2}^{N}\Omega_{t,t-1}\kappa_j^{(t)}\kappa_j^{(t-1)}\right], \tag{20}$$

where $\Omega$ denotes the $N \times N$ tri-diagonal precision matrix and $h$ is the $N$-dimensional potential vector. In light of (21), $\Omega$ and $h$ can be updated as in (S.19)-(S.21). Note that the step size $\rho$ in (S.19)-(S.21) is chosen via line search. After obtaining the mean, the variance and the pairwise covariance of $\kappa_j^{(1:N)}$ via message passing in the Gauss-Markov chain, we can update $q(\beta)$ as $q(\beta) = \text{Ga}((N-1)P/2, \sum_{j=1}^{P}\sum_{t=2}^{N}\langle(\kappa_j^{(t)} - \kappa_j^{(t-1)})^2\rangle/2)$. Detailed derivation of the variational inference algorithm can be found in Section B in the supplementary material.

## 3.2 Time Complexity

We notice that the most expensive operations in the update rules in Table S.1, are the products $\langle K_{j,-jk}^{(t)}\rangle x_{-jk}^{(t)}$ and $\langle K_{k,-jk}^{(t)}\rangle x_{-jk}^{(t)}$ in (S.12) and $\langle K_{j,-j}^{(t)}\rangle x_{-j}^{(t)}$ in (S.18). The time complexity of these operations is $\mathcal{O}(P)$. The last product is used for updating the diagonal element $\kappa_j^{(t)}$, and hence, the time complexity for updating all $NP$ diagonal elements in $K^{(1:N)}$ is $\mathcal{O}(NP^2)$. On the other hand, the first two products are used for updating one off-diagonal element $K_{jk}^{(t)}$. Take into account all $\mathcal{O}(NP^2)$ off-diagonal elements in $K^{(1:N)}$, and the overall time complexity of BASS should be $\mathcal{O}(NP^3)$. However, instead of computing these products every time when updating an off-diagonal element $K_{jk}^{(t)}$, we can first keep a record of $\langle K_{j,-j}^{(t)}\rangle x_{-j}^{(t)}$ for $j = 1, \ldots, P$ at the beginning of BASS. Next, for each off-diagonal element $K_{jk}^{(t)}$, the products can be computed as $\langle K_{j,-jk}^{(t)}\rangle x_{-jk}^{(t)} = \langle K_{j,-j}^{(t)}\rangle x_{-j}^{(t)} - \langle K_{jk}^{(t)}\rangle x_k^{(t)}$ and likewise for $\langle K_{k,-jk}^{(t)}\rangle x_{-jk}^{(t)}$. After updating the variational distribution of this off-diagonal element, we can then update the record as $\langle K_{j,-j}^{(t)}\rangle x_{-j}^{(t)} = \langle K_{j,-jk}^{(t)}\rangle x_{-jk}^{(t)} + \langle K_{jk}^{(t)}\rangle x_k^{(t)}$ and likewise for $\langle K_{k,-k}^{(t)}\rangle x_{-k}^{(t)}$. As a consequence, we can cycle through all off-diagonal elements in $K^{(1:N)}$ without recomputing the products every time. The resulting time complexity of BASS can be reduced to $\mathcal{O}(NP^2)$.

## 3.3 Simulated Annealing

As introduced in Section A in the supplementary material, the update rules derived in Section 3.1 can also be written in the manner of natural gradient ascent as

$$\theta^{\{i+1\}} = (1-\rho)\theta^{\{i\}} + \rho\nabla_{\eta}\mathcal{L}(\theta^{\{i\}}), \tag{21}$$

where $\theta$ denotes the parameters to be updated, $i$ denotes the number of iterations $\nabla_{\eta}\mathcal{L}(\theta^{\{i\}})$ denotes the natural gradient of the evidence lower bound (ELBO) w.r.t. $\theta$, and $0 < \rho < 1$ is the step size. Since the ELBO is non-convex, the

above variational inference algorithm suffers from the issue of local maxima. To counteract this problem, we employ simulated annealing and modify the update rule of the variational inference in (21) as [30]

$$\tilde{\theta} = (1 - \rho)\theta^{\{i\}} + \rho \left[ \frac{1}{T^{\{i\}}} \nabla_\eta \mathcal{L}_1(\theta^{\{i\}}) + \left( 1 - \frac{1}{T^{\{i\}}} \right) \epsilon^{\{i\}} \right],$$
(22)

where $T^{\{i\}}$ denotes the annealing temperature in iteration $i$, and $\epsilon^{\{i\}}$ denotes the annealing noise vector. Note that $T^{\{i\}} \to 1$ as $i \to \infty$. The update $\tilde{\theta}$ is accepted with the probability

$$p(\theta^{\{i+1\}} = \tilde{\theta}) = \min \left\{ 1, \exp \left[ \frac{\mathcal{L}(\tilde{\theta}) - \mathcal{L}(\theta^{\{i\}})}{1 - 1/T^{\{i\}}} \right] \right\},$$
(23)

otherwise $\theta^{\{i+1\}} = \theta^{\{i\}}$. When the temperature $T^{\{i\}}$ is high, $\tilde{\theta}$ is sufficiently volatile to avoid shallow local maxima. As $T^{\{i\}}$ decreases to 1 , the algorithm mimics the original variational inference and converges.

Next, we discuss how to build the noise vector $\epsilon^{\{i\}}$. Currently, there is no generic rule for specifying the noise distribution. In this work, we only add noise when updating the natural parameters of $q(s_{jk}^{(1:N)})$, $q(J_{jk}^{(1:N)})$, and $q(\kappa_j^{(1:N)})$, since the update rules for these parameters are more complicated than the others and so the resulting estimates are more likely to converge to local maxima. Let $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}})$ denote the natural gradient of $\mathcal{L}_1$ in (S.11) w.r.t. the natural parameters $\theta$ of $q(s_{jk}^{(1:N)})$, $q(J_{jk}^{(1:N)})$, and $q(\kappa_j^{(1:N)})$. It can be observed from the corresponding update rules (cf. Eqs. (S.12)-(S.21) in Table S.1) that $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}})$ can be decomposed into two terms: $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}}) = \nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, x^{(1:N)}) + \nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, \xi)$, where $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, x^{(1:N)})$ is a function of the observations $x^{(1:N)}$, and $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, \xi)$ is a function of the hyperparameters $\xi = \{\pi_1, A_{00}, A_{11}, \alpha, \beta\}$. Correspondingly, we decompose the noise vector as $\epsilon^{\{i\}} = \epsilon^{\{i\}}(x^{(1:N)}) + \epsilon^{\{i\}}(\xi)$.

For the hyperparameters $\xi$, we utilize their variational distributions to obtain the noise $\epsilon^{\{i\}}(\xi)$. Concretely, we draw a random sample $\tilde{\xi}$ of $\xi$ from the variational distribution in each iteration, and compute the noise vector as $\epsilon^{\{i\}}(\xi) = \nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, \tilde{\xi})$. On the other hand, for the function of the observations, let $\nabla_\eta \tilde{\mathcal{L}}_1(\theta^{\{i\}}, x^{(1:N)}) = \nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, x^{(1:N)}) / T^{\{i\}} + (1 - 1/T^{\{i\}})\epsilon^{\{i\}}(x^{(1:N)})$ denote the noisy gradient that is corrupted by the annealing noise. As $T^{\{i\}} \to 1$, $\nabla_\eta \tilde{\mathcal{L}}_1 (\theta^{\{i\}}, x^{(1:N)})$ becomes less noisy and converges to the exact gradient $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, x^{(1:N)})$. Here, instead of proposing a distribution for $\epsilon^{\{i\}}(x^{(1:N)})$, we specify the noisy gradient $\nabla_\eta \tilde{\mathcal{L}}_1(\theta^{\{i\}}, x^{(1:N)})$ directly by replacing the observations $x^{(1:N)}$ in $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, x^{(1:N)})$ with its bootstrapped sample set $\tilde{x}^{(1:N)}$. Such bootstrapped sets are often used for time-invariant graphical model selection [26], [42], [43] so as to find a network that is robust to bootstrapping. We borrow this idea and provide an empirical distribution for $x^{(1:N)}$ by bootstrapping the original observations. More specifically, for each time point $t$, we set $\tilde{x}^{(t)} = x^{(\tau)}$ by sampling $\tau$ uniformly from a window around $t$ with width $w$, namely, $\{t - w, t - w + 1, \ldots, t + w\}$. The noisy gradient can then be computed as $\nabla_\eta \tilde{\mathcal{L}}_1(\theta^{\{i\}}, x^{(1:N)}) = \nabla_\eta \tilde{\mathcal{L}}_1(\theta^{\{i\}}, \tilde{x}^{(1:N)})$. Furthermore, we set $w = (1 - 1/T^{\{i\}})N/2$ such that the variance of the noisy gradient $\nabla_\eta \tilde{\mathcal{L}}_1(\theta^{\{i\}}, x^{(1:N)})$ decreases with the

decreases of $T$ and $\nabla_\eta \tilde{\mathcal{L}}_1(\theta^{\{i\}}, x^{(1:N)})$ converges to the exact gradient $\nabla_\eta \mathcal{L}_1(\theta^{\{i\}}, x^{(1:N)})$ as $T^{\{i\}} \to 1$.

In practice, we increase $R^{\{i\}} = 1/T^{\{i\}}$ instead of decreasing $T^{\{i\}}$ as $i$ increases. Concretely, we first specify the number of iterations for annealing as $N_a$. We then begin the algorithm with $R^{\{1\}} = 0$ (i.e., $T^{\{1\}} = \infty$) and increases $R^{\{i\}}$ by $10/N_a$ in every 10th iteration. After $N_a$ iterations, $R^{\{N_a\}} = 1$ (i.e., $T^{\{N_a\}} = 1$). In our experiments, we set $N_a = 500$ unless stated otherwise. A brief summary of the proposed BASS algorithm is presented in Algorithm 1. For the detailed implementation of the proposed simulated annealing technique, we refer the readers to Algorithm S.1 in the supplementary material.

---

**Algorithm 1.** BASS

---

**Input:** The observations $x_{1:P}^{(1:N)}$ and the number of iterations for annealing $N_a$.
**Output:** The estimated precision matrices $\langle K^{(1:N)} \rangle$.
**for** i = 1 to $N_a$ **do**
  - Compute $w = (1 - R^{\{i\}})N/2$.
  - Generate $\tilde{x}_{1:P}^{(1:N)}$ by bootstrapping from $x_{1:P}^{(1:N)}$ with window width $w$.
  - Sample $\tilde{\pi}_1$, $\tilde{A}_{00}$, $\tilde{A}_{11}$, $\tilde{\alpha}_{jk}$, and $\tilde{\beta}$ from their variational distributions.
  - Update $q(s_{jk}^{(1:N)})$, $q(J_{jk}^{(1:N)})$, and $q(\kappa_j^{(1:N)})$ following the proposed simulated annealing mechanism in Section 3.3.
  - Update $q(\pi_1)$, $q(A_{00})$, $q(A_{11})$, $q(\alpha_{jk})$, $q(\beta)$ following their update rules as in Section 3.1.
  **end for**
  **repeat**
  - Update all variational distributions following their update rules as in Section 3.1.
  **until** convergence.

---

In order to demonstrate the usefulness of the proposed simulated annealing approach, we depict the convergence results of BASS without and with simulated annealing in Fig. 3. Here, we consider a synthetic data set. The true time-varying graph corresponding to this data set is given. We then randomly select 100 true edges (cf. red dashed lines) and 100 absent edges (cf. green solid lines) from the true graph and check how the corresponding $\langle s_{jk}^{(t)} \rangle$ changes as the algorithm proceeds. The initial values for all parameters are the same in both cases. We can tell from Fig. 3a that the true and false edges are partially mixed with each other in the original variational inference algorithm without annealing. By contrast, in Fig. 3b, after using the proposed simulated annealing approach, the red dashed lines clearly stand above the green lines as BASS proceeds. Note that $\langle s_{jk}^{(t)} \rangle$ denotes the existence probability of edge $(j, k)$ at time $t$. The estimated $\langle s_{jk}^{(t)} \rangle$ given by BASS with annealing is close to 1 for true edges and 0 for the false ones, successfully separating the true and false edges in an automated manner.

## 4 GRAPHICAL MODELS FOR STATIONARY TIME SERIES

In this section, we discuss how to exploit BASS to learn interactions among $P$ univariate stationary Gaussian processes (i.e., time series) $y_{1:P}^{(1:N)}$. A graphical model $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
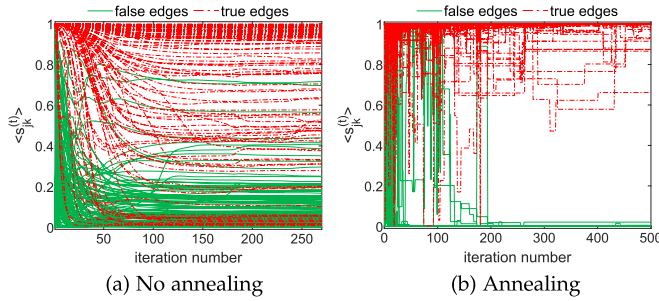
Fig. 3. Estimation of 200 randomly selected $\langle s_{jk}^{(t)}\rangle$ (i.e., zero pattern of the off-diagonal elements in $K^{(1:T)}$) as a function of iteration number $i$ for BASS when applying to synthetic data with $P = 20$ and $N = 1000$ without and with simulated annealing. Half of the selected off-diagonal elements correspond to the absent edges in the true graph and the other half correspond to the true edges.

for $\boldsymbol{y}^{(t)}$ can be constructed by letting an abscent edge $(j, k) \notin \mathcal{E}$ denote that the two entire time series $\boldsymbol{y}_j^{(1:N)}$ and $\boldsymbol{y}_k^{(1:N)}$ are conditionally independent given the remaining collection of time series $\boldsymbol{y}_{-jk}^{(1:N)}$ [44], that is

$$\mathrm{cov}\left(\boldsymbol{y}_j^{(t)}, \boldsymbol{y}_k^{(t+\tau)}|\boldsymbol{y}_{-jk}^{(1:N)}\right) = 0, \quad \forall \tau. \tag{24}$$

In other words, the lagged conditional covariance equals 0 for all time lags $\tau$. On the other hand, the conditional dependence can also be defined in the frequency domain of the time series. Concretely, we first define the spectral density matrix as the Fourier transform of the lagged covariance matrix $\mathrm{cov}(\boldsymbol{y}^{(t)}, \boldsymbol{y}^{(t+\tau)})$

$$S^{(\omega)} = \sum_\tau \mathrm{cov}(\boldsymbol{y}^{(t)}, \boldsymbol{y}^{(t+\tau)})\exp(-i\omega\tau), \tag{25}$$

for $\omega \in [0, 2\pi]$. Let $K^{(\omega)} = [S^{(\omega)}]^{-1}$, the conditional independence between $\boldsymbol{y}_j^{(1:N)}$ and $\boldsymbol{y}_k^{(1:N)}$ holds if and only if [37], [44]

$$K_{jk}^{(\omega)} = 0, \quad \forall \omega. \tag{26}$$

This suggests that one common zero entry in the inverse spectral density matrices across a certain frequency band is equivalent to the conditional independence between the corresponding two time series in this frequency band. Therefore, for a multivariate time series, we aim to infer the inverse spectral density matrices $K^{(\omega)}$.

Here, we follow the state-of-the-art Whittle approximation framework [35]: Suppose that $\boldsymbol{f}_{1:P}^{(\omega)}$ is the discrete Fourier transform of $\boldsymbol{y}_{1:P}^{(1:N)}$ at frequency $\omega$

$$f_j^{(\omega)} = \sum_t y_j^t \exp(-i\omega t), \tag{27}$$

then $\boldsymbol{f}_{1:P}^{(\omega)}$ are independent complex Gaussian random variables with mean zero and precision matrix given by the inverse spectral density matrix $K^{(\omega)}$ at the same frequency

$$\boldsymbol{f}_{1:P}^{(\omega)} \sim \mathcal{N}_c\left(\boldsymbol{0}, K^{(\omega)-1}\right). \tag{28}$$

As a result, we can learn $K^{(\omega)}$ that changes smoothly with $\omega$ from $\boldsymbol{f}^{(\omega)}$ using BASS. In this scenario, the covariate is the frequency $\omega$. It should be stressed that the complex Gaussian distribution in (28) can be written as

$$p(\boldsymbol{f}^{(\omega)}|K^{(\omega)}) \propto \det(K^{(\omega)})\exp\left(-\boldsymbol{f}^{(\omega)*}K^{(\omega)}\boldsymbol{f}^{(\omega)}\right), \tag{29}$$

where $\boldsymbol{f}^{(\omega)*}$ denotes the complex conjugate transpose of $\boldsymbol{f}^{(\omega)}$. The above density function does not have the operation of square root as in the density function of the Gaussian distribution for real numbers (2). The corresponding pseudo-likelihood of $K^{(\omega)}$ can then be expressed as

$$p(f_j^{(\omega)}|K_{jj}^{(\omega)}, K_{j,-j}^{(\omega)}) \propto K_{jj}^{(\omega)}\exp\Big[-K_{jj}^{(\omega)}f_j^{(\omega)}\overline{f_j^{(\omega)}}$$
$$-\overline{f_j^{(\omega)}}K_{j,-j}^{(\omega)}\boldsymbol{f}_{-j}^{(\omega)} - f_j^{(\omega)}\overline{K_{j,-j}^{(\omega)}\boldsymbol{f}_{-j}^{(\omega)}}$$
$$-K_{jj}^{(\omega)-1}K_{j,-j}^{(\omega)}\boldsymbol{f}_{-j}^{(\omega)}\overline{K_{j,-j}^{(\omega)}\boldsymbol{f}_{-j}^{(\omega)}}\Big], \tag{30}$$

where $\overline{f_j^{(\omega)}}$ is the complex conjugate of $f_j^{(\omega)}$. In addition, the prior distribution on $J_{jk}^{(\omega)}$ is also a complex Gaussian distribution. We therefore modify the ELBO $\mathcal{L}$ and the corresponding update rules accordingly. The detailed update rules are summarized in Table S.2 in the supplementary material.

## 5 EXPERIMENTAL RESULTS

In this section, we compare the proposed BASS algorithm with the state-of-the-art methods in the literature. Specifically, for the problem of learning time-varying graphical models, we consider three benchmark methods:

1) KERNEL [13], [14], [15], [16]: Kernel-smoothed covariance matrices $S^{(t)}$ are first estimated, and then a graphical model is inferred at each time point by solving the graphical lasso problem

$$K^{(t)} = \underset{K^{(t)}\succeq 0}{\arg\min}\ \mathrm{tr}(S^{(t)}K^{(t)}) - \log\det K^{(t)} + \lambda_1\|K^{(t)}\|_1,$$

where $\lambda_1$ controls the sparsity of $K^{(t)}$.

2) SINGLE [17], [19], [20]: It further controls the smoothness of $K^{(t)}$ over $t$ by imposing penalty on the difference between the precision matrices at every two consecutive time points

$$K^{(t)} = \underset{K^{(t)}\succeq 0}{\arg\min} \sum_{t=1}^N \Big[\mathrm{tr}(S^{(t)}K^{(t)}) - \log\det K^{(t)}$$
$$+ \lambda_1\|K^{(t)}\|_1\Big] + \lambda_2\sum_{t=2}^N \|K^{(t)} - K^{(t-1)}\|_1. \tag{31}$$

3) LOGGLE [25]: It exploits the local group lasso penalty to promote the sparsity of $K^{(t)}$ and the smoothness across time simultaneously

$$K^{(t)} = \underset{K^{(t)}\succeq 0}{\arg\min} \sum_t\ \mathrm{tr}(S^{(t)}K^{(t)}) - \log\det K^{(t)}$$
$$+ \lambda_1\sum_{j\neq k}\sqrt{\sum_{\tau\in\mathcal{D}(t,d)}K_{jk}^{(\tau)2}}, \tag{32}$$

where $\mathcal{D}(t, d) = \{\tau : |\tau - t| \le d\}$ denotes neighborhood around $t$ with width $d$. The exact likelihood in the above expression is replaced by the pseudo-likelihood in the implementation to achieve better performance [25].

The time complexity of the above three methods is $\mathcal{O}(NP^3)$. On the other hand, for the problem of learning

TABLE 1
Graph Recovery Results From Different Methods for Synthetic Data With Different Dimensions
($P = 20, 100, 500, N = 1000, N_e = P$)

| Methods | $P = 20$ | | | | $P = 100$ | | | | $P = 500$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) |
| BASS | 0.89 (5.06e-2) | 0.86 (2.31e-2) | 0.88 (3.12e-2) | 1.76e2 (2.39) | 0.82 (4.06e-2) | 0.77 (3.89e-2) | 0.80 (2.68e-2) | 4.73e3 (2.24e1) | 0.74 (1.02e-2) | 0.69 (1.11e-2) | 0.72 (1.01e-2) | 1.33e5 (3.59e3) |
| KERNEL (oracle) | 0.83 (1.75e-2) | 0.88 (4.37e-2) | 0.85 (1.88e-2) | 4.81e1 (7.24) | 0.71 (1.94e-2) | 0.79 (2.02e-2) | 0.75 (7.21e-3) | 4.50e4 (8.20e2) | 0.66 (2.45e-3) | 0.49 (6.39e-3) | 0.58 (4.50e-3) | 5.80e6 (1.38e5) |
| KERNEL (CV) | 0.63 (1.18e-2) | 0.91 (1.91e-2) | 0.74 (1.32e-2) | 6.51e2 (2.58e1) | 0.71 (1.70e-2) | 0.79 (2.23e-2) | 0.75 (9.03e-3) | 6.61e4 (1.07e3) | 0.86 (1.72e-1) | 0.29 (1.94e-1) | 0.38 (1.75e-1) | 8.59e6 (1.47e5) |
| SINGLE (oracle) | 0.86 (2.58e-2) | 0.90 (3.62e-2) | 0.88 (2.56e-2) | 1.04e4 (3.19e2) | 0.84 (1.26e-2) | 0.74 (2.10e-2) | 0.79 (1.01e-2) | 1.58e6 (2.99e5) | | | | |
| SINGLE (CV) | 0.86 (2.54e-2) | 0.89 (3.84e-2) | 0.88 (2.79e-2) | 5.09e4 (2.17e3) | 0.68 (1.60e-2) | 0.51 (1.41e-2) | 0.58 (7.53e-3) | 6.94e6 (1.04e6) | | | | |
| LOGGLE (oracle) | 0.89 (1.17e-2) | 0.86 (3.98e-2) | 0.88 (1.89e-2) | 1.75e4 (1.03e3) | 0.81 (1.51e-2) | 0.78 (2.10e-2) | 0.80 (9.59e-3) | 1.51e6 (3.60e5) | | | | |
| LOGGLE (CV) | 0.71 (2.18e-2) | 0.93 (3.36e-2) | 0.81 (1.78e-2) | 5.08e4 (1.75e3) | 0.47 (1.15e-2) | 0.80 (2.31e-2) | 0.59 (1.41e-3) | 5.07e6 (4.59e5) | | | | |

*The standard deviations are shown in the brackets.*

graphical models for stationary time series in frequency domain, we compare BASS with the GMS approach proposed in [37], which can be regarded as the counterpart of KERNEL in the frequency domain.

## 5.1 Time-Varying Graphical Models (Time Domain)

### 5.1.1 Synthetic Data

Given the dimension $P$, the number of time points $N$, and the average number of edges $N_e$, we simulate synthetic Gaussian distributed data from time-varying graphical models as follows. We first generate the off-diagonal elements in the precision matrices $K^{(1:N)}$ as

$$K_{jk}^{(t)} = A_{jk} \sin\left(\frac{\pi t}{2N}\right) + B_{jk} \cos\left(\frac{\pi t}{2N}\right) + C_{jk} \sin\left(\pi\left(\frac{t}{N} + D_{jk}\right)\right), \quad (33)$$

where for all $j$ and $k$ $A_{jk}$, $B_{jk}$, and $C_{jk}$ are drawn uniformly from $[-1, -0.5] \cup [0.5, 1]$, and $D_{jk}$ follows a uniform distribution in $[-0.25, 0.25]$. We then set a threshold and zero out the off-elements whose magnitude is smaller than the threshold such that the average number of edges in $K^{(1:N)}$ is $N_e$. Next, we compute the diagonal entries as $K_{jj}^{(t)} = \sum_{k \neq j} |K_{jk}^{(t)}| + 0.1$ to guarantee the positive definiteness of $K^{(1:N)}$. Finally, we draw a sample $x^{(t)}$ at each time point $t$ from $\mathcal{N}(0, K^{(t)-1})$.

We compare all methods in terms of precision, recall, $F_1$-score, and computational time. 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 correctly estimated edges to all the edges in the true graph; $F_1$-score is defined as $2 \cdot$ precision $\cdot$ recall/(precision+recall), which is a weighted average of the precision and recall. For the benchmark methods, since the ground truth is given, we first select the tuning parameters that maximize the $F_1$-score

and refer to the results as oracle results (a.k.a. optimal results). We also show the results when the tuning parameters are selected via cross validation (CV), which is commonly used in existing works [14], [17], [23], [24], [25]. More specifically, for all three frequentist methods, we choose the kernel bandwidth $h$ from 5 candidates $\{\exp(-4)N, \exp(-3)N, \ldots, N\}$ and $\lambda_1$ from 5 candidates $\{\exp(-4), \exp(-3), \ldots, 1\}$. We further select $\lambda_2$ from $\{\exp(-4), \exp(-2), \ldots, \exp(4)\}$ for SINGLE and $d$ from $\{\exp(-4)N, \exp(-3)N, \ldots, N\}$ for LOGGLE. We show the results for graphs with different dimensions $P$, different sample sizes $N$, and different graph densities (characterized by the average number of edges $N_e$) respectively in Tables 1, 2, and 3. The results are averaged over 5 trials and the standard deviation is presented in the brackets. All methods are implemented using R and Rcpp.

We observe that BASS typically obtains the highest $F_1$-score with the shortest period of computational time, regardless of the dimension, the sample size, and the graph density. In other words, BASS can well recover the time-varying graph structure in an automated fashion. On the other hand, the oracle results of LOGGLE and SINGLE are comparable to that of BASS. However, in practice, we have no information on the true graphs and therefore cannot choose the tuning parameters that maximize the $F_1$-score (i.e., minimize the difference between the true and estimated graphs). As mentioned before, one practical approach to choosing the tuning parameters is CV. Unfortunately, as can be observed from Tables 1, 2, and 3, the CV results are typically worse than the oracle results, indicating that CV cannot always find the optimal tuning parameters. Since we only consider 5 candidates for each tuning parameter and so the distance between every two combinations of tuning parameters is relatively large, the wrong choice of the optimal parameters may lead to a large gap between the CV and the oracle results. Additionally, we can see that LOGGLE and SINGLE outperform KERNEL in

TABLE 2
Graph Recovery Results From Different Methods for Synthetic Data With Different Sample Size
($P = 20$, $N = 500, 1000, 2000$, $N_e = P$)

| Methods | $N = 500$ | | | | $N = 1000$ | | | | $N = 2000$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) |
| BASS | 0.83 (7.02e-2) | 0.71 (2.52e-2) | 0.77 (2.86e-2) | 9.75e1 (1.85) | 0.89 (5.06e-2) | 0.86 (2.31e-2) | 0.88 (3.12e-2) | 1.76e2 (2.39) | 0.94 (4.30e-2) | 0.90 (4.98e-2) | 0.92 (2.20e-2) | 3.44e2 (6.24) |
| KERNEL (oracle) | 0.62 (4.25e-2) | 0.84 (3.21e-2) | 0.71 (1.69e-2) | 2.33e2 (5.56) | 0.83 (1.75e-2) | 0.88 (4.37e-2) | 0.85 (1.88e-2) | 4.81e1 (7.24) | 0.88 (2.02e-2) | 0.91 (2.32e-2) | 0.89 (1.34e-2) | 1.11e3 (4.48e1) |
| KERNEL (CV) | 0.53 (1.78e-2) | 0.79 (1.21e-1) | 0.61 (1.50e-1) | 3.12e2 (5.05) | 0.63 (1.18e-2) | 0.91 (1.91e-2) | 0.74 (1.32e-2) | 6.51e2 (2.58e1) | 0.84 (1.82e-2) | 0.93 (1.48e-2) | 0.88 (1.15e-2) | 1.37e3 (3.10e1) |
| SINGLE (oracle) | 0.69 (5.37e-2) | 0.82 (4.23e-2) | 0.75 (1.67e-2) | 4.17e3 (5.81e1) | 0.86 (2.58e-2) | 0.90 (3.62e-2) | 0.88 (2.56e-2) | 1.04e4 (3.19e2) | 0.92 (1.23e-2) | 0.92 (2.31e-2) | 0.92 (9.80e-3) | 2.85e4 (4.99e2) |
| SINGLE (CV) | 0.70 (4.81e-2) | 0.77 (6.95e-2) | 0.73 (2.55e-2) | 2.39e4 (1.49e3) | 0.86 (2.54e-2) | 0.89 (3.84e-2) | 0.88 (2.79e-2) | 5.09e4 (2.17e3) | 0.92 (1.23e-2) | 0.92 (2.31e-2) | 0.92 (9.80e-3) | 1.44e5 (3.81e3) |
| LOGGLE (oracle) | 0.71 (3.24e-2) | 0.83 (3.80e-2) | 0.77 (4.77e-3) | 8.28e3 (3.20e2) | 0.89 (1.17e-2) | 0.86 (3.98e-2) | 0.88 (1.89e-2) | 1.75e4 (1.03e3) | 0.93 (2.50e-2) | 0.92 (3.02e-2) | 0.92 (1.93e-2) | 3.82e4 (3.25e3) |
| LOGGLE (CV) | 0.59 (8.98e-2) | 0.79 (5.29e-2) | 0.67 (7.40e-2) | 2.01e4 (7.29e2) | 0.71 (2.18e-2) | 0.93 (3.36e-2) | 0.81 (1.78e-2) | 5.08e4 (1.75e3) | 0.80 (9.23e-2) | 0.94 (3.49e-2) | 0.86 (4.60e-2) | 1.22e5 (1.05e4) |

*The standard deviations are shown in the brackets.*

TABLE 3
Graph Recovery Results From Different Methods for Synthetic Data With Different Density ($P = 20$, $N = 1000$, $N_e = 10, 20, 40$)

| Methods | $N_e = 10$ | | | | $N_e = 20$ | | | | $N_e = 40$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) |
| BASS | 0.90 (8.89e-2) | 0.94 (2.16e-2) | 0.92 (4.63e-2) | 1.68e2 (3.18) | 0.89 (5.06e-2) | 0.86 (2.31e-2) | 0.88 (3.12e-2) | 1.76e2 (2.39) | 0.85 (4.57e-2) | 0.66 (5.84e-2) | 0.74 (2.36e-2) | 1.63e2 (1.80) |
| KERNEL (orcale) | 0.86 (1.33e-1) | 0.88 (1.29e-1) | 0.86 (2.52e-2) | 4.51e2 (7.54) | 0.83 (1.75e-2) | 0.88 (4.37e-2) | 0.85 (1.88e-2) | 4.81e2 (7.24) | 0.90 (2.93e-2) | 0.59 (6.16e-2) | 0.71 (4.27e-2) | 4.39e2 (7.26) |
| KERNEL (CV) | 0.77 (2.90e-1) | 0.80 (1.72e-1) | 0.73 (1.16e-1) | 6.02e2 (2.92e1) | 0.63 (1.18e-2) | 0.91 (1.91e-2) | 0.74 (1.32e-2) | 6.51e2 (2.58e1) | 0.58 (2.00e-1) | 0.82 (1.36e-1) | 0.65 (4.88e-2) | 5.57e2 (2.23e1) |
| SINGLE (oracle) | 0.87 (2.58e-2) | 0.90 (3.56e-2) | 0.89 (2.56e-2) | 1.04e4 (3.19e2) | 0.86 (2.58e-2) | 0.90 (3.62e-2) | 0.88 (2.56e-2) | 1.04e4 (3.19e2) | 0.87 (3.55e-2) | 0.61 (5.38e-2) | 0.72 (3.82e-2) | 9.83e3 (1.90e2) |
| SINGLE (CV) | 0.67 (7.38e-2) | 0.93 (2.42e-2) | 0.78 (4.51e-2) | 5.43e4 (5.52e3) | 0.86 (2.54e-2) | 0.89 (3.84e-2) | 0.88 (2.79e-2) | 5.09e4 (2.17e3) | 0.65 (2.26e-1) | 0.75 (1.20e-1) | 0.66 (7.79e-2) | 4.79e4 (1.60e3) |
| LOGGLE (oracle) | 0.86 (7.43e-2) | 0.89 (8.43e-2) | 0.88 (3.11e-2) | 1.77e4 (1.36e3) | 0.89 (1.17e-2) | 0.86 (3.98e-2) | 0.88 (1.89e-2) | 1.75e4 (1.03e3) | 0.86 (8.88e-3) | 6.53 (5.24e-2) | 0.74 (3.60e-2) | 1.62e4 (9.66e2) |
| LOGGLE (CV) | 0.55 (7.16e-2) | 0.98 (2.18e-2) | 0.70 (5.83e-2) | 5.16e4 (2.52e3) | 0.71 (2.18e-2) | 0.93 (3.36e-2) | 0.81 (1.78e-2) | 5.08e4 (1.75e3) | 0.76 (1.58e-1) | 0.70 (7.45e-2) | 0.72 (6.97e-2) | 5.01e4 (1.89e3) |

*The standard deviations are shown in the brackets.*

terms of $F_1$-score after capturing the temporal dependence across $K^{(t)}$. Nevertheless, they become prohibitively slow when $P \geq 100$, severely hindering their application to large-scale problems. By contrast, BASS improves the estimation accuracy while being more efficient. Indeed, as shown in Fig. 4, the computational time of BASS approximately increases *quadratically* with $P$, whereas that of the other methods is approximately a *cubic* function of $P$. These observations are in agreement with the theoretical time complexity. On the other hand, the computational time of all methods increases linearly with $N$ (see Table 2), while the graph density does not have much effect on the computational time (see Table 3). Finally, we find that the performance of BASS and all oracle results in terms of $F_1$-score improves as the dimension $P$ decreases, the graph density $N_e$ decreases or the sample size $N$ increases. In general, more information is required to reliably estimate the true graphs when $P$ or $N_e$ increases, but the sample size $N$ remains invariant in these cases, thus degrading the performance of these methods. On the other hand, according to Eq. (33), the precision matrix $K^{(t)}$ changes more slowly with $t$ as the sample size $N$ increases, and so it becomes easier to estimate the time-varying graphs. This explains the improvement of the $F_1$-score with increasing $N$.

### 5.1.2 Stock Return Data of 78 Banks

We infer time-varying financial networks from the daily stock return data during the 2008 Great Recession (from 2005 to 2013) of 78 banks in 25 countries around the world. Learning financial networks from data greatly helps to
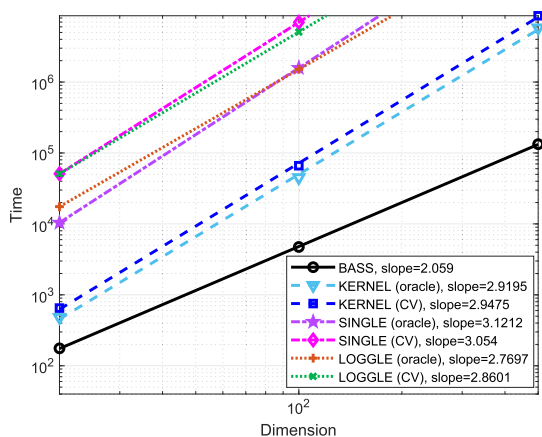
model the system risk in a financial market [45]; such networks can be utilized to find the interactions between financial institutions for risk contagion that impair the system stability as well as to determine which institutions are more contagious or subject to contagion. The 78 banks are those in the world's top 150 (by assets) whose data are publicly available at Yahoo Finance (https://finance.yahoo.com/) from 2005 to 2013. In total, we have 2348 observations for each of the 78 variables. $6.49\%$ of the data are missing at random. The proposed Bayesian method, BASS, can deal with the missing data, by inferring their variational distributions along with the distribution of the precision matrices, whereas the frequentist methods cannot. As such, we use the data imputed by BASS as the input to KERNEL, SINGLE, and LOGGLE. The tuning parameters in these methods are selected by CV. Before applying all methods, we normalize the data to have unit variance. The time-varying variance for data normalization is inferred via kernel smoothing, and the kernel bandwidth is chosen via CV.

The estimated number of edges in the time-varying networks as a function of time resulting from different methods is summarized in Fig. 5a. The critical point during the 2008 global financial crisis is the Lehman bankruptcy on Sept. 15, 2008 when the US government started to implement a huge stimulus package so as to alleviate the influence of the financial crisis [46]. As demonstrated in Fig. 5a, the number of edges resulting from all methods tends to increase before the Lehman bankruptcy and then decreases gradually after it. In the viewpoint of system risk, all banks in the financial system would progressively suffer from the financial crisis due to the risk contagion, leading to similar stock price movement and more connections in the networks during the financial crisis [47]. Apart from the major peak in 2008, BASS further yields another two peaks in 2010 and 2011. These two peaks correspond to the Greek debt crisis of 2009-2010 and the subsequent sovereign debt crisis of 2011 in Europe. Interestingly, Demirer *et al.* [48] analyzed the volatilities of 96 banks in the world's top 150 and further constructed time-varying networks by inferring sparse vector autoregressive approximation models in sliding windows of the volatility data. The three peaks were also observed in their experiment; the range of the three peaks in [48] is shown in Fig. 5a. Here we learn networks from the stock return data by means of BASS, instead of extracting them from volatilities, yet we obtain similar results. As opposed to BASS, the three peaks are not very obvious for KERNEL, SINGLE, and LOGGLE. The results of these methods may become better by selecting the tuning parameters from a larger set of candidates, at the expense of



Fig. 4. Computational time as a function of dimension $P$. We fit a straight line to the logarithm of average run time in Table 1 versus the logarithm of $P$, and compute the slope. The slope provides an empirical measure of the time complexity.
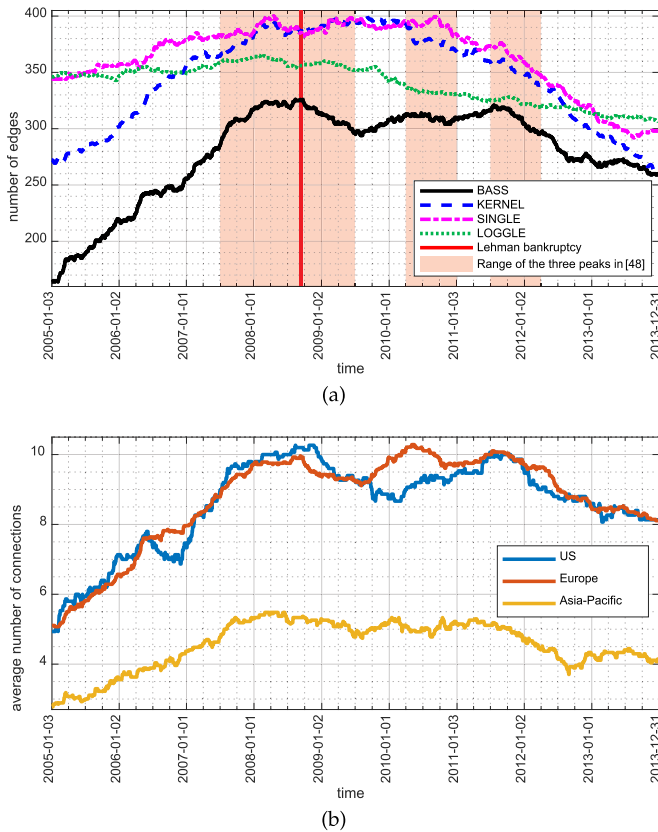
Fig. 5. Results of the stock return data of 78 banks: (a) the number of edges as a function of time resulting from different methods; (b) the average number of connections for banks in US, Europe, and Asia-Pacific given by BASS.

increased computational time. Unfortunately, the computational time for KERNEL, SINGLE, and LOGGLE is already 1.48e5, 4.90e7, and 8.06e6 seconds respectively, and testing more candidates of the tuning parameters will make these methods distressingly slow. On the other hand, BASS only takes 3.08e4 seconds to converge. In summary, BASS can better capture the changes in the financial networks in less amount of run time.

Next, let us delve into the results given by BASS. We plot the estimated financial networks at the beginning of each year from 2005 to 2013 in Fig. 6. In this figure, the banks are clustered according to their regions automatically, as banks in the same region are expected to have more interactions. We further choose the cluster of US, Europe (including UK), and Asia-Pacific and depict the average number of connections for banks in each cluster as a function of time in Fig. 5b. Both Figs. 5b and 6 tell us that number of connections for US and European banks is larger than that of banks in Asia-Pacific. In light of the theory of system risk, the financial institutions with more connections are called central institutions (i.e., sit in the center of the financial system). Such institutions are more sensitive to financial crises, and conversely, their failure can lead to the breakdown of the entire system with a larger probability [45]. With this theory in mind, we can now analyze how the networks changed during the financial crisis. Before the Lehman bankruptcy, the average number of connections for US banks first increased in 2006, due to the Government's unexpected decision to tighten monetary policy in May and June that

year [48]. There was no other major shock in 2006 though, and therefore, the average number of connections for US banks decreased later in 2006 [48]. In early 2007, the collapse of several mortgage originators led to the sharp increase of connections in US [46]. Due to the large number of connections between banks in US and Europe, the European banks also lost a tremendous amount on toxic assets and from bad loans. Consequently, the connections for European banks also increased. In late 2007 and 2008, the 2007 subprime mortgage crisis in US finally led to the global financial crisis [46], since US sat in the center of the network. We can see that the entire financial network in 2008 (see Figs. 6d and 6e) is much denser than the one before and after the crisis (see Figs. 6a and 6j). In the post-Lehman period, the US market calmed after the government injected a massive amount of capital into major US banks [46]. Thus, the number of connections for US banks decreased correspondingly. On the other hand, the European debt crisis occurred in Greece in late 2009 and further spread to Ireland and Portugal [49]. The delay of the rescue package for Greece caused the second peak of connections for European banks in May 2010. Later in 2011, the debt crisis further affected Spain and Italy [49], leading to the third peak for Europe from June to August in 2011 in Fig. 5b. Note that the number of connections for US banks was low when the debt crisis first happened in 2009. However, due to risk contagion in the financial network, the number of connections for US banks also reached another peak in 2011. On the other hand, since there is a fewer number of connections between Asia-Pacific countries and US and Europe (see Fig. 6), the financial crisis did not impact Asia-Pacific countries as severely as US and Europe.

## 5.2 Graphical Models for Time Series (Frequency Domain)

### 5.2.1 Synthetic Data

To test the proposed method for inferring graphical models in the frequency domain, we consider simulated time series with length $N$ generated from a first order vector autoregressive process for $P$ variables. Specifically, we simulate data from the model

$$\mathbf{y}^{(t)} = A\mathbf{y}^{(t-1)} + \boldsymbol{\epsilon}^{(t)}, \tag{34}$$

where $\mathbf{y}^{(t)} \in \mathbb{R}^P$, $A \in \mathbb{R}^{P \times P}$, and $\boldsymbol{\epsilon}^{(t)} \sim \mathcal{N}(\mathbf{0}, I)$. The inverse spectral density of the process is then given by [38]:

$$K^{(\omega)} = I + A'A + \exp(-i\omega)A + \exp(i\omega)A'. \tag{35}$$

We consider time series with different dimensions, sample sizes, and graph density, and compare the results given by BASS with those of GMS [37]. The graph density is characterized by the number of non-zero elements in $A$, which is denoted as $N_e$ in the sequel. For GMS, the kernel bandwidth $h$ is selected from $\{\exp(-4)N, \exp(-2)N, \dots, \exp(4)N\}$ and $\lambda_1$ is chosen from $\{\exp(-4), \exp(-3), \dots, 1\}$. Again, we select the tuning parameters by maximizing the $F_1$-score between the estimated and true graphs (i.e., the oracle results) as well as using CV. Before applying these methods, we first normalize the data $\mathbf{y}_{1:P}^{(1:N)}$ to have unit variance. We then apply the normalized Fourier transform to obtain the
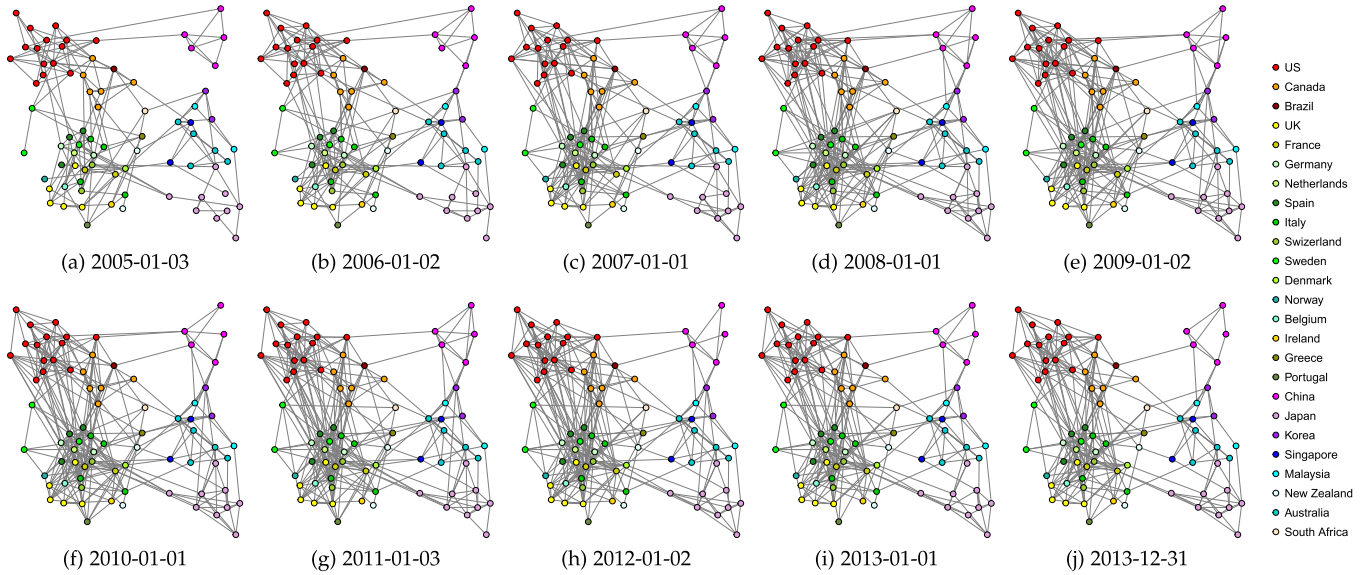
Fig. 6. Financial networks resulting from BASS from 2005 to 2013. The networks are time-varying. The banks are clustered according to their regions in an automatic fashion.

TABLE 4
Graph Recovery Results From BASS and GMS for Synthetic Time Series With Different Dimensions
($P = 20, 100$, $N = 1000$, $N_e = P$)

| Methods | $P = 20$ | | | | $P = 100$ | | | |
|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) |
| BASS | 0.90 (3.14e-2) | 0.88 (4.23e-2) | 0.89 (3.09e-2) | 2.18e2 (2.10e1) | 0.88 (4.90e-2) | 0.71 (3.92e-2) | 0.78 (4.20e-2) | 4.75e3 (2.85e2) |
| GMS (orcale) | 0.83 (1.05e-1) | 0.72 (8.27e-2) | 0.76 (7.17e-2) | 3.36e3 (5.13e1) | 0.71 (1.94e-2) | 0.79 (2.02e-2) | 0.75 (7.21e-3) | 4.16e5 (6.20e3) |
| GMS (CV) | 0.63 (1.02e-1) | 0.78 (1.20e-1) | 0.69 (8.18e-2) | 4.32e3 (7.93e2) | 0.71 (7.69e-2) | 0.68 (2.53e-2) | 0.69 (3.69e-2) | 5.05e5 (9.34e3) |

*The standard deviations are shown in the brackets.*

TABLE 5
Graph Recovery Results From BASS and GMS for Synthetic Time Series With Different Lengths
($P = 20$, $N = 500, 1000, 2000$, $N_e = P$)

| Methods | $N = 500$ | | | | $N = 1000$ | | | | $N = 2000$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) |
| BASS | 0.90 | 0.81 | 0.85 | 1.11e2 | 0.90 | 0.88 | 0.89 | 2.18e2 | 0.89 | 0.94 | 0.91 | 4.44e2 |
| | (6.51e-2) | (9.32e-2) | (6.74e-2) | (8.26) | (3.14e-2) | (4.23e-2) | (3.09e-2) | (2.10e1) | (8.40e-2) | (3.60e-2) | (5.74e-2) | (5.56e1) |
| GMS | 0.90 | 0.70 | 0.78 | 1.70e3 | 0.83 | 0.72 | 0.76 | 3.36e3 | 0.80 | 0.73 | 0.76 | 7.28e3 |
| (orcale) | (1.15e-1) | (1.09e-1) | (9.26e-2) | (9.68) | (1.05e-1) | (8.27e-2) | (7.17e-2) | (5.13e1) | (2.74e-1) | (1.12e-1) | (6.04e-2) | (5.12e2) |
| GMS (CV) | 0.50 | 0.76 | 0.53 | 2.76e3 | 0.63 | 0.78 | 0.69 | 4.32e3 | 0.75 | 0.73 | 0.74 | 9.01e3 |
| | (3.17e-1) | (1.61e-1) | (1.94e-1) | (1.55e2) | (1.02e-1) | (1.20e-1) | (8.18e-2) | (7.93e2) | (7.56e-2) | (1.03e-1) | (7.36e-2) | (9.79e2) |

*The standard deviations are shown in the brackets.*

Fourier coefficients $\boldsymbol{f}_{1:P}^{(1:N)}$ such that the variance of $\boldsymbol{f}_{1:P}^{(1:N)}$ does not increase with $N$. The results averaged over 5 trials are summarized in Tables 4, 5, and 6.

The results are similar to those in the time domain. Hence, we only provide a brief summary of the results here. BASS yields the best performance in the terms of the $F_1$-score with the least amount of computational time in all cases. The oracle results are the second best. CV cannot always select the optimal tuning parameters, and so the CV results are worse than the oracle results. We can also observe that the increase of dimension $P$ or the graph density $N_e$ deteriorates the performance, whereas the increase of the sample size $N$ improves the performance, as expected. The computational time of BASS

and GMS is approximately a linear function of $NP^2$ and $NP^3$ respectively.

### 5.2.2 Scalp EEG of AD Patients

In this section, we consider the problem of inferring functional brain networks from scalp EEG recordings. Specifically, we analyze two data sets. The first one contains 22 patients with mild cognitive impairment (MCI, a.k.a. pre-dementia) and 38 healthy control subjects [50]. The patients complained of the memory problem, and later on, they all developed mild AD (i.e., the first stage of AD). The second data set consists of 17 patients with mild AD and 24 control subjects [51]. For more details, please refer to Section C in the supplementary material.

TABLE 6
Graph Recovery Results From BASS and GMS for Synthetic Time Series With Different Graph Density
($P = 20$, $N = 1000$, $N_e = 10, 20, 40$)

| Methods | $N_e = 10$ | | | | $N_e = 20$ | | | | $N_e = 40$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) | Precision | Recall | $F_1$-score | Time(s) |
| BASS | 0.93 | 0.97 | 0.95 | 2.28e2 | 0.90 | 0.88 | 0.89 | 2.18e2 | 0.93 | 0.72 | 0.81 | 2.04e2 |
| | (1.16e-1) | (6.88e-2) | (6.49e-2) | (1.18e1) | (3.14e-2) | (4.23e-2) | (3.09e-2) | (2.10e1) | (4.20e-2) | (2.89e-2) | (2.37e-2) | (7.90) |
| GMS | 0.90 | 0.81 | 0.85 | 3.46e3 | 0.83 | 0.72 | 0.76 | 3.36e3 | 0.74 | 0.68 | 0.71 | 3.68e3 |
| (orcale) | (6.87e-2) | (2.64e-2) | (3.77e-2) | (9.44e1) | (1.05e-1) | (8.27e-2) | (7.17e-2) | (5.13e1) | (7.61e-2) | (2.10e-2) | (4.12e-2) | (3.32e1) |
| GMS | 0.79 | 0.80 | 0.79 | 4.70e3 | 0.63 | 0.78 | 0.69 | 4.32e3 | 0.74 | 0.68 | 0.71 | 5.03e3 |
| (CV) | (1.62e-1) | (3.60e-2) | (8.35e-2) | (7.87e2) | (1.02e-1) | (1.20e-1) | (8.18e-2) | (7.93e2) | (7.61e-2) | (2.10e-2) | (4.12e-2) | (7.12e2) |

*The standard deviations are shown in the brackets.*

Although AD cannot be cured at present, existing symptom-delaying medications are proven to be more effective at early stages of AD, such as MCI and mild AD [50]. On the other hand, scalp EEG recording systems are inexpensive and potentially mobile, thus making it a useful tool to screen a large population for the risk of AD. As a result, it is crucial to identify the patients from scalp EEG signals at early stages of AD. Learning functional brain networks can help to distinguish between patients and healthy people [52].

We first perform the normalized Fourier transform on all channels of EEG signals to obtain $f^{(\omega)}$. We only consider $f^{(\omega)}$ in the frequency band $4 - 30$Hz in order to filter out the noise in the signal. $K^{(\omega)}$ is then inferred from $f^{(\omega)}$ by applying the proposed BASS algorithm. We further split $K^{(\omega)}$ into three frequency ranges: $4 - 8$Hz, $8 - 12$Hz, and $12 - 30$Hz, as suggested by previous works on the same data sets [50], [51]. For each frequency band, we infer the corresponding graphical models by finding the common zero patterns of all $K^{(\omega)}$ for $\omega$ in this band. We compare BASS with GMS (CV) in this experiment to check which method can yield graphs that can better distinguish between the patients and the control subjects. The candidate set of the kernel bandwidth $h$ and the penalty parameter $\lambda_1$ for GMS (CV) is respectively $\{\exp(-3)N, \exp(-2.5)N, \ldots, \exp(-1)N\}$ and $\{\exp(-5), \exp(-4.5), \ldots, \exp(-0.5)\}$.

First, we count the number of edges in the graphical models, which can be regarded as a measure of synchrony between different EEG channels. We observe that graphical models in $4 - 8$Hz can best distinguish between patients and controls for both data sets and both methods. We depict in Figs. 7 and 8 the boxplots of the number of edges in the graphical models. Clearly, the graphical models for patients are more sparse than those for healthy

people, and this phenomenon becomes more pronounced for Mild AD patients. Such findings are consistent with the loss of synchrony within the EEG signals for AD patients as reported in the literature [50], [51]. We further conduct the Mann-Whitney test on the number of edges in the two sets of graphical models, respectively for the patients and the controls. The resulting p-value given by BASS for the two data sets is respectively $7.55 \times 10^{-3}$ and $1.49 \times 10^{-3}$, which are statistically significant. As a comparison, the p-value resulting from GMS is $3.99 \times 10^{-1}$ for the MCI data and $9.26 \times 10^{-2}$ for the Mild AD data. Next, we further train a random forest classifier based on the estimated brain networks to differentiate between the patients and the controls. The input to the classifier is the adjacency matrices associated with the networks. We then use the leave-one-out CV to evaluate the performance of the classifier. The accuracy yielded by BASS for the two data sets is $80.00\%$ and $85.37\%$ respectively, whereas that of GMS is $65.85\%$ and $70\%$. On the other hand, the computational time averaged over all subjects is $3.50 \times 10^2$ seconds for BASS and $1.40 \times 10^4$ seconds for GMS. Apparently, BASS can better describe the perturbations in the EEG synchrony for MCI and mild AD patients, while being more efficient.

## 6 CONCLUSION

In this paper, we propose a novel Bayesian model BASS to solve the problem of estimating dynamic undirected graphical models. In contrast to the existing methods that have a high time complexity of $\mathcal{O}(NP^3)$ and require extensive parameter tuning, the time complexity of BASS is only $\mathcal{O}(NP^2)$ and it is free of tuning. Specifically, we develop a
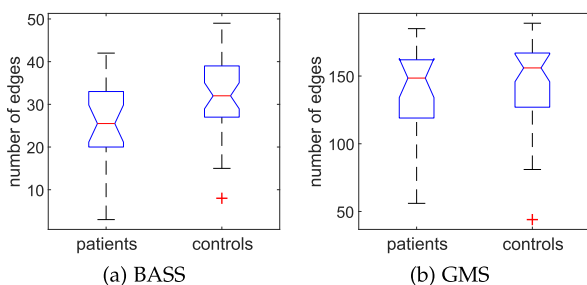


Fig. 7. Boxplots of the number of edges given by BASS and GMS for the first data set with MCI patients.
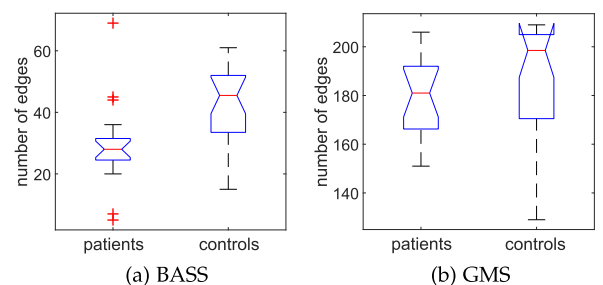


Fig. 8. Boxplots of the number of edges given by BASS and GMS for the second data set with Mild AD patients.

natural gradient based variational inference algorithm to learn the Bayesian model. To deal with the problem of local maxima, we resort to simulated annealing and propose to use bootstrapping to generate the annealing noise. Compared with the existing methods, BASS can better recover the true graphs with the least amount of computational time. We then apply BASS to analyze the stock return data of 78 banks worldwide, and observe that the resulting financial network becomes denser during the 2008 global financial crisis and the subsequent European debt crises. On the other hand, we find the resemblance between inferring time-varying inverse covariance matrices and frequency-varying inverse spectral density matrices, and extend BASS to learn graphical models among a multitude of stationary time series in the frequency domain. Results from EEG data of MCI and mild AD patients show that the proposed model may help to diagnose AD from scalp EEG at an early stage.

As BASS can only tackle Gaussian distributed data at present, we intend to extend it to non-Gaussian data by means of Gaussian copulas [53] in future work. Additionally, it is interesting to extend BASS to deal with piece-wise constant graphical models [52]. In this case, the data can be partitioned into a certain number of time segments. The graph within each segment remains unchanged, but the graphs for every two consecutive segments can be completely different. Such piece-wise constant graphical models also find wide applications in practice [12], [52].

## ACKNOWLEDGMENTS

## REFERENCES

[1] H. Yu and J. Dauwels, "Variational bayes learning of time-varying graphcial models," in *Proc. IEEE 26th Int. Workshop Mach. Learn. Signal Process.*, 2016, pp. 1–6.

[2] D. Koller and N. Friedman, *Probabilistic Graphical Models*. Cambridge, MA, USA: MIT Press, 2009.

[3] N. Meinshausen and B. Bühlmann, "High-dimensional graphs and variable selection with the lasso," *Ann. Statist.*, vol. 34, no. 3, pp. 1436–1462, 2006.

[4] M. Yuan, "High dimensional inverse covariance matrix estimation via linear programming," *J. Mach. Learn. Res.*, vol. 11, pp. 2261–2286, 2010.

[5] T. Sun and C. H. Zhang, "Sparse matrix inversion with scaled Lasso," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 3385–3418, 2013.

[6] S. Oh, O. Dalal, K. Khare, and B. Rajaratnam, "Optimization methods for sparse pseudo-likelihood graphical model selection," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 667–675.

[7] H. Liu and L. Wang, "TIGER: A tuning-insensitive approach for optimally estimating Gaussian graphical models," *Electron. J. Statist.*, vol. 11, no. 1, pp. 241–294, 2017.

[8] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical Lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[9] C. J. Hsieh, M. A. Sustik, I. S. Dhillon, and P. Ravikumar, "QUIC: Quadratic approximation for sparse inverse covariance estimation," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2911–2947, 2014.

[10] H. Yu, L. Xin, and J. Dauwels, "Variational Wishart approximation for graphical model selection: Monoscale and multiscale models," *IEEE Trans. Signal Process.*, vol. 67, no. 24, pp. 6468–6482, Dec. 2019.

[11] H. Yu, S. Wu, L. Xin, and J. Dauwels, "Fast Bayesian inference of sparse networks with automatic sparsity determination," *J. Mach. Learn. Res.*, vol. 21, no. 124, pp. 1–54, 2020.

[12] M. A. Kramer, U. T. Eden, E. D. Kolaczyk, R. Zepeda, E. N. Eskandar, and S. S. Cash, "Coalescence and fragmentation of cortical networks during focal seizures," *J. Neurosci.*, vol. 30, no. 30, pp. 10076–10085, 2010.

[13] S. Zhou, J. Lafferty, and L. Wasserman, "Time varying undirected graphs," *Mach. Learn.*, vol. 80, pp. 295–319, 2010.

[14] M. Kolar, L. Song, A. Ahmed, and E. P. Xing, "Estimating time-varying networks," *Ann. Appl. Math.*, vol. 4, no. 1, pp. 94–123, 2010.

[15] M. Kolar and E. P. Xing, "On time varying undirected graphs," *J. Mach. Learn. Res.*, vol. 15, pp. 407–415, 2011.

[16] H. Qiu and F. Han, "Joint estimation of multiple graphical models from high dimensional time series," *J. Roy. Statist. Soc. Ser. Statist. Methodol.*, vol. 78, no. 2, pp. 487–507, 2016.

[17] R. P. Monti, P. Hellyer, D. Sharp, R. Leech, C. Anagnostopoulos, and G. Montana, "Estimating time-varying brain connectivity networks from functional MRI time series," *NeuroImage*, vol. 103, pp. 427–443, 2014.

[18] A. Ahmed and E. P. Xing, "Recovering time-varying networks of dependencies in social and biological studies," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 29, pp. 11878–11883, 2009.

[19] A. J. Gibberd and J. D. B. Nelson, "High dimensional changepoint detection with a dynamic graphical lasso," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2014, pp. 2684–2688.

[20] S. Yang, Z. Lu, X. Shen, P. Wonka, and J. Ye, "Fused multiple graphical lasso," *SIAM J. Optim.*, vol. 25, no. 2, pp. 916–943, 2015.

[21] A. J. Gibberd and J. D. B. Nelson, "Estimating dynamic graphical models from multivariate time-series data," in *Proc. Int. Workshop Adv. Anal. Learn. Temporal Data*, 2015, pp. 111–128.

[22] E. C. Wit and A. Abbruzzo, "Inferring slowly-changing dynamic gene-regulatory networks," *BMC Bioinf.*, vol. 16, no. S6, 2015, Art. no. S5.

[23] A. J. Gibberd and J. D. Nelson, "Regularized estimation of piece-wise constant Gaussian graphical models: The group-fused graphical lasso," *J. Comput. Graphical Statist.*, vol. 26, no. 3, pp. 623–634, 2017.

[24] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 205–213.

[25] J. Yang and J. Peng, "Estimating time-varying graphical models," *J. Comput. Graphical Statist.*, vol. 29, no. 1, pp. 191–202, 2020.

[26] H. Liu, K. Roeder, and L. Wasserman, "Stability approach to regularization selection (StARS) for high dimensional graphical models," in *Proc. 24th Annu. Conf. Neural Inf. Process. Syst.*, 2010, pp. 1432–1440.

[27] J. Ziniel and P. Schniter, "Dynamic compressive sensing of time-varying signals via approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 21, pp. 5270–5284, Nov. 2013.

[28] M. R. Andersen, O. Winther, and L. K. Hansen, "Bayesian inference for structured spike and slab priors," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 1745–1753.

[29] H. Yu, J. Dauwels, and P. Johnathan, "Extreme-value graphical models with multiple covariates," *IEEE Trans. Signal Process.*, vol. 62, no. 21, pp. 5734–5747, Nov. 2014.

[30] S. Gultekin, A. Zhang, and J. Paisley, "Asymptotic simulated annealing for variational inference," in *Proc. IEEE Global Commun. Conf.*, 2018, pp. 1–7.

[31] R. Dahlhaus, "Graphical interaction models for multivariate time series," *Metrika*, vol. 51, no. 2, pp. 157–172, 2000.

[32] Y. Matsuda, "A test statistic for graphical modelling of multivariate time series," *Biometrika*, vol. 93 no. 2, pp. 399–409, 2006.

[33] R. J. Wolstenholme and A. T. Walden, "An efficient approach to graphical modeling of time series," *IEEE Trans. Signal Process.*, vol. 63, no. 12, pp. 3266–3276, Jun. 2015.

[34] J. K. Tugnait, "Edge exclusion tests for graphical model selection: Complex Gaussian vectors and time series," *IEEE Trans. Signal Process.*, vol. 67, no. 19, pp. 5062–5077, Oct. 2019.

[35] P. Whittle, "The analysis of multiple stationary time series," *J. Roy. Statist. Soc. Ser. Methodol.*, vol. 15, no. 1, pp. 125–139, 1953.

[36] F. R. Bach and M. I. Jordan, "Learning graphical models for stationary time series," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2189–2199, Aug. 2004.

[37] A. Jung, G. Hannak, and N. Goertz, "Graphical LASSO based model selection for time series," *IEEE Signal Process. Lett.*, vol. 22, no. 10, pp. 1781–1785, Oct. 2015.

[38] J. Songsiri and L. Vandenberghe, "Topology selection in graphical models of autoregressive processes," *J. Mach. Learn. Res.*, vol. 11, pp. 2671–2705, 2010.

[39] A. Tank, N. J. Foti, and E. B. Fox, "Bayesian structure learning for stationary time series," in *Proc. 31st Conf. Uncertainty Artif. Intell.*, 2015, pp. 872–881.

[40] H. Ishwaran and J. S. Rao, "Spike and slab variable selection: Frequentist and Bayesian strategies," *Ann. Statist.*, vol. 33, no. 2, pp. 730–773, 2005.

[41] C. Zhang, J. Butepage, H. Kjellström, and S. Mandt, "Advances in variational inference," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 2008–2026, Aug. 2019.

[42] N. Meinshausen and P. Bühlmann, "Stability selection," *J. Roy. Statist. Soc. Ser. Statist. Methodol.*, vol. 72, pp. 417–473, 2010.

[43] S. Li, L. Hsu, J. Peng, and P. Wang, "Bootstrap inference for network construction with an application to a breast cancer microarray study," *Ann. Appl. Statist.*, vol. 7, no. 1, pp. 391–417, 2013.

[44] J. Dauwels *et al.*, "Inferring brain networks through graphical models with hidden variables," in *Proc. 1st Int. Conf. Mach. Learn. Interpretation Neuroimag.*, 2012, pp. 194–201.

[45] M. Billio, M. Getmansky, A. W. Lo, and L. Pelizzon, "Econometric measures of connectedness and systemic risk in the finance and insurance sectors," *J. Financial Econ.*, vol. 104, no. 3, pp. 535–559, 2012.

[46] M. Williams, *Uncontrolled Risk: Lessons of Lehman Brothers and How Systemic Risk can Still Bring Down the World Financial System.* New York, NY, USA: McGraw Hill Professional, 2010.

[47] P. Glasserman, and H. P. Young, "Contagion in financial networks," *J. Econ. Literature*, vol. 54, no. 3, pp. 779–831, 2016.

[48] M. Demirer, F. X. Diebold, L. Liu, and K. Yilmaz, "Estimating global bank network connectedness," *J. Appl. Econometrics*, vol. 33, no. 1, pp. 1–15, 2018.

[49] V. Constancio, "Contagion and the European debt crisis," *Financial Stability Rev.*, vol. 16, pp. 109–121, 2012.

[50] J. Dauwels, F. Vialatte, T. Musha, and A. Cichocki, "A comparative study of synchrony measures for the early diagnosis of Alzheimer's disease based on EEG," *NeuroImage*, vol. 49, pp. 668–693, 2010.

[51] G. Henderson *et al.*, "Development and assessment of methods for detecting dementia using the human electroencephalogram," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 8, pp. 1557–1568, Aug. 2006.

[52] H. Yu and J. Dauwels. "Modeling functional networks via piecewise-stationary graphical models," in *Signal Processing and Machine Learning for Biomedical Big Data.* Boca Raton, FL, USA: CRC Press, 2018, pp. 193–208.

[53] H. Yu, J. Dauwels, and X. Wang, "Copula Gaussian graphical models with hidden variables," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2012, pp. 2177–2180.

**Hang Yu** (Member, IEEE) received the PhD degree in electrical and electronics engineering from Nanyang Technological University (NTU), Singapore, in 2015. He is currently a senior algorithm expert with Ant Group, China. Before joining Ant Group, he was a senior research fellow with Nanyang Technological University, Singapore. His research interests include graphical models, Bayesian inference, stochastic optimization, extreme value statistics, deep learning, and machine learning.



**Songwei Wu** received the BS degree in statistics from Sun Yat-Sen University (SYSU), Guangzhou, China, in 2011, and the PhD degree in electrical and electronic engineering from Nanyang Technological University(NTU), Singapore, in 2020. He is currently works as a quantitative researcher. His research interest lies in convex optimization, machine learning, and quantitative finance.



**Justin Dauwels** (Senior Member, IEEE) received the PhD degree in electrical engineering from the Swiss Polytechnical Institute of Technology (ETH), Zurich, Switzerland, in December 2005. He is currently an associate professor with TU Delft since January 2021. Prior to this, he was an associate professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore. His research interests include Bayesian statistics, iterative signal processing, and computational neuroscience. He was a postdoctoral fellow with the RIKEN Brain Science Institute (2006–2007) and a research scientist with the Massachusetts Institute of Technology (2008–2010). He has been a JSPS postdoctoral fellow (2007), a BAEF fellow (2008), a Henri-Benedictus fellow of the King Baudouin Foundation (2008), and a JSPS invited fellow (2010, 2011). His research team has won several best paper awards at international conferences. He has filed five U.S. patents related to data analytics.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.