# GRAPH TOPOLOGY LEARNING AND SIGNAL RECOVERY VIA BAYESIAN INFERENCE

*Mahmoud Ramezani-Mayiami*[†], *Mohammad Hajimirsadeghi*[*‡], *Karl Skretting*[††], *Rick S. Blum*[‡], *H. Vincent Poor*[*]
[†]Department of ICT, University of Agder, Norway, mahmoud.ramezani@uia.no
[‡]Department of ECE, Lehigh University, USA
[††]Department of Electrical Engineering and Computer Science, University of Stavanger, Norway
[*]Department of Electrical Engineering, Princeton University, USA

## ABSTRACT

The estimation of a meaningful affinity graph has become a crucial task for representation of data, since the underlying structure is not readily available in many applications. In this paper, a topology inference framework, called Bayesian Topology Learning, is proposed to estimate the underlying graph topology from a given set of noisy measurements of signals. It is assumed that the graph signals are generated from Gaussian Markov Random Field processes. First, using a factor analysis model, the noisy measured data is represented in a latent space and its posterior probability density function is found. Thereafter, by utilizing the minimum mean square error estimator and the Expectation Maximization (EM) procedure, a filter is proposed to recover the signal from noisy measurements and an optimization problem is formulated to estimate the underlying graph topology. The experimental results show that the proposed method has better performance when compared to the current state-of-the-art algorithms with different performance measures.

## 1. INTRODUCTION

Various real-world applications generate rapidly growing volumes of structured data, e.g. social networks activities, patient records of healthcare systems, and financial data. Analyzing these data sets is often easier when the structure of the data is represented in an effective way. The emerging field of Graph Signal Processing (GSP) [1] helps in the analysis of large data sets by the use of graph theory, where each graph node represents a component of the system. Moreover, a sequence of data is generated by each component, residing at each node. In this framework, many applications can be modeled using a graph, which captures the underlying topology among different entities of the network. For example, in a network of thermal sensors which measure temperatures in different locations, the entities are the sensors that can be modeled as the graph nodes. What is measured by the entire sensor network consists of the set of graph signals, which stores the samples of all nodes at a specific time. A desired goal is to estimate the underlying topology using a set of graph signals. Some work has concentrated on directed topology estimation [2–5], while other works focused on undirected graph structures [6–11].

Given a set of measurements, the empirical covariance matrix can be estimated by computing the inner product of each pair of measurements, which helps in inferring the underlying topology. For example, [12] proposed the "covariance selection" to find the graph connectivity from Gaussian measurements. An $\ell_1$ regularized optimization problem was proposed by [13] which generates a sparse inverse covariance matrix. [14] proposed a fast algorithm to implement inverse covariance estimation. To estimate a wider class of affinity graphs, several researches in GSP proposed to learn the graph weight/Laplacian matrix instead of the inverse covariance matrix. The author in [15] has formulated a minimization problem to learn a graph from smooth signals by estimating the graph weight matrix. In that work, the graph has only one connected component and node degrees are positive (they can not be zero) despite the fact that in many applications, the graph is sparse but may have one or more disjoint components. In [16], a general optimization problem has been formulated which can be reduced to the above methods for graph learning and inverse covariance estimation for specific parameters setups. However, all of the above mentioned approaches has only focused on the graph topology learning and did not discuss graph signal representation and noise filtering. Dong et al. [8] proposed an algorithm called GL-SigRep, which estimates the topology and remove the noise from measured signals, simultaneously. This method uses a maximum a posteriori (MAP) estimate to find the underlying topology of the Gaussian Markov Random Field (GMRF) and filters the graph signals iteratively.

**Our Contribution**: Given a set of multi-dimensional noisy observations generated by a set of nodes, we propose a novel algorithm to find the underlying graph topology, called Bayesian Topology Learning (BTL). We apply a minimum

This material is based upon work partially supported by the U. S. Army Research Laboratory and the U. S. Army Research Office under grant number W911NF-17-1-0331 and by the National Science Foundation under grants ECCS-1405579, CNS-1702555, DMS-1736417 and ECCS-1824710.

mean square error (MMSE) estimator to learn the Laplacian matrix, representing the affinity graph of the given data set. In signal denoising step, we propose an analytical approach to estimate the filter coefficient, while [8] does it empirically via a grid search and [16] has not discussed it. Since the proposed algorithm updates the noise variance estimation in each iteration, it can adaptively reduce the noise effect on the graph learning procedure as well, leading to a better performance when compared to the Combinatorial Graph Laplacian (CGL) learning method [16] or GL-SigRep [8]. The simulation results show that BTL achieves better performance in terms of NMSD, NMSE, Recall, Precision and F-measure, especially for large number of measurements.

## 2. GRAPH SIGNAL PROCESSING

Assume $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ is a graph with vertices $v_i \in \mathcal{V}$ and edges $(v_i, v_j) \in \mathcal{E}$ connecting neighboring nodes. Each edge $(v_i, v_j), 1 \leq i, j \leq N$ has a weight of $w_{ij}$ in the corresponding entry of the undirected weight matrix $\mathbf{W} \in \mathbb{R}_{+}^{N \times N}$. In other words, $w_{ij} = 0$ means no connection and $w_{ij} = w_{ji} > 0$ denotes the strength of connection between the nodes $v_i$ and $v_j$. We assume that there is no self loop, i.e., $w_{ii} = 0$. The degree matrix is defined as $\mathbf{D} = \text{diag}(\mathbf{W} \cdot \mathbf{1}_N)$, where $\mathbf{1}_N$ is the all ones $N \times 1$ vector and diag$(\cdot)$ refers to the diagonal elements of the input matrix. The combinatorial graph Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$ and since it is real and symmetric, its eigendecomposition can be written as

$$\mathbf{L} = \boldsymbol{\chi} \boldsymbol{\Lambda} \boldsymbol{\chi}^T, \tag{1}$$

where $\boldsymbol{\Lambda}$ and $\boldsymbol{\chi} \in \mathbb{R}^{N \times N}$ are eigenvalue and eigenvector matrices, respectively, and $(\cdot)^T$ is the matrix transpose operator. Finally, $\mathbf{y}[k]$ represents the $k$'th graph signal as

$$\begin{aligned} &\mathbf{y}[k] : \mathcal{V} \to \mathbb{R}^N, v_i \mapsto y_i[k] \\ &\mathbf{y}[k] = \big(y_1[k], y_2[k], ..., y_N[k]\big)^T \in \mathbb{R}^N, \end{aligned} \tag{2}$$

If a graph signal has smooth variations on the underlying topology, or in other words, strongly connected nodes have similar values, the signal is called smooth with respect to the graph. A measure for total smoothness is represented by the quadratic term $\mathbf{y}^T \mathbf{L} \mathbf{y}$ [1].

## 3. BAYESIAN TOPOLOGY INFERENCE

We are given the data matrix $\mathbf{X} \in \mathbb{R}^{N \times K}$ containing noisy measurements $\mathbf{x}[k] = \mathbf{y}[k] + \mathbf{e}[k]$ in its columns. The measurement noise $\mathbf{e}[k]$ is drawn from a multivariate Gaussian distribution with the probability density function (pdf)

$$p(\mathbf{e}[k]; \sigma_e) \sim \mathcal{N}(\mathbf{0}_N, \sigma_e \mathbf{I}_N) \tag{3}$$

where $\mathbf{0}_N \in \mathbb{R}^N$ and $\mathbf{I}_N$ are all zero vector and the identity matrix of size $N \times N$, respectively. The goal is to find the underlying graph topology which can capture the undirected dependencies among these signals. We consider the factor analysis model in which each graph signal is modeled as $\mathbf{y}[k] = \boldsymbol{\chi} \mathbf{h}[k], \ \forall k$ where the unobserved latent variable $\mathbf{h}[k] \in \mathbb{R}^N$ controls each graph signal via the eigenvector matrix $\boldsymbol{\chi}$ [17]. As mentioned in [8], the motivation of this model is to use a representation matrix that can be linked to the graph Laplacian/topology directly. Thus, each graph signal can contribute to the graph structure. Similar to the traditional factor analysis model, we assume $\mathbf{h}[k]$ follows a degenerate zero mean multivariate Gaussian distribution, given as

$$p\big(\mathbf{h}[k]; \boldsymbol{\Lambda}^\dagger\big) \sim \mathcal{N}(\mathbf{0}_N, \boldsymbol{\Lambda}^\dagger), \tag{4}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo-inverse operator and $\boldsymbol{\Lambda}$ is the precision matrix of the latent variable. Considering all graph signals, the measurements can be rewritten in matrix form by use of the notations $\mathbf{H} = \big[\mathbf{h}[1], \ldots, \mathbf{h}[K]\big]$ and $\mathbf{E} = \big[\mathbf{e}[1], \ldots, \mathbf{e}[K]\big]$ as follows

$$\mathbf{X} = \boldsymbol{\chi} \mathbf{H} + \mathbf{E}, \tag{5}$$

or equivalently by using the Kronecker product $\otimes$, as the following vectorized form

$$\mathbf{x} = \mathbf{B}\mathbf{h} + \mathbf{e}, \tag{6}$$

where $\mathbf{x} = \text{vec}\big(\mathbf{X}\big), \mathbf{B} = \mathbf{I}_K \otimes \boldsymbol{\chi}, \mathbf{h} = \text{vec}\big(\mathbf{H}\big), \mathbf{e} = \text{vec}\big(\mathbf{E}\big)$, and vec(.) stacks columns of its input in a vertical vector. Using (4) and (6), we have the following distributions

$$p\big(\mathbf{h}; \boldsymbol{\Lambda}^\dagger\big) \sim \mathcal{N}(\mathbf{0}_N, \mathbf{C}_0^\dagger), \tag{7}$$

$$p\big(\mathbf{x} \mid \mathbf{h}; \boldsymbol{\chi}, \sigma_e\big) \sim \mathcal{N}\big(\mathbf{B}\mathbf{h}, \sigma_e \mathbf{I}_{NK}\big), \tag{8}$$

$$p\big(\mathbf{x}; \boldsymbol{\Lambda}^\dagger, \boldsymbol{\chi}, \sigma_e\big) \sim \mathcal{N}\big(\mathbf{0}, \mathbf{B}\mathbf{C}_0^\dagger\mathbf{B}^T + \sigma_e \mathbf{I}_{NK}\big), \tag{9}$$

where $\mathbf{C}_0 = \mathbf{I}_K \otimes \boldsymbol{\Lambda}$. Considering $\boldsymbol{\chi}\boldsymbol{\chi}^T = \mathbf{I}_N$ and Kronecker product properties, the covariance matrix of $p(\mathbf{x})$ in (9) can be simplified as

$$\begin{aligned} \mathbf{B}\mathbf{C}_0^\dagger\mathbf{B}^T + \sigma_e \mathbf{I} &= \big(\mathbf{I}_K \otimes \boldsymbol{\chi}\big)\big(\mathbf{I}_K \otimes \boldsymbol{\Lambda}^\dagger\big)\big(\mathbf{I}_K \otimes \boldsymbol{\chi}\big)^T + \sigma_e \mathbf{I}_{NK} \\ &= \Big(\mathbf{I}_K \otimes \big(\mathbf{L}^\dagger + \sigma_e \mathbf{I}_N\big)\Big), \end{aligned} \tag{10}$$

where by using (1), we have $\mathbf{L}^\dagger = \boldsymbol{\chi}\boldsymbol{\Lambda}^\dagger\boldsymbol{\chi}^T$ [8]. By applying Bayes rule, the posterior pdf of $\mathbf{h}$ is obtained as

$$p\big(\mathbf{h} \mid \mathbf{x}; \boldsymbol{\Lambda}^\dagger, \boldsymbol{\chi}, \sigma_e\big) \sim \mathcal{N}\big(\boldsymbol{\mu}_\mathbf{h}, \mathbf{C}_\mathbf{h}\big) \tag{11}$$

with the mean

$$\boldsymbol{\mu}_\mathbf{h} = \mathbb{E}\big[\mathbf{h} \mid \mathbf{x}\big] = \frac{1}{\sigma_e}\mathbf{C}_\mathbf{h}\mathbf{B}^T\mathbf{x}, \tag{12}$$

where $\mathbb{E}(\cdot)$ denotes the expectation and

$$\mathbf{C}_\mathbf{h} = \big(\mathbf{C}_0 + \frac{1}{\sigma_e}\mathbf{B}^T\mathbf{B}\big)^{-1} = \mathbf{I}_K \otimes \big(\boldsymbol{\Lambda} + \frac{1}{\sigma_e}\mathbf{I}_N\big)^{-1} \tag{13}$$

The graph signal can be estimated by using the MMSE estimator as $\hat{\mathbf{y}}_{\text{MMSE}} = \mathbf{B}\hat{\mathbf{h}}_{\text{MMSE}} = \mathbf{B}\mathbb{E}\big[\mathbf{h} \mid \mathbf{x}\big]$ which can be simplified to

$$\hat{\mathbf{y}} = \Big(\mathbf{I}_K \otimes \big(\mathbf{I}_N + \sigma_e \mathbf{L}\big)^{-1}\Big)\mathbf{x}, \tag{14}$$

where $\sigma_e$ and $\mathbf{L}$ will be derived shortly. Similarly, using matrix notation, we have $\hat{\mathbf{Y}} = \big(\mathbf{I}_N + \sigma_e \mathbf{L}\big)^{-1}\mathbf{X}$ which is similar to $\hat{\mathbf{Y}} = \big(\mathbf{I} + \alpha \mathbf{L}\big)^{-1}\mathbf{X}$ in [8]. However, $\alpha$ was not estimated analytically in [8] while we propose an expectation maximization (EM) procedure to update $\sigma_e$ here. The EM formulation proceeds by maximizing

$$\begin{aligned} Q\big(\boldsymbol{\Lambda}^\dagger, \boldsymbol{\chi}, \sigma_e\big) =& \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\log p\big(\mathbf{x}, \mathbf{h}; \boldsymbol{\Lambda}^\dagger, \boldsymbol{\chi}, \sigma_e\big)\big] \\ =& \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\log p\big(\mathbf{x} \mid \mathbf{h}; \boldsymbol{\chi}, \sigma_e\big)\big] \\ &+ \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\log p\big(\mathbf{h}; \boldsymbol{\Lambda}^\dagger\big)\big] \\ =& Q_1\big(\boldsymbol{\chi}, \sigma_e\big) + Q_2\big(\boldsymbol{\Lambda}^\dagger\big), \end{aligned} \tag{15}$$

where $\hat{\boldsymbol{\Lambda}}^\dagger$, $\hat{\boldsymbol{\chi}}$, and $\hat{\sigma}_e$ are the estimations of $\boldsymbol{\Lambda}^\dagger$, $\boldsymbol{\chi}$, and $\sigma_e$ in the previous iteration. To find the optimum parameters, $Q$ must be maximized with respect to each of its three inputs iteratively up to a convergence. First, to maximize with respect to $\sigma_e$, the $Q_1$ function is simplified as follows

$$\begin{aligned} Q_1(\boldsymbol{\chi}, \sigma_e) =& -\frac{NK}{2}\log \sigma_e - \frac{1}{2\sigma_e}\mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\,\|\mathbf{x} - \mathbf{B}\mathbf{h}\|_2^2\,\big] \\ =& -\frac{NK}{2}\log \sigma_e - \frac{1}{2\sigma_e}\times \\ & \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\,\|\mathbf{x} - \mathbf{B}\boldsymbol{\mu}_\mathbf{h} + \mathbf{B}\boldsymbol{\mu}_\mathbf{h} - \mathbf{B}\mathbf{h}\|_2^2\,\big] \\ \overset{(a)}{=}& -\frac{NK}{2}\log \sigma_e - \frac{1}{2\sigma_e}\Big[\,\|\mathbf{x} - \mathbf{B}\boldsymbol{\mu}_\mathbf{h}\|_2^2 \\ & + \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\,\|\mathbf{B}(\mathbf{h} - \boldsymbol{\mu}_\mathbf{h})\|_2^2\,\big]\Big] \\ \overset{(b)}{=}& -\frac{NK}{2}\log \sigma_e - \frac{1}{2\sigma_e}\|\mathbf{x} - \mathbf{B}\boldsymbol{\mu}_\mathbf{h}\|_2^2 \\ & - \frac{K}{2\sigma_e}\text{tr}\big(\hat{\boldsymbol{\Lambda}} + \frac{1}{\hat{\sigma}_e}\mathbf{I}_N\big)^{-1}, \end{aligned} \tag{16}$$

where $\text{tr}(\cdot)$ is the trace of the matrix, $\overset{(a)}{=}$ follows from $\mathbb{E}\big[\mathbf{x} - \mathbf{B}\mathbf{h}\big] = \mathbf{0}_{NK}$ and we use $\mathbf{B}^T\mathbf{B} = \mathbf{I}_{NK}$ in $\overset{(b)}{=}$. The new estimate of $\sigma_e$ is obtained by setting the derivative of (16) with respect to $\sigma_e$ to zero, leading to

$$\begin{aligned} \hat{\sigma}_e^{\text{new}} &= \frac{\|\mathbf{x} - \hat{\mathbf{y}}\|_2^2 + K\text{tr}\Big(\big(\hat{\boldsymbol{\Lambda}} + \frac{1}{\hat{\sigma}_e}\mathbf{I}_N\big)^{-1}\Big)}{NK} \\ &= \frac{1}{NK}\Big\|\mathbf{X} - \hat{\mathbf{Y}}\Big\|_F^2 + \frac{1}{N}\text{tr}\Big(\big(\hat{\mathbf{L}} + \frac{1}{\hat{\sigma}_e}\mathbf{I}_N\big)^{-1}\Big), \end{aligned} \tag{17}$$

where $\|\cdot\|_F^2$ computes the Frobenius norm.

In the next step, we can maximize $Q_2(\cdot)$ with respect to $\boldsymbol{\chi}$ and find $\hat{\boldsymbol{\chi}}$. Here, since we are not interested in the eigenvector matrix $\boldsymbol{\chi}$ and the ultimate goal is to estimate the Laplacian matrix, i.e. $\mathbf{L} = \boldsymbol{\chi}\boldsymbol{\Lambda}\boldsymbol{\chi}^T$, we skip this step. Instead, by applying the eigenvector matrix property $\boldsymbol{\chi}\boldsymbol{\chi}^T = \boldsymbol{\chi}^T\boldsymbol{\chi} = \mathbf{I}_N$ and maximizing $Q_2$ (equivalently $Q$) with respect to $\boldsymbol{\Lambda}$, the Laplacian matrix is estimated. To find the expectation of the second term in (15), i.e. $Q_2(\cdot)$, we have ($\propto$ means proportional to)

$$\log p\big(\mathbf{h}; \boldsymbol{\Lambda}\big) \propto \log|\mathbf{C}_0| - \mathbf{h}^T\mathbf{C}_0\mathbf{h}, \tag{18}$$

where due to the singularity of $\boldsymbol{\Lambda}$ and $\mathbf{C}_0$, the pseudo-determinant $|\cdot|$ is used, i.e. the product of all non-zero eigenvalues of a square matrix. Then, we have

$$\begin{aligned} Q_2\big(\boldsymbol{\Lambda}^\dagger\big) &\propto \log|\mathbf{C}_0| - \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\mathbf{h}^T\mathbf{C}_0\mathbf{h}\big] \\ &= K\cdot\log|\boldsymbol{\Lambda}| - \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\boldsymbol{\chi}},\hat{\sigma}_e,\hat{\boldsymbol{\Lambda}}^\dagger}\big[\mathbf{y}^T\big(\mathbf{I}_K \otimes \mathbf{L}\big)\mathbf{y}\big] \\ &= K\cdot\log|\mathbf{L}| - \text{tr}\big(\hat{\mathbf{Y}}^T\mathbf{L}\hat{\mathbf{Y}}\big), \end{aligned} \tag{19}$$

where $\mathbf{B}^T\mathbf{B} = \mathbf{I}_{NK}$ and $|\mathbf{L}| = |\boldsymbol{\Lambda}|$ are used. Thus, to maximize $Q_2$ in (15), the EM algorithm solves

$$\begin{aligned} \underset{\mathbf{L}}{\text{argmax}}\ & K\cdot\log|\mathbf{L}| - \text{tr}\big(\hat{\mathbf{Y}}^T\mathbf{L}\hat{\mathbf{Y}}\big) \\ \text{s.t. }\ & \mathbf{L}_{ij} = \mathbf{L}_{ji},\ \ \mathbf{L}_{ij} \le 0 \text{ if } i \ne j,\ \ \mathbf{L}\cdot\mathbf{1}_N = \mathbf{0}_N, \\ & \text{Tr}(\mathbf{L}) = c, \end{aligned} \tag{20}$$

where the first three constraints ensure a valid Laplacian and the last one avoids the trivial solution by controlling the diagonal entries, for $c > 0$. The constant $c$ is usually determined by the graph sparsity in a specific application since $\text{Tr}(\mathbf{L}) = \frac{1}{2}\|\mathbf{L}\|_1$, i.e. $\|\mathbf{L}\|_1 = 2c$. The second term of the objective function promotes the signal smoothness over the estimated topology [1]. The optimization problem in (20) is convex, but not easy to implement due to $|\cdot|$ term. Instead, the following equivalent minimization problem can be solved,

$$\begin{aligned} \underset{\mathbf{L}}{\text{argmin}}\ & -K\cdot\log\det\big(\mathbf{L} + \mathbf{J}\big) + \text{tr}\big(\hat{\mathbf{Y}}^T\mathbf{L}\hat{\mathbf{Y}}\big) \\ \text{s.t. }\ & \mathbf{L}_{ij} = \mathbf{L}_{ji},\ \ \mathbf{L}_{ij} \le 0 \text{ if } i \ne j,\ \ \mathbf{L}\cdot\mathbf{1}_N = \mathbf{0}_N, \\ & \text{Tr}(\mathbf{L}) = c, \end{aligned} \tag{21}$$

where $\det(\cdot)$ denotes the determinant and $\mathbf{J} = \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T$ (for a combinatorial graph Laplacian matrix, $\log\det\big(\mathbf{L} + \mathbf{J}\big) = \log|\mathbf{L}|$ [16]). Moreover, the underlying topology is sparse in many applications. Thus, we propose to add a term in the objective and a constraint as follows to control edge sparsity

$$\begin{aligned} \underset{\mathbf{L}}{\text{argmin}}\ & -K\cdot\log\det\big(\mathbf{L} + \mathbf{J}\big) + \text{tr}\big(\hat{\mathbf{Y}}^T\mathbf{L}\hat{\mathbf{Y}}\big) + \beta\|\mathbf{L}\|_F^2 \\ \text{s.t. }\ & \mathbf{L} = \mathbf{L}^T,\ \mathbf{L}_{ij} \le 0 \text{ if } i \ne j,\ \mathbf{L}\cdot\mathbf{1}_N = \mathbf{0}_N,\ \text{Tr}(\mathbf{L}) = c, \end{aligned} \tag{22}$$

where $\beta$ is the regularization parameter, and $\|\mathbf{L}\|_F^2$ controls the distribution of off-diagonal elements, i.e. the edge weights of the estimated graph. A smaller $\beta$ allows $\|\mathbf{L}\|_F^2$ to have higher values and then a more sparse Laplacian matrix is achieved since $\|\mathbf{L}\|_1$ is fixed by the constraint $\text{Tr}(\mathbf{L}) = c$. Since (22) is a convex optimization problem, any existing

**Algorithm 1:** Bayesian Topology Learning (BTL).

**Input:** Given measurements $\mathbf{X}$
**Initialize:** $\hat{\mathbf{Y}} = \mathbf{X}$, $\hat{\sigma}_e = \sigma_0$, $c$, and $\beta$
**while** *Not Converged* **do**
  **while** *EM Not Converged* **do**
    Laplacian matrix learning by solving (22),
    Noise variance estimation via (17),
  **end**
  Signal denoising: $\hat{\mathbf{Y}} = \left(\mathbf{I}_N + \hat{\sigma}_e \hat{\mathbf{L}}\right)^{-1} \mathbf{X}$.
**end**
**Output**: $\hat{\mathbf{L}}$, $\hat{\mathbf{Y}}$ and $\hat{\sigma}_e$.

convex solver can be used. Here, we used a toolbox for optimization in MATLAB, called YALMIP [18]. The Bayesian Topology Learning method is shown in Algorithm 1.

## 4. SIMULATIONS

The BTL algorithm is tested using synthetic data drawn from a GMRF processes with the following scenario (similar to the scenario proposed and used in [8] and [16]): In each trial, first, the coordinates of $N = 30$ vertices are generated uniformly at random in the unit square and the edge weights are computed with a Gaussian radial basis function, i.e. $\exp\big(-d(i,j)^2/2\sigma^2\big)$ where $d(i,j)$ is the distance between vertices $i$ and $j$. We set $\sigma = 0.5$ and remove the edges whose weights are smaller than 0.75. The graph Laplacian is computed and normalized by its trace. Then, each data vector is sampled from a N-variate Gaussian distribution $\mathbf{y}[k] \sim \mathcal{N}(\mathbf{0}_N, \mathbf{L}^\dagger)$ and contaminated by independent and identically distributed Gaussian noise employing a signal-to-noise ratio of 5dB to have $\mathbf{x}[k] = \mathbf{y}[k] + \mathbf{e}[k]$. The measurements $\mathbf{x}[k], k = 1, \ldots, 1800$ are inserted in the columns of the matrix $\mathbf{X}$. Given $K$ graph signals, i.e. $\mathbf{X}(:, 1:K)$ where $K \in \{300, 600, 900, 1200, 1500, 1800\}$, the graph topology is estimated. The methods are compared with the following performance metrics:

- Normalized Mean Squared Deviation of graph topology estimation: NMSD $= \frac{1}{N^2} \cdot \frac{\|\mathbf{L}-\hat{\mathbf{L}}\|_2^2}{\|\mathbf{L}\|_2^2}$, where $\hat{\mathbf{L}}$ denotes the estimated Laplacian matrix,
- Normalized Mean Squared Error of signal reconstruction: NMSE $= \frac{1}{N \cdot K} \cdot \frac{\|\mathbf{X}-\hat{\mathbf{X}}\|_2^2}{\|\mathbf{X}\|_2^2}$,
- Precision: the number of truly recovered edges to the total number of reconstructed edges in the estimated graph,
- Recall: the number of truly recovered edges to the number of edges in the ground-truth graph,
- F-measure $= \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$.

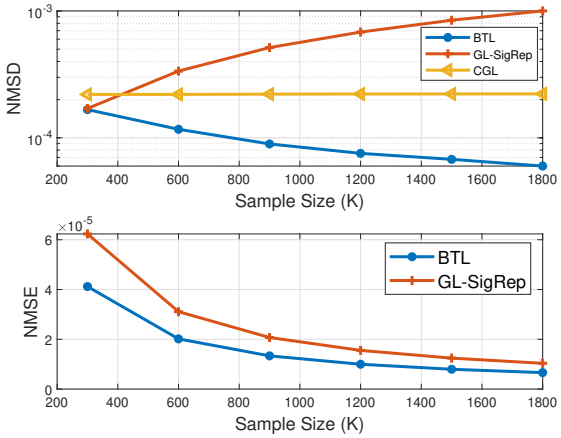


**Fig. 1**: Top: The error of graph topology estimation; Bottom: The error of signal denoising ($N = 30$ and SNR $= 5dB$).

**Table 1**: Averages of the performance measures.

| | Recall | Precision | F-measure |
|---|---|---|---|
| BTL | 0.86 | 0.76 | 0.81 |
| CGL [16] | 0.90 | 0.30 | 0.46 |
| GL-SigRep [8] | 0.44 | 0.97 | 0.59 |

To run algorithm 1, we set $c = N$ [8] and $\sigma_0 = 1$. The two competitive algorithms are Graph Learning for Smooth Signal Representation (GL-SigRep) [8] and Combinatorial Graph Laplacian (CGL) learning [16][1]. In the graph learning step, if the absolute value of $L_{ij}$ is less than 0.0001, it will be pruned. We run simulations for 100 trials and averaged the results.

Fig. 1 shows the higher performance of BTL in terms of NMSD and NMSE for different sample sizes. As we discussed in section 3, both BTL and GL-SigRep filter the measurements to estimate the graph signals by a first order linear shift invariant graph filter, but CGL did not discuss a method for noise reduction. Thus, in the NMSE comparison figure, CGL is not shown. Since BTL uses new measurements to update the $\sigma_e$ learning rule, it has lower NMSE when compared to GL-SigRep, in which the corresponding parameter is found by an experimental grid search. As the number of measurements increases, the BTL performance in both graph learning and signal denoising steps are improved. In fact, by having more observations, we will have more accurate estimation of $\sigma_e$ which results in lower NMSD and NMSE. Table 1 compares the three performance measures by averaging the results over different trials and all $K$'s (because of the lack of space, we could not show the result for each $K$ separately). The higher F-measure value shows BTL is successful in learning a graph close to the ground-truth one.

[1]The code is publicly available in `https://github.com/STAC-USC/Graph_Learning`

## 5. REFERENCES


[1] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[2] J. Mei and J. M. Moura, "Signal processing on graphs: Causal modeling of unstructured data," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 2077–2092, 2017.

[3] A. Bolstad, B. D. V. Veen, and R. Nowak, "Causal network inference via group sparse regularization," *IEEE Transactions on Signal Processing*, vol. 59, no. 6, pp. 2628–2641, June 2011.

[4] Y. Shen, B. Baingana, and G. B. Giannakis, "Kernel-based structural equation models for topology identification of directed networks," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2503–2516, May 2017.

[5] M. Ramezani-Mayiami, "Joint graph learning and signal recovery via Kalman filter for multivariate autoregressive processes," in *Proc. IEEE 26th European Signal Processing Conference (EUSIPCO)*, Sep. 2018, pp. 907–911.

[6] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graph," in *Proc. 32nd Annual Meeting of the Cognitive Science Society*, 2010, pp. 6160–6173.

[7] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo, "Graph topology inference based on transform learning," in *Proc. IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec 2016, pp. 356–360.

[8] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, Dec 2016.

[9] E. Isufi, P. Banelli, P. Di Lorenzo, and G. Leus, "Observing and tracking bandlimited graph processes," *arXiv preprint arXiv:1712.00404*, 2017.

[10] R. Varma, S. Chen, and J. Kovačević, "Graph topology recovery for regular and irregular graphs," in *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, Dec 2017, pp. 1–5.

[11] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 3, pp. 467–483, Sept 2017.

[12] A. P. Dempster, "Covariance selection," *Biometrics*, pp. 157–175, 1972.

[13] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *Journal of Machine Learning Research*, vol. 9, no. Mar, pp. 485–516, 2008.

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

[15] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. 19th International Conference of Artificial Intelligence and Statistics, AISTATS, Cadiz*, 2016, pp. 920–929.

[16] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, 2017.

[17] A. T. Basilevsky, *Statistical Factor Analysis and Related Methods: Theory and Applications*. John Wiley & Sons, 2009, vol. 418.

[18] J. Löfberg, "YALMIP : A toolbox for modeling and optimization in MATLAB," in *Proc. IEEE International Conference on Robotics and Automation*, Sep. 2004, pp. 284–289.