

## Bayesian Topology Learning and noise removal from network data

Mahmoud Ramezani Mayiami<sup>1</sup> · Mohammad Hajimirsadeghi<sup>2</sup> · Karl Skretting<sup>3</sup> · Xiaowen Dong<sup>4</sup> · Rick S. Blum<sup>2</sup> · H. Vincent Poor<sup>5</sup>

Received: 4 February 2021 / Accepted: 10 March 2021

© The Author(s) 2021 [OPEN](#)

### Abstract

Learning the topology of a graph from available data is of great interest in many emerging applications. Some examples are social networks, internet of things networks (intelligent IoT and industrial IoT), biological connection networks, sensor networks and traffic network patterns. In this paper, a graph topology inference approach is proposed to learn the underlying graph structure from a given set of noisy multi-variate observations, which are modeled as graph signals generated from a Gaussian Markov Random Field (GMRF) process. A factor analysis model is applied to represent the graph signals in a latent space where the basis is related to the underlying graph structure. An optimal graph filter is also developed to recover the graph signals from noisy observations. In the final step, an optimization problem is proposed to learn the underlying graph topology from the recovered signals. Moreover, a fast algorithm employing the proximal point method has been proposed to solve the problem efficiently. Experimental results employing both synthetic and real data show the effectiveness of the proposed method in recovering the signals and inferring the underlying graph.

**Keywords** Graph signal processing · Topology learning · Signal representation · Bayesian inference · Internet of things

## 1 Introduction

Many applications including internet of things (IoT) networks, sensor networks, social networks, gene networks, customer consumption data in utility companies, financial data, regional temperature data, and brain computer interface measurements result in rapidly growing data volumes. IoT networks are a rapidly emerging technology where smart sensors are connected to easily enable data collection anywhere and anytime [1]. Figure 1 illustrates how an IoT network may involve collecting large volume of multi-variate time series data which are connected in some abstract senses.

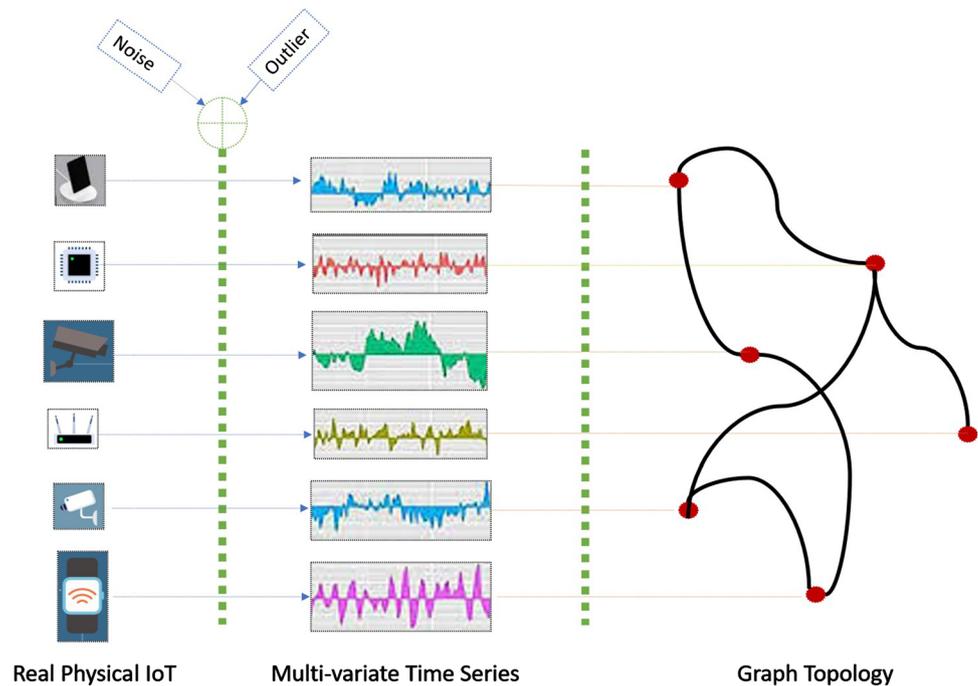
Storing and analyzing these huge data volumes is challenging due to the need for high computation and power resources. The emerging field of graph signal processing (GSP) [2, 3] simplifies the analysis of large data volumes by the use of graph theory, where each graph node represents a component of the system. Several applications can be analyzed by graphs, which can capture the underlying topology among different entities of the network. For example, a network of thermal sensors which measures temperatures in different regions, can be modeled by a graph.

In some cases, the graph topology is not known a priori. Thus, one of the desired goals of the GSP framework is estimating the underlying topology given a set of measured data. Some researches concentrated on directed topology estimation [4–14], where specific process models have been used which are suitable for limited applications. However,

✉ Mahmoud Ramezani Mayiami, mahmoud.ramezani@uia.no; Mohammad Hajimirsadeghi, mohammad@winlab.rutgers.edu; Karl Skretting, karl.skretting@uis.no; Xiaowen Dong, xdong@robots.ox.ac.uk; Rick S. Blum, rb0f@lehigh.edu; H. Vincent Poor, poor@princeton.edu | <sup>1</sup>Department of ICT, University of Agder, Grimstad, Norway. <sup>2</sup>Department of ECE, Lehigh University, Bethlehem, USA. <sup>3</sup>Department of EECS, University of Stavanger, Stavanger, Norway. <sup>4</sup>Department of Engineering Science, University of Oxford, Oxford, UK. <sup>5</sup>Department of Electrical Engineering, Princeton University, Princeton, USA.



**Fig. 1** Illustration of learning a network graph structure from IoT data



the Gaussian model is a ubiquitous signal model which can be utilized for a wide range of applications. By using data sets generated by Gaussian distributions, the solution of a topology inference problem usually leads to an undirected graph structure. The “covariance selection” proposed by Dempster [15], was one of the pioneering works to capture the connectivity underlying the Gaussian measurements. Later on, Banerjee et. al. [16] proposed an  $\ell_1$  regularized optimization problem to find a sparse precision matrix which is the inverse of the covariance matrix. The precision matrix carries information about the partial correlations between the random variables. Friedman and his colleagues have proposed a fast algorithm, called graphical Lasso, to implement inverse covariance estimation [17]. There are other works investigating the inverse covariance matrix estimation with different approaches and different rates of convergence [18, 19]. However, none of these approaches can leverage the signal model to infer the underlying graph topology. In other words, they focus on pairwise relationships of entities to find their structure instead of applying the signal model information. Besides, these approaches are very sensitive to noise since they are based purely on the measurement values.

A state-of-the-art machine learning approach for estimating a graph topology from measurements generated by a Gaussian Markov Random Field processes has been proposed in [20]. The main limitations of this algorithm are twofold. First, no low cost implementation is proposed that scales efficiently with the number of nodes. Second, the effect of noisy measurements on the performance evaluation has not been considered. Kalofolias has proposed an efficient and scalable algorithm to learn a graph from noise-free observations where it is assumed that the node degrees are positive (they can not be zero) [21]. A generalization of the methods in [20] and [21] for different Laplacian and structural constraints has been proposed in [22]. Using graph signal frequency domain analysis, the underlying network topology is inferred from spectral templates in [23]. The authors assumed that there exists a diffusion process in the graph shift operator (GSO<sup>1</sup>) that can generate the given observations and the GSO is jointly diagonalizable with the observations covariance matrix. This method was first applied to capture the underlying topology when the graph signals are generated from a GSO filter output fed by a white signal input. Then, this method was extended to the colored input, generating non-stationary diffusion processes<sup>2</sup>. Some other research works with similar ideas are presented for different stationary and non-stationary processes in [24–28]. Dictionary learning [29, 30] and transform learning [31] have also been used for inferring the graph topology. In [29–31], a specific relation between the Laplacian matrix and the dictionary atoms has been sought and hence these algorithms are applicable when we have some knowledge about signal representation in

<sup>1</sup> The GSO is a matrix which captures the graph’s local topology and the graph Fourier transform is defined using its eigenvectors.

<sup>2</sup> A process is wide sense stationary if its first moment is constant over the graph vertex set and its covariance matrix is invariant with respect to the localization operator [24].

the transform domain. There are also many other papers investigating the topology identification and graph learning methods from different perspectives [8, 32–41]. However, none of the above approaches has discussed graph signal recovery and topology learning when the measurements are corrupted by noise.

The inexpensive sensors found in a wide variety of applications distort the observed data samples. The resulting measurement errors in the data analysis are conventionally modeled as noise [42] and it is generally desirable to remove this noise in many applications [43, 44], which generally produced a more accurate learned topology. In a recent research [45], Liu and her colleagues proposed a noisy data cleansing algorithm to remove unwanted distortion from Industrial IoT (IIoT) sensor data in manufacturing. They showed the efficiency of their proposed method on the multi-variate time-series data collected from a real IIoT based four stage compressor manufacturing plant. In [46], the effect of distortion on the trading data has been discussed, where the stock market's impact on this data is characterized by a graph [47]. There are also other studies in the GSP framework considering the noise effect on the observations, e.g. [48], but they assumed that they have knowledge about the underlying graph and did not tackle graph topology inference and noise removal at the same time. While it is a naive strategy to find the graph topology first and then remove the noise from the graph signal, this approach has a low performance due to the accumulation of errors in topology learning and signal estimation.

To solve this type of problem in a topology learning framework, an optimization problem is usually formulated with at least two objective terms. The first term is always a data fidelity term controlling the energy of the signal recovery error. The second term takes care of the graph signal smoothness over the learned topology. A smooth signal has smooth variations on the underlying topology, or in other words, connected nodes in the estimated structure have similar signal values. To overcome over-fitting, the data fidelity term is regularized by a smoothing term and an exhaustive search is applied to find the regularization parameter. In many applications, the relationship between the noise variance and the regularization parameters may be held directly, e.g. image restoration [49]. Thus, estimating the noise variance not only provides a cleaner version of the signal for further analysis, but also helps to overcome the overfitting problem. All things considered, the estimation of noise variance is the main step to design the filter for smoothing the signal and denoising the given measurements.

It is usually assumed that the measurement noise has a zero mean Gaussian distribution model which is independent of the desired signal. Thus, finding the noise variance is of interest to design a filter for signal denoising. The noise variance estimation is not only useful from a signal recovery perspective, it is also important from a graph learning point of view. Since finding the underlying structure from a noise-free data set is more accurate than from noisy graph signals, signal denoising can contribute to estimating a more accurate graph topology. In this respect, Chepuri et al. [50] proposed an approach to find the connectivity graph and remove noise from measurements. They proposed a non-convex, cardinality constrained, boolean optimization problem along with a convex relaxation. They have assumed that the number of edges is known in advance and the proposed method scales with the desired number of edges. Another approach has been proposed in [51], which estimates the topology and removes the noise from the measured signals, simultaneously. They adopted a factor analysis model for the multi-variate signals and imposed a Gaussian prior on the latent variables that control the graph signals. By applying a Bayesian framework, the maximum a posteriori (MAP) probability estimate of the latent variable is investigated. Considering signal smoothness over the underlying graph, this procedure leads to an optimization problem over the graph topology and graph signals, simultaneously. A limitation of this approach is that an exhaustive search is applied to find the regularization parameter or noise variance which is not very suitable, especially for the situation in which the measurement noise varies over time.

*Our contribution* Given a set of noisy multi-variate signal measurements, we propose a new algorithm to perform the underlying topology learning and graph signal noise removal. This graph topology characterizes the affinity relationship among the multi-variate signals. We have proposed a minimum mean square error estimation (MMSE) approach for signal recovery and topology learning and unlike [51], we estimate the regularization parameters analytically rather than by using a grid search. We show that the regularization parameters are linked to the noise variance which is usually unknown, and hence, finding the noise variance helps adjusting the parameters precisely. Besides, it is the main role in designing a filter to denoise the graph measurements. In other words, to compare the current paper with other similar ones, especially [51], two specific points must be considered; (1) understanding the relation between amount of noise and the regularization parameters in the denoising procedure, and (2) a rigorous strategy to estimate the noise variance instead of assuming that it is known in advance. The MMSE procedure is utilized to estimate the optimal Laplacian matrix, whose eigenvalue matrix is the precision matrix of the Gaussian Markov Random Field process. This work is an extension of our previous paper [52], in which we studied the problem of joint graph signal recovery and topology learning using an off-the-shelf optimization toolbox to implement the algorithm. However, in this version, we propose a fast algorithm and prove its convergence analytically. Moreover, we provide simulation results on real world data sets in this paper. To

**Table 1** Table of notation

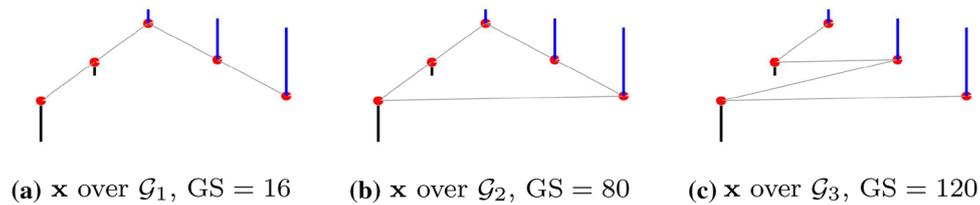
$\mathbf{W}   \mathbf{L}$	Graph weight matrix graph Laplacian matrix
$\mathcal{V}   \mathcal{E}$	Vertex set edge set
$N   K$	Number of vertices number of data samples
$\mathcal{G}$	Weighted graph
$\mathbf{0}_N$	Column vector of zeros of size N
$\mathbf{1}_N$	Column vector ones of size N
$\mathbf{D}$	Graph degree matrix
$\mathbf{A}   \mathcal{X}$	Eigenvalue matrix eigenvector matrix (of $\mathbf{L}$ )
$\boldsymbol{\Theta}^{-1}   \boldsymbol{\Theta}^\dagger$	Inverse of $\boldsymbol{\Theta}$ pseudo-inverse of $\boldsymbol{\Theta}$
$\boldsymbol{\Theta}^T$	Transpose of $\boldsymbol{\Theta}$
$\det(\boldsymbol{\Theta})    \boldsymbol{\Theta} $	Determinant of $\boldsymbol{\Theta}$ pseudo-determinant of $\boldsymbol{\Theta}$
$(\boldsymbol{\Theta})_{ij}$	Entry of $\boldsymbol{\Theta}$ at $i$ th row and $j$ th column
$\mathcal{S}^M$	The set of symmetric matrices
$\mathcal{S}_+^M$	The set of symmetric positive semi-definite matrices
$\text{Tr} \log \det(\boldsymbol{\Theta})$	Trace operator natural logarithm of $\det(\boldsymbol{\Theta})$
$\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$	Zero mean multi-variate Gaussian with covariance $\boldsymbol{\Sigma}$
$\ \boldsymbol{\Theta}\ _F^2, \ \boldsymbol{\theta}\ _2^2$	Sum of squared values of all elements
$\text{diag}(\boldsymbol{\theta})$	Diagonal matrix formed by elements of $\boldsymbol{\theta}$
$\otimes   \mathbb{E}$	Kronecker product expectation operator
$p(\cdot   \cdot)$	Conditional probability density function
$\hat{\theta}   \text{abs}(\theta)$	Estimate of $\theta$   absolute value of $\theta$
$\text{sign}(\boldsymbol{\Theta})$	Sign of each element of $\boldsymbol{\Theta}$
$\nabla   \nabla^2$	Gradient Hessian
$\text{prox}_f$	Proximal operator of function $f$
$\mathcal{L}$	Lagrangian function

compare the results with the state of the art methods, we applied some measures which evaluate the performances from two distinct perspectives which are signal recovery error and graph topology estimation accuracy. Therefore, we can compare our proposed method against other methods that employ signal denoising strategies, like the one proposed in [22]. A simple graph filter is applied based on the estimated noise variance which can be used in many multi-variate measurements applications, including industrial internet of things IloT applications. In the preprocessing stage in some IloT applications, it is necessary to denoise the measurements, remove outlier data, detect anomalies, and estimate the missing values of some data. By the proposed method, the knowledge of the underlying data structure can be applied to implement all of these tasks. As an application of our idea in the area of IloT networks, we apply our proposed method to estimated the missing values of sensor readings for a power consumption dataset.

The rest of the paper is organized as follows; in Sect. 2, some preliminaries about graph theory and signal processing on graphs are presented. Section 3 formulates the graph topology learning problem via a Bayesian framework and proposes a general algorithm for implementation, called Bayesian Topology Learning (BTL). In Sect. 4, a method is proposed to implement BTL efficiently via solving a proximal point algorithm, called BTL-PPA. The proof of convergence is also presented in Sect. 5. Section 6 presents experimental results obtained from the proposed method for temperature sensor networks and IloT applications by using both real and synthetically simulated data. Finally, the Sect. 7 concludes the paper. Throughout the paper, lowercase regular letters, lowercase bold letters, and uppercase bold letters denote scalars, vectors, and matrices, respectively. The rest of the notation is presented in Table 1.

## 2 Background on graph signal processing

Let  $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$  to be a graph with the vertices  $v_i \in \mathcal{V}$ , the edge set  $\mathcal{E}$ , and the undirected weight matrix  $\mathbf{W}$ . Each edge  $(v_i, v_j), 1 \leq i, j \leq N$  has a weight of  $w_{ij}$  in the corresponding entry of  $\mathbf{W} \in \mathbb{R}_+^{N \times N}$ . Thus,  $w_{ij} = 0$  implies no connection and  $w_{ij} = w_{ji} > 0$  quantifies the strength of connection between the node  $v_i$  and the node  $v_j$ . For simplicity of presentation, we assume that the diagonal entries of the weight matrix are zero, i.e. there is no self loop. The adjacency matrix



**Fig. 2** A unique graph signal resided on three different graph topology. The blue and black bars show positive and negative signal values, respectively, and red circles denotes graph vertices. The signal is smooth with respect to the  $\mathcal{G}_1$ , less smooth with respect to  $\mathcal{G}_2$  and likely to be non-smooth with respect to  $\mathcal{G}_3$ . The edge weights are all ones. These figures are generated by GSPBOX [53]

$\mathbf{A}$  stores only the existence of the edges, regardless of their weights. In other words, if there is a connection or edge between the vertices  $v_i$  and  $v_j$ , the adjacency matrix entry  $a_{ij} = 1$  and zero otherwise. The degree matrix is defined as  $\mathbf{D} = \text{diag}(d_i)$ ,  $1 \leq i \leq N$ , where  $\text{diag}(d_i)$  denotes a square diagonal matrix with the degrees  $d_1, d_2, \dots, d_N$  on its main diagonal. The degree  $d_i$  is the sum of the edge weights connecting node  $i$  to its neighbors. Then,  $\mathbf{D} = \text{diag}(\mathbf{W} \cdot \mathbf{1}_N)$ , where  $\mathbf{1}_N$  is the all ones  $N \times 1$  vector. The combinatorial and normalized graph Laplacian matrices are also defined as  $\mathbf{L} = \mathbf{D} - \mathbf{W}$  and  $\mathbf{L}_{\text{norm}} = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}}$ , respectively, where  $\mathbf{I}_N$  is the identity matrix of size  $N \times N$ . Since the Laplacian matrix is real and symmetric, its eigendecomposition is written as follows

$$\mathbf{L} = \chi \mathbf{A} \chi^T, \quad (1)$$

where  $\mathbf{A}$  and  $\chi \in \mathbb{R}^{N \times N}$  are eigenvalue and eigenvector matrices, respectively, and  $(\cdot)^T$  denotes the matrix transpose operator. For the normalized Graph Laplacian, all eigenvalues are between 0 and 2. Since the Laplacian matrix can uniquely characterize the graph structure, the graph topology learning problem can be formulated as a problem of graph Laplacian matrix learning. The  $k$ 'th graph signal is represented as below

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

An important concept in the GSP framework is signal smoothness with respect to the intrinsic structure of the graph. The local smoothness around vertex  $i$  at time  $k$  is given as [2]

$$\|\nabla_i \mathbf{y}[k]\|_2 := \left[ \sum_{j \in \mathcal{N}_i} W_{ij} [y_j[k] - y_i[k]]^2 \right]^{\frac{1}{2}}, \quad (3)$$

where  $\mathcal{N}_i$  is the neighborhood of node  $i$ . Then, to quantify the global smoothness (GS), we have [2]

$$\begin{aligned} GS_2(\mathbf{y}[k]) &= \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{N}_i} W_{ij} [y_j[k] - y_i[k]]^2 \\ &= \sum_{(i,j) \in \mathcal{E}} W_{ij} [y_j[k] - y_i[k]]^2 = \mathbf{y}^T[k] \mathbf{L} \mathbf{y}[k]. \end{aligned} \quad (4)$$

Figure 2 compares the smoothness of a signal living in different topologies.

### 3 Bayesian Topology Learning

Bayes' rule relates the probability of an event to the prior knowledge that we have around it and updates the belief. Here, in the same way, it is assumed that we have some knowledge about the latent variable which factorizes the graph signals and also relates to the underlying graph structure. By Bayesian inference, we investigate the posterior probability of this latent variable and use its probability distribution function to find the graph Laplacian matrix and denoise the observed graph signals.

Assume that we are given a data matrix  $\mathbf{X}$  of size  $N \times K$ , containing noisy graph signal measurements in its columns. The rows of the matrix  $\mathbf{X}$  correspond to the vertices of the underlying graph and each column/measurement is as follows

$$\mathbf{x}[k] = \mathbf{y}[k] + \mathbf{e}[k], k = 1, \dots, K. \tag{5}$$

Let us adopt a multi-variate Gaussian distribution for the measurement noise  $\mathbf{e}[k]$ , given by the following probability density function (pdf)

$$p(\mathbf{e}[k]; \sigma) \sim \mathcal{N}(\mathbf{0}, \sigma \mathbf{I}_N) \tag{6}$$

where  $\mathbf{0} \in \mathbb{R}^N$  is all zero vector.<sup>3</sup>

To find the underlying graph topology, first, we use the factor analysis model to leverage a representation matrix that can be linked to the graph Laplacian/topology directly. In this model, each graph signal is represented by  $\mathbf{y}[k] = \chi \mathbf{h}[k] + \mathbf{u}[k]$ ,  $\forall k$ , where the unobserved latent variable  $\mathbf{h}[k] \in \mathbb{R}^N$  controls each graph signal via the eigenvector matrix  $\chi$  [54] and  $\mathbf{u}[k] \in \mathbb{R}^N$  is the mean of graph signal  $\mathbf{y}[k]$ . For simplicity and without loss of generality, hereafter, it is assumed that  $\forall k, \mathbf{u}[k] = \mathbf{0}$ . As discussed in [51], the motivation of using the factor analysis model is that each graph signal can contribute to the underlying graph structure estimation. Similar to [51], it is assumed that  $\mathbf{h}[k]$  follows a degenerate zero mean multi-variate Gaussian distribution as follows

$$p(\mathbf{h}[k]; \mathbf{A}^\dagger) \sim \mathcal{N}(\mathbf{0}, \mathbf{A}^\dagger), \tag{7}$$

where  $(\cdot)^\dagger$  denotes the Moore–Penrose pseudo-inverse and  $\mathbf{A}$  is the precision matrix. Considering all graph signals, we have

$$\mathbf{X} = \chi \mathbf{H} + \mathbf{E}, \tag{8}$$

where  $\mathbf{H} = [\mathbf{h}[1], \dots, \mathbf{h}[K]]$  and  $\mathbf{E} = [\mathbf{e}[1], \dots, \mathbf{e}[K]]$ . Equivalently, by using the Kronecker product  $\otimes$  property, all given data can be vectorized as follows

$$\mathbf{x} = \mathbf{B}\mathbf{h} + \mathbf{e}, \tag{9}$$

where  $\mathbf{x} = \text{vec}(\mathbf{X})$ ,  $\mathbf{B} = \mathbf{I} \otimes \chi$ ,  $\mathbf{h} = \text{vec}(\mathbf{H})$ ,  $\mathbf{e} = \text{vec}(\mathbf{E})$ , and  $\text{vec}(\cdot)$  stacks all columns of the matrix in a column vector. Thus, we have the following pdf's

$$p(\mathbf{h}; \mathbf{A}^\dagger) \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_0^\dagger), \tag{10}$$

$$p(\mathbf{x} \mid \mathbf{h}; \chi, \sigma) \sim \mathcal{N}(\mathbf{B}\mathbf{h}, \sigma \mathbf{I}), \tag{11}$$

$$p(\mathbf{x}; \mathbf{A}^\dagger, \chi, \sigma) \sim \mathcal{N}(\mathbf{0}, \mathbf{B}\mathbf{C}_0^\dagger \mathbf{B}^T + \sigma \mathbf{I}), \tag{12}$$

where  $\mathbf{C}_0 = \mathbf{I} \otimes \mathbf{A}$ . By applying the eigenvector matrix identity  $\chi \chi^T = \mathbf{I}$  and Kronecker product properties, we have

$$\mathbf{B}\mathbf{C}_0^\dagger \mathbf{B}^T = (\mathbf{I} \otimes \chi)(\mathbf{I} \otimes \mathbf{A}^\dagger)(\mathbf{I} \otimes \chi)^T = \mathbf{I} \otimes \chi \mathbf{A}^\dagger \chi^T, \tag{13}$$

and thus, the covariance matrix of  $\mathbf{x}$  in (12) is simplified as follows

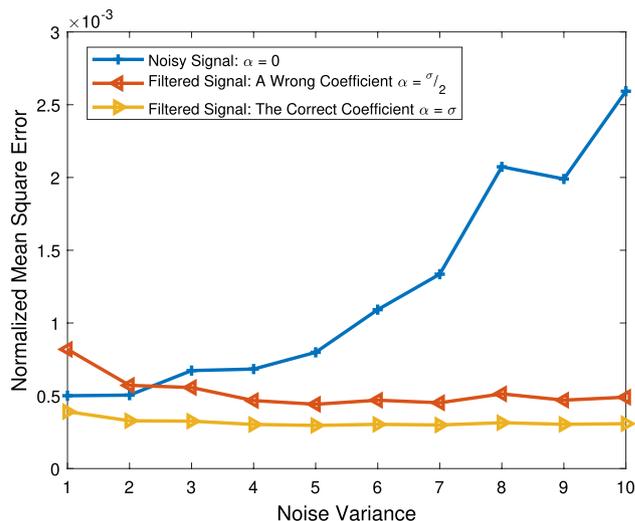
$$\mathbf{B}\mathbf{C}_0^\dagger \mathbf{B}^T + \sigma \mathbf{I} = \mathbf{I} \otimes (\mathbf{L}^\dagger + \sigma \mathbf{I}), \tag{14}$$

where  $\mathbf{L}^\dagger = \chi \mathbf{A}^\dagger \chi^T$ .

The posterior density of  $\mathbf{h}$  is obtained by applying the Bayes rule as follows

<sup>3</sup> we used  $\sigma$  to refer to the noise variance instead of commonly used  $\sigma^2$  which is mostly used in the literature. Hereafter, we also remove the subscript notation of  $\mathbf{0}_N, \mathbf{1}_N$  and  $\mathbf{I}_N$  and assume that the reader infers the corresponding dimensions from the context. The reason is the notation simplification in the next sections.

**Fig. 3** The effect of denoising the graph signals when using a correct noise variance estimator



$$p(\mathbf{h} | \mathbf{x}; \mathbf{A}^\dagger, \mathcal{X}, \sigma) = \frac{p(\mathbf{x} | \mathbf{h}; \mathcal{X}, \sigma)p(\mathbf{h}; \mathbf{A}^\dagger)}{p(\mathbf{x}; \mathbf{A}^\dagger, \mathcal{X}, \sigma)}, \tag{15}$$

and the Minimum Mean Square Error (MMSE) estimator of the latent variable is defined as below [55]

$$\hat{\mathbf{h}} = \frac{1}{\sigma} \mathbf{C}_\epsilon \mathbf{B}^T \mathbf{x}, \tag{16}$$

where the covariance matrix is

$$\mathbf{C}_\epsilon = \left( \mathbf{C}_0 + \frac{1}{\sigma} \mathbf{B}^T \mathbf{B} \right)^{-1} = \left( \mathbf{C}_0 + \frac{1}{\sigma} \mathbf{I} \right)^{-1} = \mathbf{I} \otimes \left( \mathbf{A} + \frac{1}{\sigma} \mathbf{I} \right)^{-1} \tag{17}$$

If  $\sigma$  increases, the distributions in (11) and (12) have larger uncertainties and thus the estimation of (16) is less accurate and has a larger MSE. Due to the linear relationship between  $\mathbf{y}$  and  $\mathbf{h}$ , i.e.  $\mathbf{y} = \mathbf{B}\mathbf{h}$ , the graph signals can be estimated by  $\hat{\mathbf{y}} = \mathbf{B}\hat{\mu}_h$ , which can be simplified as follows

$$\hat{\mathbf{y}} = (\mathbf{I} \otimes (\mathbf{I} + \sigma \mathbf{L}))^{-1} \mathbf{x}, \tag{18}$$

where  $\sigma$  and  $\mathbf{L}$  will be estimated later. Equivalently, (18) can be represented in a matrix form as  $\hat{\mathbf{Y}} = (\mathbf{I} + \sigma \mathbf{L})^{-1} \mathbf{X}$  which is similar to the one presented in [51], where  $\hat{\mathbf{Y}} = (\mathbf{I} + \alpha \mathbf{L})^{-1} \mathbf{X}$  for a parameter  $\alpha$ . In [51],  $\alpha$  is the regularization parameter corresponding to the smoothness of graph signals over the topology. In other words, the graph signals  $\mathbf{X}$  are smoothed by the graph filter  $(\mathbf{I} + \sigma \mathbf{L})^{-1}$  and hence  $\sigma = \alpha$  can be called the filter coefficient or the regularization parameter or noise variance, interchangeably. If  $\sigma$  approaches zero,  $\hat{\mathbf{Y}}$  and  $\mathbf{X}$  are going to be identical and for larger  $\sigma$ , the effect of graph Laplacian matrix on filtering the signal is larger. In other words, we have

$$\hat{\mathbf{Y}} = (\mathbf{I} + \sigma \mathbf{L})^{-1} (\mathbf{Y} + \mathbf{E}) = \mathbf{Y} + \mathbf{E} - \sigma \mathbf{L} \hat{\mathbf{Y}}, \tag{19}$$

and thus for the true denoised version of  $\mathbf{Y}$ , i.e.  $\hat{\mathbf{Y}} = \mathbf{Y}$ , we have  $\mathbf{E} = \sigma \mathbf{L} \hat{\mathbf{Y}}$ . Given a fixed error  $\mathbf{E}$ , if  $\sigma$  is large, then  $\hat{\mathbf{Y}}$  is sufficiently smooth hence  $\mathbf{L} \hat{\mathbf{Y}}$  will likely be small. This is the case where we make a big effort to denoise the observation  $\mathbf{X}$ . If  $\sigma$  is small, the opposite will be true. So this shows that given a noisy observation (hence the amount of noise added to the clean signal is “fixed”), different  $\sigma$  estimation approaches (or equivalently  $\alpha$  adjustment in [51]) will lead to different denoising effect. However,  $\sigma$  has not been investigated analytically in [51] and it was estimated by a grid search.

To make the importance of noise variance estimation more clear, we generate a graph with  $N = 50$  nodes with  $K = 100$  graph signals, i.e.  $\mathbf{Y}$ , via the procedure explained in the simulation Sect. 6.1. Then, the graph signals are contaminated by Gaussian noises with difference variances to provide  $\mathbf{X}$ . Given  $\mathbf{X}$  and the known Laplacian matrix  $\mathbf{L}$ , we try to denoise the graph signals via (18) by two different filters, i.e. employing the correct variance and an incorrect estimated variance which is half of the true value. This procedure is repeated for 10 different variances and the

normalized mean square error of the true signals and the estimated ones are computed. The results in Fig. 3 show how we can improve our measurements by the filter in (18), especially when we have a better estimation of the noise variance. Since the Laplacian matrix estimation is performed in the next step based on these measurements, using denoised graph signals certainly helps in a better topology estimation.

To estimate the parameters  $\sigma$  and  $\mathbf{A}^\dagger$ , an expectation maximization procedure is used which proceeds by optimizing the following log-likelihood function

$$Q(\mathbf{A}^\dagger, \chi, \sigma) = \mathbb{E}_{\mathbf{h}|\mathbf{x}; \hat{\sigma}, \hat{\chi}, \hat{\mathbf{A}}^\dagger} [\log p(\mathbf{x}, \mathbf{h} | \sigma, \chi, \mathbf{A}^\dagger)], \tag{20}$$

where  $\mathbb{E}(\cdot)$  denotes the expectation and  $\hat{\mathbf{A}}^\dagger$ ,  $\hat{\chi}$ , and  $\hat{\sigma}$  are the estimators of  $\mathbf{A}^\dagger$ ,  $\chi$ , and  $\sigma$ . Hereafter, for brevity of notation,  $\Theta$  denotes all the parameters, i.e.  $\Theta := (\sigma, \chi, \mathbf{A}^\dagger) = (\sigma, \mathbf{L})$ . By using (10)–(12) and (17) and some manipulations, (20) is rewritten as follows (see Appendix 1 for more details)

$$Q(\Theta) = \mathbb{E}_{\mathbf{h}|\mathbf{x}; \hat{\sigma}} [\log p(\mathbf{x} | \mathbf{h})] + \mathbb{E}_{\mathbf{h}|\mathbf{x}; \hat{\sigma}} [\log p(\mathbf{h})] \\ \alpha - N \log \sigma - \text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1}) + \log |\mathbf{L}| - \text{Tr}(\mathbf{S}\mathbf{L}), \tag{21}$$

where  $\mathbf{S} = \frac{1}{K} \hat{\mathbf{Y}} \hat{\mathbf{Y}}^T$  is the empirical covariance matrix of the graph signals and the pseudo-determinant  $|\cdot|$  is used due to the singularity of  $\mathbf{A}$  and  $\mathbf{C}_0$ . The pseudo-determinant of a matrix is the product of its non-zero eigenvalues. To find the optimal parameters,  $Q$  must be optimized with respect to each argument iteratively up to a convergence. By taking the derivative with respect to  $\sigma$ , we have

$$-\frac{N}{\sigma} + \text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1} \cdot \mathbf{L} \cdot (\sigma \mathbf{L} + \mathbf{I})^{-1}) = 0, \tag{22}$$

where the optimal  $\sigma$  is found by a numerical method, e.g. Newton–Raphson algorithm. Then,  $Q$  is maximized with respect to the Laplacian matrix as follows

$$\underset{\mathbf{L}}{\text{argmax}} \quad \log |\mathbf{L}| - \text{Tr}(\mathbf{S}\mathbf{L}) - \text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1}) \\ \text{s.t.} \quad \mathbf{L}_{ij} = \mathbf{L}_{ji}, \quad \mathbf{L}_{ij} \leq 0 \text{ if } i \neq j, \quad \mathbf{L} \cdot \mathbf{1} = \mathbf{0}, \quad \text{Tr}(\mathbf{L}) = c, \tag{23}$$

where the first three constraints ensure a valid Laplacian matrix and the last one avoids trivial solution for  $c > 0$ . The second term of the objective function promotes the signal smoothness over the estimated topology [as explained in (4)]. The optimization problem in (23) is not easy to implement due to the first and third terms. To solve the issue with the pseudo determinant term, we propose the following minimization problem

$$\underset{\mathbf{L}}{\text{argmin}} \quad -\log \det(\mathbf{L} + \mathbf{J}) + \text{Tr}(\mathbf{S}\mathbf{L}) + \text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1}) \\ \text{s.t.} \quad \mathbf{L}_{ij} = \mathbf{L}_{ji}, \quad \mathbf{L}_{ij} \leq 0 \text{ if } i \neq j, \quad \mathbf{L} \cdot \mathbf{1} = \mathbf{0}, \quad \text{Tr}(\mathbf{L}) = c, \tag{24}$$

where  $\det(\cdot)$  stands for the determinant and  $\mathbf{J} = \frac{1}{N} \mathbf{1}\mathbf{1}^T$ . The equivalence of  $\log \det(\mathbf{L} + \mathbf{J})$  and  $\log |\mathbf{L}|$  has been justified in [22]. The matrix inverse term which needs high computational power can also be replaced by a less computationally expensive term. Theorem 1 of [56] proposed upper and lower bounds for the trace of symmetric positive definite matrices inverse as

$$\text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1}) \leq N - \frac{c^2 \sigma}{\sigma \|\mathbf{L}\|_F^2 - c}, \tag{25}$$

$$\text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1}) \geq \frac{N}{2\sigma + 1} - \frac{\frac{c^2 \sigma - 2cN\sigma + 2N^2 \sigma}{2\sigma + 1}}{\sigma \|\mathbf{L}\|_F^2 - c - 2N - 2c\sigma} \tag{26}$$

and thus for a fixed  $\sigma$ , minimizing  $\sigma \|\mathbf{L}\|_F^2$  results in the minimization of  $\text{Tr}((\sigma \mathbf{L} + \mathbf{I})^{-1})$ . To sum up, we propose to solve the following minimization problem for the Laplacian matrix estimation

$$\begin{aligned} \underset{\mathbf{L}}{\operatorname{argmin}} \quad & -\log\det(\mathbf{L} + \mathbf{J}) + \operatorname{Tr}(\mathbf{S}\mathbf{L}) + \sigma\|\mathbf{L}\|_F^2 \\ \text{s.t.} \quad & \mathbf{L}_{ij} = \mathbf{L}_{ji}, \quad \mathbf{L}_{ij} \leq 0 \text{ if } i \neq j, \quad \mathbf{L} \cdot \mathbf{1} = \mathbf{0}, \operatorname{Tr}(\mathbf{L}) = c, \end{aligned} \quad (27)$$

where  $\|\mathbf{L}\|_F^2$  can also be considered as the control term for the distribution of off-diagonal elements, i.e. the edge weights of the estimated graph. Since (27) is a convex optimization problem, any off the shelf convex solver may be used, e.g. YALMIP [57]. However, in the next section, a proximal point algorithm is proposed to solve (27) efficiently. To conclude this section, the three steps of the proposed method are presented in Algorithm 1.

---

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

---

**Input:** Given measurements  $\mathbf{X}$

**Initialize:**  $\hat{\mathbf{Y}} = \mathbf{X}$ ,  $\hat{\sigma} = \sigma_0$ , and  $c$

**while** *Not Converged* **do**

    1: Laplacian matrix learning by solving (27),

    2: Find  $\sigma$  via (22),

    3: Signal recovery:  $\hat{\mathbf{Y}} = (\mathbf{I} + \hat{\sigma}\hat{\mathbf{L}})^{-1}\mathbf{X}$ .

**end**

**Output:**  $\hat{\mathbf{L}}$  and  $\hat{\mathbf{Y}}$ .

---

## 4 The efficient implementation

In this section, we discuss milestones to implement the proposed method with a fast and efficient algorithm. The proposed algorithm to solve (27) follows these steps: first by Proposition 2, the last three constraints in (27) are rewritten in the form of inner product of two matrices, helping to form a simpler Lagrangian function. Then, the Lagrangian is iteratively maximized with respect to the dual variable and then minimized with respect to the primal variable, i.e. the graph Laplacian matrix. To do the minimization step, we apply a proximal point algorithm. The maximization can be done by the Newton or quasi-Newton methods where the derivative and Hessian with respect to the dual variable are obtained by Lemma 2 and 1. Theorem 1 finds the stopping criterion guaranteeing the convergence of iteration between the maximization and minimization problem.

**Proposition 1** *Since  $\mathbf{L}$  is a symmetric matrix, the summation over all entries of the  $i$ 'th row (or  $i$ 'th column) is rewritten as*

$$\sum_{j=1}^N L_{ij} = \frac{1}{2} \operatorname{Tr}(\mathbf{L}(\mathbf{U}_i + \mathbf{U}_i^T)) \quad (28)$$

where  $\mathbf{U}_i$  is an  $N \times N$  matrix in which the  $i$ 'th column is the all one vector and the rest is zero.

**Proposition 2** *Having,  $\mathbf{L} \cdot \mathbf{1} = \mathbf{0}$ , the constraint  $\mathbf{L}_{ij} \leq 0$  for  $i \neq j$  in the problem (27) can be replaced by  $\|\mathbf{L}\|_1 = 2\operatorname{Tr}(\mathbf{L})$ , where  $\|\mathbf{L}\|_1 = \sum_{i,j} \operatorname{abs}(L_{ij})$  and  $\mathbf{L} \in \mathcal{S}_+^N$ .*

**Proof** The weight matrix  $\mathbf{W}$  is a matrix with zero diagonal entries as long as we have assumed that there is no self loop. If all edges are positive (and so  $\mathbf{L}_{ij} \leq 0$  for  $i \neq j$ ), we have  $\|\mathbf{D}\|_1 = \|\mathbf{W}\|_1$  and

$$\|\mathbf{L}\|_1 = \|\mathbf{D} - \mathbf{W}\|_1 = \|\mathbf{D}\|_1 + \|\mathbf{W}\|_1 = 2\|\mathbf{D}\|_1 = 2\operatorname{Tr}(\mathbf{D}) = 2\operatorname{Tr}(\mathbf{L}). \quad (29)$$

If there is only one edge with negative weight (or correspondingly  $\exists i \neq j, \mathbf{L}_{ij} > 0$ ),  $\|\mathbf{D}\|_1 \neq \|\mathbf{W}\|_1$ .

Note that if all edges are negative and thus  $\mathbf{L}_{ij} \geq 0$  and  $\mathbf{L}_{ii} \leq 0$ , this property is proved in the same way. However, this is not the case here as long as it is assumed that  $\mathbf{L} \in \mathcal{S}_+^N$ , forcing the diagonal entries of  $\mathbf{L}$  to be non-negative.  $\square$

By applying the Propositions 1 and 2, the minimization problem of (27) is rewritten in the following form

$$\begin{aligned} \underset{\mathbf{L} \in \mathcal{S}_+^N}{\operatorname{argmin}} \quad & -\log\det(\mathbf{L} + \mathbf{J}) + \operatorname{Tr}(\mathbf{L}\mathbf{S}) + \sigma\|\mathbf{L}\|_F^2 \\ \text{s.t.} \quad & \mathcal{B}(\mathbf{L}) = \mathbf{a}, \end{aligned} \tag{30}$$

where  $\mathbf{a} = [\mathbf{0}; \quad c; \quad 2c]^T$ , and  $\mathcal{B}(\cdot) : \mathcal{R}^{N \times N} \rightarrow \mathcal{R}^{(N+2) \times 1}$  is a matrix operator as follows

$$\mathcal{B}(\mathbf{L}) = \left[ \frac{1}{2}\operatorname{Tr}(\mathbf{L}(\mathbf{U}_1 + \mathbf{U}_1^T)), \dots, \frac{1}{2}\operatorname{Tr}(\mathbf{L}(\mathbf{U}_N + \mathbf{U}_N^T)), \operatorname{Tr}(\mathbf{L}), \quad \|\mathbf{L}\|_1 \right]^T, \tag{31}$$

Hereafter, we interchangeably use the inner product of two matrices instead of the trace operator, i.e.  $\operatorname{Tr}(\mathbf{L}_1 \cdot \mathbf{L}_2) = \langle \mathbf{L}_1, \mathbf{L}_2 \rangle$ . It is assumed that the feasible set  $\mathcal{F} = \{\mathbf{L} \in \mathcal{S}_+^N, \quad \mathcal{B}(\mathbf{L}) = \mathbf{a}\}$  is not empty and then due to the convexity of (30), any local optimal solution is also global optimal. The Lagrangian function over the primal variable  $\mathbf{L}$  and the dual variable  $\mathbf{v} = [v_1, \dots, \quad v_N, \quad v_{N+1}, \quad v_{N+2}]$  is given as below

$$\mathcal{L}(\mathbf{L}; \mathbf{v}) = -\log\det(\mathbf{L} + \mathbf{J}) + \operatorname{Tr}(\mathbf{L}\mathbf{S}) + \sigma\|\mathbf{L}\|_F^2 + \mathbf{v}^T(\mathbf{a} - \mathcal{B}(\mathbf{L})), \tag{32}$$

Since the objective function of (30) is convex and there exists a strictly feasible point (in  $\mathcal{F}$ ), Slater’s condition holds and there is a strong duality or saddle point property [58]. Strong duality means that it is possible to find the optimal of the dual problem instead of the primal one. Thus,  $\mathbf{L}$  can be estimated via the following optimization problem

$$\hat{\mathbf{L}} = \underset{\mathbf{L} \in \mathcal{S}_+^N}{\operatorname{argmin}} \quad g(\mathbf{L}), \tag{33}$$

where

$$g(\mathbf{L}) = \max_{\mathbf{v}} \quad \mathcal{L}(\mathbf{L}; \mathbf{v}). \tag{34}$$

However, it is difficult to implement (33) directly. Therefore, we propose to use the proximal algorithm, which is a standard tool for solving constrained and nonsmooth minimization problems [59]. The proximal operator of a closed proper convex function  $g : \mathcal{R}^n \rightarrow \mathcal{R} \cup \{+\infty\}$  is represented by  $\operatorname{prox}_g : \mathcal{R}^n \rightarrow \mathcal{R}^n$  and its scaled version with parameter  $\eta$  is given as below

$$\operatorname{prox}_{\eta g}(\mathbf{x}) = \underset{\mathbf{v}}{\operatorname{argmin}} \quad \left( g(\mathbf{v}) + \frac{1}{2\eta}\|\mathbf{x} - \mathbf{v}\|_2^2 \right). \tag{35}$$

One of the tools for solving the general optimization problem  $\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \quad g(\mathbf{x})$  is the proximal point algorithm, also called proximal iteration, given by

$$\mathbf{x}^{(t+1)} := \operatorname{prox}_{\eta g}(\mathbf{x}^{(t)}) \tag{36}$$

where  $t$  is the iteration index. In this algorithm,  $\mathbf{x}^k$  and  $g(\mathbf{x}^k)$  converge to the set of minimizers of  $g$  and its optimal value, respectively [60]. The proximal operator provides a smoothed version of the function by adding the regularized term. The proximal operator can also be computed by [59]

$$\operatorname{prox}_{\eta g}(\mathbf{x}) = \mathbf{x} - \eta \nabla M_{\eta g}(\mathbf{x}), \tag{37}$$

where  $\nabla$  is the gradient operator and  $M_{\eta g}(\mathbf{x})$  is the Moreau–Yosida regularization. The Moreau–Yosida regularization of  $g(\mathbf{L})$  in (33) with parameter  $\eta > 0$  is defined as [59, 61, 62]

$$\begin{aligned} M_{\eta g}(\mathbf{L}) &= \min_{\mathbf{L}' \in \mathcal{S}_+^N} \left\{ g(\mathbf{L}') + \frac{1}{2\eta}\|\mathbf{L} - \mathbf{L}'\|_F^2 \right\} \\ &\stackrel{(a)}{=} \max_{\mathbf{v}} \min_{\mathbf{L}' \in \mathcal{S}_+^N} \left\{ \mathcal{L}(\mathbf{L}'; \mathbf{v}) + \frac{1}{2\eta}\|\mathbf{L} - \mathbf{L}'\|_F^2 \right\} \end{aligned} \tag{38}$$

where  $\stackrel{(a)}{=}$  follows from Von Neumann–Fan minimax theorem [63, 64]. Like the proximal operator, the Moreau–Yosida regularization provides a smooth version of the function. Moreover,  $M_{\eta g}(\mathbf{L})$  and  $g(\mathbf{L})$  have the same set of minimizers

and thus we solve (38) instead of (34), equivalently. In particular, we propose to find the minimum of  $M_{\eta\sigma}(\mathbf{L})$  by applying the proximal point algorithm in (36).

#### 4.1 Finding $\mathbf{P}_\eta(\mathbf{L};\mathbf{v})$

The adjoint operator of  $\mathcal{B}$ , called hereafter  $\mathcal{B}_a$ , is obtained as follows

$$\mathcal{B}_a(\mathbf{v}) = \frac{\partial \langle \mathbf{L}, \mathcal{B}_a(\mathbf{v}) \rangle}{\partial \mathbf{L}} \stackrel{(a)}{=} \frac{\partial \langle \mathcal{B}(\mathbf{L}), \mathbf{v} \rangle}{\partial \mathbf{L}}, \quad (39)$$

where  $\frac{\partial}{\partial \mathbf{L}}$  denotes the partial derivative with respect to  $\mathbf{L}$  and (a) follows the inner product property. By using (31) and some simple manipulations, we have

$$\mathcal{B}_a(\mathbf{v}) = \frac{1}{2}v_1(\mathbf{U}_1 + \mathbf{U}_1^T) + \dots + \frac{1}{2}v_N(\mathbf{U}_N + \mathbf{U}_N^T) + v_{N+1}\mathbf{I} + v_{N+2}(2\mathbf{I} - N\mathbf{J}). \quad (40)$$

To simplify (38), we introduce the change of variable  $\mathbf{T}_\eta$  as

$$\mathbf{T}_\eta(\mathbf{L};\mathbf{v}) \triangleq \frac{1}{1 + 2\eta\sigma}(\mathbf{L} - \eta(\mathbf{S} - \mathcal{B}_a(\mathbf{v}))), \quad (41)$$

and then  $\mathbf{P}_\eta(\mathbf{L};\mathbf{v})$  can be rewritten as follows

$$\mathbf{P}_\eta(\mathbf{L};\mathbf{v}) = \mathbf{v}^T \mathbf{a} + \frac{1}{2\eta} \|\mathbf{L}\|_F^2 - \frac{1 + 2\eta\sigma}{2\eta} \|\mathbf{T}_\eta(\mathbf{L};\mathbf{v})\|_F^2 + \min_{\mathbf{L}'} \mathcal{J}_\eta(\mathbf{L}', \mathbf{L};\mathbf{v}) \quad (42)$$

where

$$\mathcal{J}_\eta(\mathbf{L}', \mathbf{L};\mathbf{v}) = -\log \det(\mathbf{L}') + \frac{1 + 2\eta\sigma}{2\eta} \|\mathbf{L}' - \mathbf{T}_\eta(\mathbf{L};\mathbf{v})\|_F^2. \quad (43)$$

To find the minimizer of  $\mathcal{J}_\eta(\mathbf{L}', \mathbf{L};\mathbf{v})$  and simplify (42), the following Lemma from [65] will be used.

**Lemma 1** [65]: Let  $\mathbf{Z} \in \mathcal{S}^N$  with eigenvalue decomposition  $\mathbf{Z} = \mathbf{P}\boldsymbol{\Theta}\mathbf{P}^T$  and  $\gamma > 0$  where  $\boldsymbol{\Theta} = \text{diag}(\boldsymbol{\theta})$  is the eigenvalue matrix. Assume two scalar functions  $\phi_\gamma^+(x) \triangleq \frac{1}{2}(\sqrt{x^2 + 4\gamma} + x)$  and  $\phi_\gamma^-(x) \triangleq \frac{1}{2}(\sqrt{x^2 + 4\gamma} - x)$  are defined and their matrix counterparts are as follows

$$\begin{aligned} \mathbf{Z}_1 &= \phi_\gamma^+(\mathbf{Z}) = \mathbf{P} \text{diag}(\phi_\gamma^+(\boldsymbol{\theta})) \mathbf{P}^T \\ \mathbf{Z}_2 &= \phi_\gamma^-(\mathbf{Z}) = \mathbf{P} \text{diag}(\phi_\gamma^-(\boldsymbol{\theta})) \mathbf{P}^T \end{aligned} \quad (44)$$

Then,

1.  $\mathbf{Z} = \mathbf{Z}_1 - \mathbf{Z}_2$  and  $\mathbf{Z}_1\mathbf{Z}_2 = \gamma\mathbf{I}$
2.  $\phi_\gamma^+$  is continuously differentiable and its derivative at  $\mathbf{Z}$  for every  $\mathbf{H} \in \mathcal{S}^N$  is given as

$$\left. \frac{\partial \phi_\gamma^+}{\partial x} \right|_{x=\mathbf{Z}} \cdot [\mathbf{H}] = \mathbf{P}(\boldsymbol{\Omega} \circ (\mathbf{P}^T \mathbf{H} \mathbf{P})) \mathbf{P}^T, \quad (45)$$

where  $\circ$  denotes the Hadamard product and  $\boldsymbol{\Omega} \in \mathcal{S}^N$  is defined as follows

$$\Omega_{ij} = \frac{\phi_\gamma^+(\theta_i) + \phi_\gamma^+(\theta_j)}{\sqrt{\theta_i^2 + 4\gamma} + \sqrt{\theta_j^2 + 4\gamma}}, \quad 1 < i, j < N. \quad (46)$$

$$3. \quad \left. \frac{\partial \phi_{\gamma}^+}{\partial x} \right|_{x=\mathbf{Z}} \cdot [\mathbf{Z}_1 + \mathbf{Z}_2] = \phi_{\gamma}^+(\mathbf{Z}). \tag{47}$$

**Proof** See [65]. □

To find the minimizer of  $\mathcal{J}_{\eta}(\mathbf{L}', \mathbf{L}; \mathbf{v})$ , the derivative with respect to  $\mathbf{L}'$  is set to zero as follows

$$-(\mathbf{L}')^{-1} + \frac{1 + 2\eta\sigma}{\eta} (\mathbf{L}' - \mathbf{T}_{\eta}(\mathbf{L}; \mathbf{v})) \tag{48}$$

and by some simple manipulations, the solution is  $\mathbf{L}' = \phi_{\gamma'}^+(\mathbf{T}_{\eta}(\mathbf{L}; \mathbf{v}))$  where  $\gamma' = \frac{\eta}{1+2\eta\sigma}$ . Then (42) is simplified as follows (see Appendix 2)

$$\mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v}) = \mathbf{v}^T \mathbf{a} + \frac{1}{2\eta} \|\mathbf{L}\|_F^2 - \log \det \left( \phi_{\gamma'}^+(\mathbf{T}_{\eta}(\mathbf{L}; \mathbf{v})) \right) - \frac{1}{2\gamma'} \left\| \phi_{\gamma'}^+(\mathbf{T}_{\eta}(\mathbf{L}; \mathbf{v})) \right\|_F^2 + N. \tag{49}$$

### 4.2 Find $M_{\eta g}(\mathbf{L})$ via (38)

**Lemma 2** The derivative of  $\mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v})$  with respect to  $\mathbf{v}$  is

$$\nabla_{\mathbf{v}} \mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v}) = \mathbf{a} - \mathcal{B}(\Phi^+), \tag{50}$$

where  $\Phi^+ := \phi_{\gamma'}^+(\mathbf{T}_{\eta}(\mathbf{L}; \mathbf{v}))$ .

**Proof** See Appendix 3. □

By taking the derivative of  $\nabla_{\mathbf{v}} \mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v})$  with respect to  $\mathbf{v}$  and applying (45), the following corollary follows.

**Corollary 1** The Hessian of  $\mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v})$  with respect to  $\mathbf{v}$  is

$$\begin{aligned} \nabla_{\mathbf{v}\mathbf{v}}^2 \mathbf{P}_{\eta} = & -\gamma' [\mathcal{B}((\Phi^+)' \cdot [\frac{1}{2}(\mathbf{U}_1 + \mathbf{U}_1^T)]), \dots, \mathcal{B}((\Phi^+)' \cdot [\frac{1}{2}(\mathbf{U}_N + \mathbf{U}_N^T)])], \\ & \mathcal{B}((\Phi^+)' \cdot [\mathbf{I}]), \mathcal{B}((\Phi^+)' \cdot [2\mathbf{I} - \mathbf{1}\mathbf{1}^T]), \end{aligned} \tag{51}$$

Using the first and second order derivatives of  $\mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v})$  with respect to  $\mathbf{v}$ , the unconstrained maximization problem of (38) can be solved by Newton or quasi-Newton methods, like L-BFGS. Let  $\mathbf{v}_{\text{opt}}$  denote  $\underset{\mathbf{v}}{\text{argmax}} \mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v})$  and by using (38), we have  $M_{\eta g}(\mathbf{L}) = \mathbf{P}_{\eta}(\mathbf{L}; \mathbf{v}_{\text{opt}})$ .

**Lemma 3** The derivative of  $M_{\eta g}(\mathbf{L})$  with respect to the graph Laplacian matrix is

$$\nabla M_{\eta g}(\mathbf{L}) = \frac{1}{\eta} \left( \mathbf{L} - \phi_{\gamma'}^+(\mathbf{T}_{\eta}(\mathbf{L}; \mathbf{v}_{\text{opt}})) \right). \tag{52}$$

**Proof** The result follows by taking the derivative of (49) with respect to  $\mathbf{L}$  and applying Lemma 1, part (3), similar to the proof of Lemma 2. □

Considering (36), (37), and (52), the graph Laplacian matrix is estimated via the following iteration rule

$$\mathbf{L}^{(t+1)} = \phi_{\gamma'}^+(\mathbf{T}_{\eta}(\mathbf{L}^{(t)}; \mathbf{v}_{\text{opt}}^{(t+1)})). \tag{53}$$

where  $(t)$  stands for the iteration index. All steps to update the Laplacian matrix are summarized in Algorithm 2. It is also possible to use  $\eta_t$  rather than  $\eta$  to show the possibility of adjustment in each iteration for a faster convergence of the objective variable  $\mathbf{L}$ . To find  $\eta_t$  in each iteration, the exact or backtracking line search can be applied [58, 59].

---

**Algorithm 2** Bayesian Topology Learning via Proximal Point Algorithm (BTL-PPA)

---

**Input:**  $\hat{\mathbf{Y}}, \mathbf{a}, \mathbf{L}^{(0)} \in \mathcal{S}_+^N$ , and  $e_{\text{thr}} > 0$ .

**while** *stopping criterion is not satisfied* **do**

- 1: Find the optimal dual variable at the current iteration by  $\mathbf{v}_{\text{opt}}^{(t+1)} = \underset{\mathbf{v}}{\operatorname{argmax}} \mathbf{P}_\eta(\mathbf{L}^{(t)}; \mathbf{v})$ ,
- 2: Update the Laplacian matrix via (53),
- 3: If  $\|\mathbf{L}^{(t+1)} - \mathbf{L}^{(t)}\|_F^2 / \eta_t \leq e_{\text{thr}}$ , stop,

**end**

**Output:** The graph Laplacian matrix  $\mathbf{L}$ .

---

## 5 Convergence analysis

Assume that the eigendecomposition of  $\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})$  is represented as  $\mathbf{T}_\eta = \mathbf{P}\mathbf{O}\mathbf{P}^T$ . Hence, we have

$$\phi_\gamma^+(\mathbf{T}_\eta) = \mathbf{P}\operatorname{diag}(\phi_\gamma^+(\boldsymbol{\theta}))\mathbf{P}^T. \quad (54)$$

where the scalar function  $\phi_\gamma^+(\cdot)$  is a convex function and also bounded with bounded eigenvalues. From theorems B and C in [66], it follows that  $\phi_\gamma^+(\boldsymbol{\theta})$  is Lipschitz continuous. Thus,

$$\left\| \phi_{\gamma'}^+(\mathbf{T}_\eta^{(t)}) - \phi_{\gamma'}^+(\mathbf{T}_\eta^{(t-1)}) \right\|_F^2 \leq l_c \left\| \mathbf{T}_\eta^{(t)} - \mathbf{T}_\eta^{(t-1)} \right\|_F^2, \quad (55)$$

where  $l_c$  is the Lipschitz constant and  $\mathbf{T}_\eta^{(t)} = \mathbf{T}_\eta(\mathbf{L}^{(t)}; \mathbf{v}_{\text{opt}}^{(t)})$ .

**Lemma 4** *The scalar function  $\phi_\gamma^+(x)$  is Lipschitz continuous with the Lipschitz constant  $\frac{3}{2}$ .*

*Proof* See Appendix 4. □

**Lemma 5** *If  $\mathbf{Z}$  is a real symmetric matrix with the eigenvalue matrix  $\boldsymbol{\Theta}_Z$ , then  $\|\mathbf{Z}\|_F^2 = \|\boldsymbol{\Theta}_Z\|_F^2$ .*

*Proof* See Appendix 5. □

**Corollary 2** *The matrix valued function  $\phi_\gamma^+(\mathbf{Z})$  is Lipschitz continuous with the constant  $l_c = \frac{9}{4}$ .*

*Proof* The proof follows readily by applying the eigenvalue decomposition (44) and using Lemma 4 and 5. □

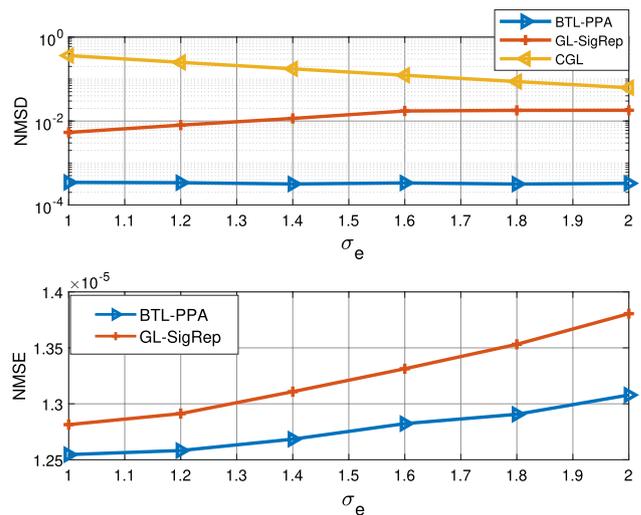
**Theorem 1** *The proposed BTL-PPA converges when the following stopping criterion is set for dual variables update*

$$\left\| \mathbf{v}^{(t)} - \mathbf{v}^{(t_*)} \right\|_2^2 \leq \frac{1}{\eta^2 N^2} \left\| \mathbf{L}^{(t)} - \mathbf{L}^{(t_*)} \right\|_F^2, \quad (56)$$

where the optimum primal and dual variables are  $\mathbf{L}^{(t_*)}$  and  $\mathbf{v}^{(t_*)}$ , respectively.

*Proof* See Appendix 6. □

**Fig. 4** The capability of different algorithms in estimating the graph topology and denoising the signals. Top: The normalized mean square deviation of graph topology estimation; Bottom: The normalized mean square error of signal recovery ( $N = 40$  and  $K = 4000$ )



## 6 Numerical results

The proposed algorithm is tested for simulated and real data for different scenarios. For topology learning performance, the results of our algorithm are compared to those of three existing algorithms: GL-SigRep [51], CGL [22], and the learning sparse graph algorithm in [20], called LSG here. In synthetic data simulation part, although the performance of the LSG algorithm is similar to the BTL-PPA, it has a higher computational complexity compared to the other three algorithms. The results for signal recovery performance are only compared to those of GL-SigRep, because the other two methods have no policy for signal representation and recovery. To implement GL-SigRep, an optimal selection of the regularization parameters via exhaustive search is applied and thus the appropriate  $\frac{\alpha}{\beta}$  ratio is found in order to maximize the performance for the algorithm in [51].

### 6.1 Synthetic data

The synthetic data is drawn from a Gaussian Markov Random Field process. First, an Erdős–Rényi graph is generated with  $N = 40$  vertices and an edge probability of 0.2. The weight matrix and then the graph Laplacian matrix are computed and normalized by its trace. Then, each data vector is sampled from a N-variate Gaussian distribution  $\mathbf{y}[k] \sim \mathcal{N}(\mathbf{0}, \mathbf{L}^\dagger)$  and contaminated by independent and identically distributed Gaussian noise. The measurements are  $\mathbf{x}[k] = \mathbf{y}[k] + \mathbf{e}[k]$  for  $k = 1, \dots, 4000$  and different value of noise variance  $\sigma$ . The measurements are stacked in the columns of the matrix  $\mathbf{X}$ , which is the given input to Algorithm 1. We set  $c = N$  [51] and initialize  $\sigma$  to 1. Finally, the simulation results are averaged over 100 different trials of experiments.

The most important performance measures to compare different algorithms in this literature are as follows [51, 67–70]:

- Normalized Mean Squared Deviation of graph topology estimation:  $NMSD = \frac{1}{N^2} \cdot \frac{\|\mathbf{L} - \hat{\mathbf{L}}\|_F^2}{\|\mathbf{L}\|_F^2}$ , where  $\hat{\mathbf{L}}$  denotes the estimated Laplacian matrix,
- Normalized Mean Squared Error of signal reconstruction:  $NMSE = \frac{1}{N \cdot K} \cdot \frac{\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2}{\|\mathbf{X}\|_F^2}$ ,
- F-measure =  $\frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FN + FP}$ , where TP, FP and FN are the numbers of true positives, false positives, and false negatives, respectively. Also, precision is the number of correctly recovered edges to the number of reconstructed edges in the estimated graph and recall is the number of correctly recovered edges to the number of edges in the ground-truth graph. This performance measure solely takes into account the support of the recovered graph while ignoring the weights,
- Normalized Mutual Information:  $NMI(\mathbf{L}, \hat{\mathbf{L}}) = \frac{2MI(\mathbf{L}, \hat{\mathbf{L}})}{H(\mathbf{L}) + H(\hat{\mathbf{L}})}$ , where MI is the mutual information between the set of edges in the estimated graph and the true graph and  $H(\mathbf{L})$  is the entropy based on the probability distribution of edges, i.e. the probability of zeros and ones in the matrix  $\mathbf{L}$  [70].

**Fig. 5** The capability of different algorithms in finding the correct edges; Top: F-measure; Bottom: Normalized Mutual Information

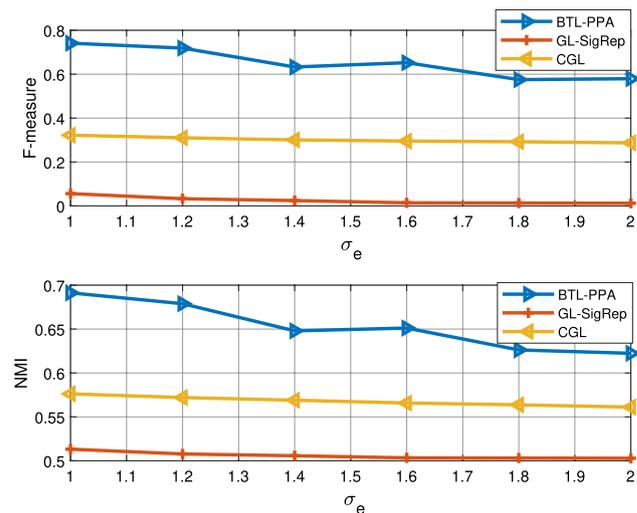


Figure 4 shows that BTL-PPA outperforms GL-SigRep for different noise variances and its NMSE are lower than those of GL-SigRep. Considering the NMSD comparison, the BTL-PPA topology learning algorithm works very well compared to other algorithms and its topology learning capability is robust in different noise powers. The experiments are run for different values of noise variances, for a fixed signal power. Hence, the horizontal axes can also be linked to different signal to noise (SNR) ratio. As shown in Fig. 4, when the noise increases, the NMSD for CGL algorithm decreases which seems to be counter-intuitive at first glance. However, it is necessary to note that for the current chosen threshold to remove weak edges, we get more edges leading to a lower NMSD (due to less variance in edge weights) and at the same time, as shown in Fig. 5, F-measure is decreasing due to an increasingly lower precision. Figure 5 also corroborates the strong performance of the proposed algorithm from the perspective that how much percentage of edges are learned correctly.

## 6.2 Signal inpainting in IoT applications

In the IoT framework, a challenging area of research is preprocessing of data after sensing and before providing it to further processing (for more detail, please see [1] Figure 3). One of these preprocessing steps is to estimate the missing values, i.e. signal inpainting. These missing values may be due to faulty recording, signal transmission to the central unit or storage.

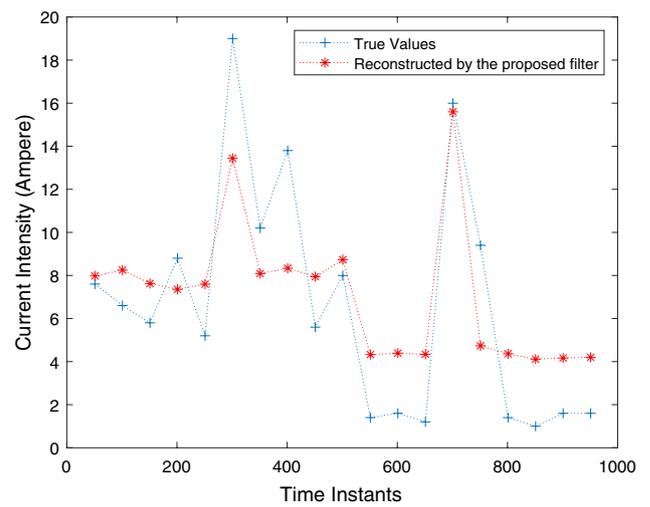
Assume the sensor readings are recorded in the vector  $\mathbf{y} = [\mathbf{y}_M; \mathbf{y}_U]$ , where  $\mathbf{y}_M$  is the portion of the signal that is correctly recorded and known while  $\mathbf{y}_U$  includes missing values. Following the procedure explained in [71] but with our designed filter in (18), the signal is estimated as follows

$$\mathbf{y}^* = \left( \begin{bmatrix} \mathbf{I}_M & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \sigma^2 \mathbf{L}\mathbf{L} \right)^{-1} \begin{bmatrix} \mathbf{y}_M \\ \mathbf{0} \end{bmatrix} \quad (57)$$

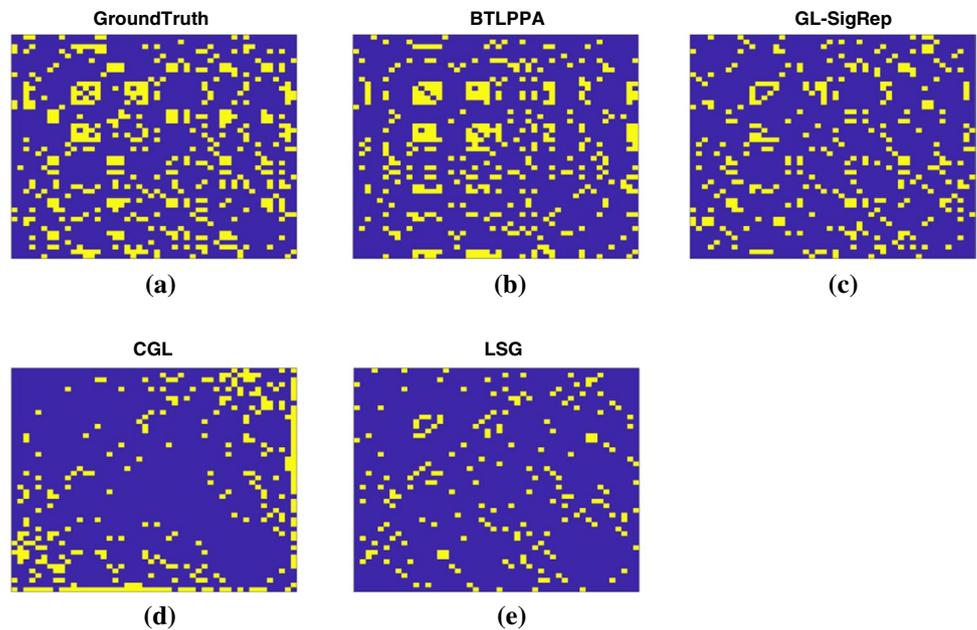
where  $M$  is the cardinality of  $\mathbf{y}_M$ .

To apply the proposed methods on a relevant IIoT data, the ‘‘Household Power Consumption’’ data set has been downloaded from [72]. This data set is an individual household electric consumption data set collected via submeters in 5 years (2006–2010), while we only use a subset of it here. First  $K = 100,000$  time samples have been saved from the first block of data and then it is downsampled to 1000 samples. The data set contains 7 measurements, including the active power, reactive power, voltage, intensity and three energy submeter recordings from different places in a house. The intensity sensor readings are chosen to be tested here and then all multipliers of 50 in its time instances were deleted to model missing values, i.e.  $y_k \text{ for } k \in \{50, 150, \dots, 950, 1000\}$  are the missing values. Now, this dataset is given to the Algorithm 2 to find the underlying graph Laplacian matrix along with the estimated noise variance. The missing values are estimated by applying (57). Figure 6 shows the result of signal inpainting via our proposed filter. For a better illustration, only the missing values and the estimated ones have been shown.

**Fig. 6** Signal inpainting for an IIoT application. The true values for 20 missing samples are compared with the estimated ones using the proposed graph filter. The original data is the time series of household power consumption readings [72]



**Fig. 7** Visual comparisons among **a** the Ground-truth adjacency matrix, and the adjacency matrix learned by **b** BTL-PPA, **c** GL-SigRep, **d** CGL, and **e** LSG. Here, blue pixel shows that there is a connection between two states (an edge between two nodes in the graph)



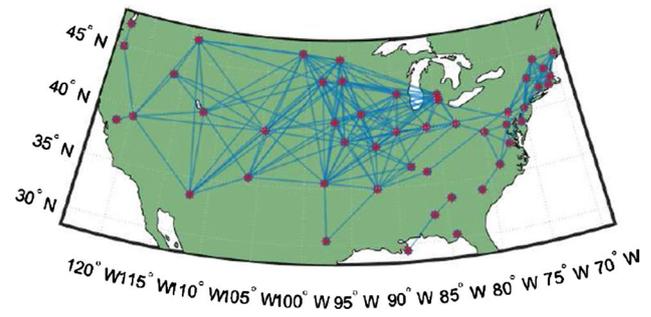
The main reason for the good performance is that the designed filter considers each dimension of the multi-variate signal as an entity of the entire network which is related to others based on a structure and hence it tries to use this knowledge in the filtering process. In other words, based on the measured signals, the algorithm learns the structure of the sensor readings first and then tries to use this data structure to interpolate the missing values.

### 6.3 Temperature data

In this experiment, the daily temperatures of  $N = 48$  states of the USA mainland are stored for the years 2011 to 2014, i.e.  $K = 1461$  [73]. Here, the graph signals are average daily temperatures and the underlying topology describes the temperature relation between states. We do not have access to the ground-truth topology, but a geographical based graph may be proposed to compare the results. A graph is considered where the nodes are the states and an edge weight between two states is computed by the Gaussian RBF of the physical distance.

First, the temperature data of 2011 is used to learn the underlying topology. Figure 7 compares different graph learning algorithms with respect to their capability to capture the underlying connections. It is shown that the proposed BTL-PPA can detect most of the edges. Figure 8 shows the topology learned by our proposed method over the real map of the United States. It can be inferred that a state weather influences that of the neighboring states, which is also

**Fig. 8** The learned graph topology via BTL-PPA algorithm from real temperature data in 2011: the background map is the USA mainland



**Table 2** Performance measures for learned topology from average temperature for the US mainland in 2011

	NMSD	NMI	F-measure
BTL-PPA	$10^{-4}$	0.78	0.79
GL-SigRep [51]	$10^{-4}$	0.72	0.69
CGL [22]	$10^{-3}$	0.44	0.36
LSG [20]	$10^{-4}$	0.44	0.35

**Table 3** Performance of different algorithms for learning the graph from the test data set, when compared to the one learned from the training data

	NMSD	NMI	F-measure
BTL-PPA	$10^{-8}$	0.93	0.93
GL-SigRep [51]	$10^{-5}$	0.90	0.88
CGL [22]	$10^{-4}$	0.66	0.62
LSG [20]	$10^{-5}$	0.87	0.83

corroborated by physics of temperature propagation. There are more edges in the far right side of the map due to the higher density of nodes, representing east coast regions. Then, from right to left of the map, there is a bit discontinuity among edges which can be representing the blockage of weather propagation by Appalachian mountains. The smaller number of connections in the middle of the figure can also be due to the Rocky mountain chain, also mentioned in [6]. At the same way, a few edges in the right side of the California corroborates the effect of Sierra Nevada mountain range. The numerical results to compare different algorithms' performances are given in Table 2.

In the second experiment with real temperature data, the data set is divided into two groups of data, the training set and the testing set. The first half of the data, i.e. the averaged daily temperature of 2011 and 2012  $k = 1, \dots, 731$ ,<sup>4</sup> is utilized as the training data to learn the underlying topology. Then we consider this topology as the ground-truth graph and the remaining data, i.e. the data from the year 2013 and 2014, is used to estimate the topology and compare to the ground-truth to check the consistency of the learning algorithms. In other words, a cross validation procedure is done to verify how the learned topology from given training data is different from the one learned by the test data.

Table 3 shows the results for the cross validation scenario. The proposed BTL-PPA algorithm has the best consistency results among all algorithms with respect to all performance measure.

## 7 Conclusion

Many information networks involve multiple interacting entities for which finding the topology connecting these entities connections is somehow important for real-world applications. In the graph topology inference framework, we tried to estimate the structure of the underlying data from multi-variate measurements. In other words, we investigated an algorithm to explore the link between the signal model and the graph topology. In this paper, a factor analysis model was used for signal representation in the graph domain and a Bayesian inference method was applied to learn the Laplacian matrix (which can

<sup>4</sup> The year 2012 was a leap year.

uniquely represent the graph topology) and to estimate the noise variance at the same time. To formulate the problem, we used a Bayesian framework and proposed a minimum mean square estimation approach to denoise the measurements. Finally, a convex optimization problem over the graph Laplacian matrix was proposed and solved via a proximal point method to estimate the topology from a denoised version of graph signals. The experimental results corroborate the performance of the proposed algorithm for a wide range of sensor networks and IoT applications.

**Acknowledgements** 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-1744129, CNS-1702555, DMS-1736417 and ECCS-1824710.

**Authors' contributions** MRM: proposed the main idea of the paper, worked on analytical perspective and computations, derived the both algorithms in the paper, collected the data and had done simulations. MH: helped in writing and edited the paper. KS: Helped partially in some maths of the paper. XD: Helped in the main idea and adding the first three figures. RSB: was the main reviewer and editor of the paper. HVP: reviewed the paper and gave general comments. All authors reviewed the manuscript. All authors read and approved the final manuscript.

**Declarations**

**Competing interests** The authors declare no competing interests.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

**Appendices**

**Appendix 1: Notes on the derivation of (21)**

$$\begin{aligned} \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\theta}} [\log p(\mathbf{x} | \mathbf{h})] &= -\frac{1}{2} \log 2\pi - \frac{NK}{2} \log \sigma - \frac{1}{2\sigma} \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\theta}} [\|\mathbf{x} - \mathbf{B}\mathbf{h}\|_2^2] \\ &\stackrel{(a)}{\propto} -\frac{NK}{2} \log \sigma - \frac{1}{2\sigma} \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\theta}} [\|\mathbf{e}\|_2^2] \\ &= -\frac{NK}{2} \log \sigma - \frac{1}{2\sigma} \text{Tr}(\mathbf{C}_\epsilon), \end{aligned} \tag{58}$$

where <sup>(a)</sup> is due to the fact that the first term is constant and will be ignored in the optimization.

$$\begin{aligned} \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\theta}} [\log p(\mathbf{h})] &= \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\theta}} [\log |\mathbf{C}_0| - \mathbf{h}^T \mathbf{C}_0 \mathbf{h}] \\ &\propto K \cdot \log |\mathbf{A}| - \mathbb{E}_{\mathbf{h}|\mathbf{x};\hat{\theta}} [\mathbf{y}^T (\mathbf{I} \times \mathbf{L}) \mathbf{y}] \\ &= K \cdot \log |\mathbf{L}| - \text{Tr}(\hat{\mathbf{Y}}^T \mathbf{L} \hat{\mathbf{Y}}) = K \cdot \log |\mathbf{L}| - K \cdot \text{Tr}(\mathbf{S}\mathbf{L}). \end{aligned} \tag{59}$$

**Appendix 2: Derivation of  $P_\eta(\mathbf{L}; \mathbf{v})$ , i.e. (49)**

$$\begin{aligned} \mathbf{P} &= \mathbf{v}^T \mathbf{a} + \frac{1}{2\eta} \|\mathbf{L}\|_F^2 - \frac{1}{2\gamma'} \left\| \mathbf{T}_\eta(\mathbf{L}; \mathbf{v}) \right\|_F^2 \\ &\quad + \frac{1}{2\gamma'} \left\| \phi_{\gamma'}^-(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})) \right\|_F^2 - \log \det \left( \phi_{\gamma'}^+(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})) \right) \\ &= \mathbf{v}^T \mathbf{a} + \frac{1}{2\eta} \|\mathbf{L}\|_F^2 - \log \det \left( \phi_{\gamma'}^+(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})) \right) \\ &\quad + \frac{1}{2\gamma'} \left\| \phi_{\gamma'}^-(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})) \right\|_F^2 - \frac{1}{2\gamma'} \left\| \mathbf{T}_\eta(\mathbf{L}; \mathbf{v}) \right\|_F^2 \\ &\stackrel{(a)}{=} \mathbf{v}^T \mathbf{a} + \frac{1}{2\eta} \|\mathbf{L}\|_F^2 - \log \det \left( \phi_{\gamma'}^+(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})) \right) - \frac{1}{2\gamma'} \left\| \phi_{\gamma'}^+(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v})) \right\|_F^2 + N, \end{aligned} \tag{60}$$

where  $\underline{(a)}$  follows from Lemma 1.

### Appendix 3: Proof of Lemma 2

To simplify the notation, we define  $\Phi^+ := \phi_{\gamma'}^+(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v}))$ ,  $\Phi^- := \phi_{\gamma'}^-(\mathbf{T}_\eta(\mathbf{L}; \mathbf{v}))$  and  $(\Phi^+)' := \frac{\partial \phi_{\gamma'}^+}{\partial \mathbf{T}_\eta}$ ;

$$\begin{aligned} \nabla_{\mathbf{v}} \mathbf{P}_\eta(\mathbf{L}; \mathbf{v}) &= \mathbf{a} - \mathcal{B}\left((\Phi^+)' \Phi^+\right) - \gamma' \mathcal{B}\left((\Phi^+)' (\Phi^+)^{-1}\right) \\ &= \mathbf{a} - \mathcal{B}\left((\Phi^+)' [\Phi^+ + \Phi^-]\right) = \mathbf{a} - \mathcal{B}(\Phi^+), \end{aligned} \quad (61)$$

where the last equality follows from Lemma 1, part (3).

### Appendix 4: Proof of Lemma 4

Since the domain of  $\phi_\gamma^+(x)$  is  $\mathcal{R}$ , the function is differentiable everywhere, and the derivative is bounded. Thus the Lipschitz constant is

$$\sup_x \left| \frac{\partial \phi_\gamma^+(x)}{\partial x} \right| = \sup_x \left| \frac{x}{\sqrt{x^2 + 4\gamma}} + 0.5 \right| = 1.5, \quad (62)$$

where  $x \in \mathbb{R}$  and for all  $\gamma > 0$ .

### Appendix 5: Proof of Lemma 5

We know that when  $\theta_{ii}$  is an eigenvalue of the matrix  $\mathbf{Z}$ ,  $\theta_{ii}^2$  will be an eigenvalue of  $\mathbf{Z}^2$ . Thus,

$$\|\mathbf{Z}\|_F^2 = \text{Tr}(\mathbf{Z}\mathbf{Z}^T) = \text{Tr}(\mathbf{Z}^2) = \sum_i [\theta_{Z^2}]_{ii} = \sum_i [\theta_Z]_{ii}^2 = \|\theta_Z\|_F^2, \quad (63)$$

### Appendix 6: Proof of Theorem 1

$$\begin{aligned} \|\mathbf{L}^{(t+1)} - \mathbf{L}^{(t_*)}\|_F^2 &= \left\| \phi_{\gamma'}^+(\mathbf{T}_\eta^{(t)}) - \phi_{\gamma'}^+(\mathbf{T}_\eta^{(t_*)}) \right\|_F^2 \\ &\leq l_c \|\mathbf{T}_\eta^{(t)} - \mathbf{T}_\eta^{(t_*)}\|_F^2 \\ &= \frac{l_c}{(1 + 2\eta\sigma)^2} \left\| (\mathbf{L}^{(t)} - \mathbf{L}^{(t_*)}) + \eta \cdot \mathcal{B}_a(\mathbf{v}^{(t)} - \mathbf{v}^{(t_*)}) \right\|_F^2 \\ &\leq \frac{l_c}{(1 + 2\eta\sigma)^2} \left( \|\mathbf{L}^{(t)} - \mathbf{L}^{(t_*)}\|_F^2 + \eta^2 N^2 \|\mathbf{v}^{(t)} - \mathbf{v}^{(t_*)}\|_2^2 \right), \end{aligned} \quad (64)$$

To find the optimum dual variable in the second step of the Algorithm 2, if we set the stopping criterion as (56), then (64) can be simplified to

$$\|\mathbf{L}^{(t+1)} - \mathbf{L}^{(t_*)}\|_F^2 \leq \frac{2l_c}{(1 + 2\eta\sigma)^2} \|\mathbf{L}^{(t)} - \mathbf{L}^{(t_*)}\|_F^2, \quad (65)$$

where  $\frac{2l_c}{(1+2\eta\sigma)^2} < 1$  is required for the convergence of  $\mathbf{L}$  which can be guaranteed by using Corollary 2 and setting  $\eta \geq \frac{3\sqrt{2}-2}{4\sigma}$ .

## References

- Li S, Da Xu L, Zhao S. The internet of things: a survey. *Inf Syst Front*. 2015;17(2):243.

2. Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P. The emerging field of signal processing on graphs: extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Process Mag.* 2013;30(3):83. <https://doi.org/10.1109/MSP.2012.2235192>.
3. Ortega A, Frossard P, Kovačević J, Moura JM, Vandergheynst P. Graph signal processing: overview, challenges, and applications. *Proc IEEE.* 2018;106(5):808.
4. Friston KJ. Functional and effective connectivity in neuroimaging: a synthesis. *Human Brain Map.* 1994;2(1–2):56.
5. Goebel R, Roebroeck A, Kim DS, Formisano E. Investigating directed cortical interactions in time-resolved fMRI data using vector autoregressive modeling and Granger causality mapping. *Magn Resonance Imaging.* 2003;21(10):1251.
6. Mei J, Moura JM. Signal processing on graphs: causal modeling of unstructured data. *IEEE Trans Signal Process.* 2017;65(8):2077.
7. Bolstad A, Veen BDV, Nowak R. Causal network inference via group sparse regularization. *IEEE Trans Signal Process.* 2011;59(6):2628. <https://doi.org/10.1109/TSP.2011.2129515>.
8. Shen Y, Baingana B, Giannakis GB. Kernel-based structural equation models for topology identification of directed networks. *IEEE Trans Signal Process.* 2017;65(10):2503. <https://doi.org/10.1109/TSP.2017.2664039>.
9. Baccalá LA, Sameshima K. Partial directed coherence: a new concept in neural structure determination. *Biol Cybern.* 2001;84(6):463.
10. Songsiri J, Vandenberghe L. Topology selection in graphical models of autoregressive processes. *J Mach Learn Res.* 2010;11(Oct):2671.
11. Shen Y, Baingana B, Giannakis GB. Nonlinear structural vector autoregressive models for inferring effective brain network connectivity; 2016. arXiv preprint. [arXiv:1610.06551](https://arxiv.org/abs/1610.06551).
12. Baingana B, Giannakis GB. Tracking switched dynamic network topologies from information cascades. *IEEE Trans Signal Process.* 2016;65(4):985.
13. Ramezani-Mayiami M, Beferull-Lozano B. Graph recursive least squares filter for topology inference in causal data processes In: Proceedings of the IEEE 7th international workshop on computational advances in multi-sensor adaptive processing (CAMSAP); 2017. p. 1–5.
14. Traganitis PA, Shen Y, Giannakis GB. Network topology inference via elastic net structural equation models In: Proceedings of the IEEE 25th European signal processing conference (EUSIPCO) (IEEE); 2017. p. 146–50.
15. Dempster AP. Covariance selection. *Biometrics.* 1972;25:157–75.
16. Banerjee O, Ghaoui LE, d'Aspremont A. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *J Mach Learn Res.* 2008;9(Mar):485.
17. Friedman J, Hastie T, Tibshirani R. Sparse inverse covariance estimation with the graphical LASSO. *Biostatistics.* 2008;9(3):432.
18. Yuan M, Lin Y. Model selection and estimation in the Gaussian graphical model. *Biometrika.* 2007;94(1):19.
19. Scheinberg K, Rish I. Learning sparse Gaussian Markov networks using a greedy coordinate ascent approach. In: Joint European conference on machine learning and knowledge discovery in databases. Berlin: Springer; 2010. p. 196–212.
20. Lake B, Tenenbaum J. Discovering structure by learning sparse graph. In: Proceedings of the 32nd annual meeting of the cognitive science society; 2010. p. 6160–73.
21. Kalofolias V. How to learn a graph from smooth signals. In: Proceedings of the 19th international conference of artificial intelligence and statistics, AISTATS, Cadiz; 2016. p. 920–9.
22. Egilmez HE, Pavez E, Ortega A. Graph learning from data under Laplacian and structural constraints. *IEEE J Selected Topics Signal Process.* 2017;11(6):825.
23. Segarra S, Marques AG, Mateos G, Ribeiro A. Network topology inference from spectral templates. *IEEE Trans Signal Inf Process Over Netw.* 2017;3(3):467. <https://doi.org/10.1109/TSIPN.2017.2731051>.
24. Pasdeloup B, Gripon V, Mercier G, Pastor D, Rabbat MG. Characterization and inference of graph diffusion processes from observations of stationary signals. *IEEE Trans Signal Inf Process Over Netw.* 2018;4(3):481. <https://doi.org/10.1109/TSIPN.2017.2742940>.
25. Shafipour R, Segarra S, Marques AG, Mateos G. Network topology inference from non-stationary graph signals In: Proceedings of the IEEE international conference on acoustics, speech and signal processing (ICASSP). New York: IEEE; 2017. p. 5870–4.
26. Thanou D, Dong X, Kressner D, Frossard P. Learning heat diffusion graphs. *IEEE Trans Signal Inf Process Over Netw.* 2017;3(3):484. <https://doi.org/10.1109/TSIPN.2017.2731164>.
27. Egilmez HE, Pavez E, Ortega A. Graph learning from filtered signals: Graph system and diffusion kernel identification. *IEEE Trans Signal Inf Process Over Netw.* 2018;5(2):360–74.
28. Ma H, Yang H, Lyu MR, King I. Mining social networks using heat diffusion processes for marketing candidates selection In: Proceedings of the 17th ACM conference on Information and knowledge management (ACM); 2008. p. 233–42.
29. Yankelevsky Y, Elad M. Dual graph regularized dictionary learning. *IEEE Trans Signal Inf Process Over Netw.* 2016;2(4):611.
30. Ramezani-Mayiami M, Skretting K. Topology inference and signal representation using dictionary learning In: Proceedings of the IEEE 27th European signal processing conference (EUSIPCO); 2019.
31. Sardellitti S, Barbarossa S, Di Lorenzo P. Graph topology inference based on sparsifying transform learning. *IEEE Trans Signal Process.* 2019;67(7):1712.
32. Giannakis GB, Shen Y, Karanikolas GV. Topology identification and learning over graphs: accounting for nonlinearities and dynamics. *Proc IEEE.* 2018;106(5):787.
33. Varma R, Chen S, Kovačević J. Graph topology recovery for regular and irregular graphs. In: 2017 IEEE 7th international workshop on computational advances in multi-sensor adaptive processing (CAMSAP). 2017. p. 1–5. <https://doi.org/10.1109/CAMSAP.2017.8313202>.
34. Marinakis D, Dudek G. A practical algorithm for network topology inference. In: Proceedings of the IEEE International Conference on Robotics and Automation, 2006. ICRA 2006. New York: IEEE. 2006. p. 3108–15.
35. Pavez E, Egilmez HE, Ortega A. Learning graphs with monotone topology properties and multiple connected components. *IEEE Trans Signal Process.* 2018;66(9):2399.
36. Satya JP, Bhatt N, Pasumarthy R, Rajeswaran A. Identifying topology of power distribution networks based on smart meter data; 2016. arXiv preprint. [arXiv:1609.02678](https://arxiv.org/abs/1609.02678).
37. Pavez E, Ortega A. Generalized Laplacian precision matrix estimation for graph signal processing In: Proceedings of the IEEE international conference on acoustics, speech and signal processing (ICASSP). New York: IEEE. 2016. p. 6350–4.

38. Kalofolias V, Loukas A, Thanou D, Frossard P. Learning time varying graphs. In: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). New York: IEEE. 2017. p. 2826–30.
39. Dong X, Thanou D, Rabbat M, Frossard P. Learning graphs from data: a signal representation perspective. *IEEE Signal Process Mag.* 2019;36(3):44. <https://doi.org/10.1109/MSP.2018.2887284>.
40. Mateos G, Segarra S, Marques AG, Ribeiro A. Connecting the dots: identifying network structure via graph signal processing. *IEEE Signal Process Mag.* 2019;36(3):16. <https://doi.org/10.1109/MSP.2018.2890143>.
41. Kumar S, Ying J, de Miranda Cardoso JV, Palomar D. Structured graph learning via laplacian spectral constraints. In: *Advances in neural information processing systems*. 2019. p. 11651–63.
42. Wentzell PD. Measurement errors in multivariate chemical data. *J Braz Chem Soc.* 2014;25(2):183.
43. Azadkia M. Adaptive estimation of noise variance and matrix estimation via usvt algorithm. 2018. arXiv preprint. [arXiv:1801.10015](https://arxiv.org/abs/1801.10015)
44. Zhang S, Ding Z, Cui S. Introducing hypergraph signal processing: theoretical foundation and practical applications. *IEEE Internet Things J.* 2019;7(1):639.
45. Liu Y, Dillon T, Yu W, Rahayu W, Mostafa F. Noise removal in the presence of significant anomalies for Industrial IoT sensor data in manufacturing. *IEEE Internet Things J.* 2020;7:7084–96.
46. Ramezani-Mayiami M, Skretting K. Robust Graph Topology Learning and Application in Stock Market Inference In: 2019 IEEE international conference on signal and image processing applications (ICSIPA). New York: IEEE. 2019. p. 240–4.
47. Black F. Noise. *J Fin.* 1986;41(3):528.
48. Chen S, Sandryhaila A, Moura JM, Kovacevic J. Signal denoising on graphs via graph filtering. In: 2014 IEEE global conference on signal and information processing (GlobalSIP). New York: IEEE. 2014. p. 872–6.
49. Galatsanos NP, Katsaggelos AK. Methods for choosing the regularization parameter and estimating the noise variance in image restoration and their relation. *IEEE Trans Image Process.* 1992;1(3):322.
50. Chepuri SP, Liu S, Leus G, Hero AO. Learning sparse graphs under smoothness prior. In: *Proceedings of the IEEE international conference on acoustics, speech and signal processing (ICASSP)*. New York: IEEE. 2017. p. 6508–12.
51. Dong X, Thanou D, Frossard P, Vandergheynst P. Learning Laplacian matrix in smooth graph signal representations. *IEEE Trans Signal Process.* 2016;64(23):6160. <https://doi.org/10.1109/TSP.2016.2602809>.
52. Ramezani-Mayiami M, Hajimirsadeghi M, Skretting K, Blum RS, Vincent Poor H. Graph topology learning and signal recovery via Bayesian inference. In: *Proceedings of the IEEE data science workshop (DSW)*. 2019. p. 52–6. <https://doi.org/10.1109/DSW.2019.8755601>.
53. Perraudin N, Paratte J, Shuman D, Martin L, Kalofolias V, Vandergheynst P, Hammond DK. GSPBOX: a toolbox for signal processing on graphs. *ArXiv e-prints* 2014.
54. Basilevsky AT. *Statistical factor analysis and related methods: theory and applications*, vol. 418. Hoboken: John Wiley & Sons; 2009.
55. Kay SM. *Fundamentals of statistical signal processing*. Prentice: Prentice Hall PTR; 1993.
56. Bai Z, Golub GH. Bounds for the trace of the inverse and the determinant of symmetric positive definite matrices. *Ann Numer Math.* 1996;4:29.
57. Löfberg J. YALMIP: a toolbox for modeling and optimization in MATLAB. In: *Proceedings of the IEEE international conference on robotics and automation*. 2004. p. 284–9.
58. Boyd S, Vandenberghe L. *Convex optimization*. Cambridge: Cambridge University Press; 2004.
59. Parikh N, Boyd S, et al. Proximal algorithms. *Foundations Trends<sup>®</sup> Opt.* 2014;1(3):127.
60. Bauschke H, Combettes P, Noll D. Joint minimization with alternating Bregman proximity operators. *Pacific J Opt.* 2006. <https://hal.archives-ouvertes.fr/hal-01868791/>. [https://scholar.google.com/scholar?hl=en&as\\_sdt=0%2C5&q=Joint+minimization+with+alternating+Bregman+proximity+operators&btnG=](https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=Joint+minimization+with+alternating+Bregman+proximity+operators&btnG=)
61. Yosida K. *Functional analysis*. Berlin: Springer; 1964.
62. Moreau JJ. Proximité et dualité dans un espace hilbertien. *Bull Soc Math France.* 1965;93(2):273.
63. Neumann Jv. Zur Theorie der Gesellschaftsspiele. *Mathematische Annalen.* 1928;100(1):295. <https://doi.org/10.1007/BF01448847>.
64. Von Neumann J. Über ein ökonomisches gleichungssystem und eine verallgemeinerung des browerschen fixpunktsatzes. *Erge Math Kolloq.* 1937;8:73–83.
65. Wang C, Sun D, Toh KC. Solving log-determinant optimization problems by a Newton-CG primal proximal point algorithm. *SIAM J Opt.* 2010;20(6):2994.
66. Roberts AW, Varberg DE. Another proof that convex functions are locally Lipschitz. *Am Math Monthly.* 1974;81(9):1014.
67. Hu C, Cheng L, Sepulcre J, Johnson KA, Fakhri GE, Lu YM, Li Q. A spectral graph regression model for learning brain connectivity of Alzheimer's disease. *PLoS ONE.* 2015;10(5):e0128136.
68. Cover TM, Thomas JA. *Elements of information theory*. Hoboken: John Wiley & Sons; 2012.
69. Goutte C, Gaussier E. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In: *European conference on information retrieval*. Berlin: Springer. 2005. p. 345–59.
70. Manning C, Raghavan P, Schütze H. *Introduction to information retrieval*. Nat Lang Eng. 2010;16(1):100.
71. Chen S, Sandryhaila A, Lederman G, Wang Z, Moura JM, Rizzo P, Bielak J, Garrett JH, Kovačević J. Signal inpainting on graphs via total variation minimization. In: 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). New York: IEEE. 2014. p. 8267–71.
72. Ortiz J. Household power consumption 2016; Household Power Consumption, Individual household electric power consumption dataset collected via submeters placed in 3 distinct areas of a home, <https://data.world/databeats/household-power-consumption>.
73. National climatic data center. <ftp://ftp.ncdc.noaa.gov/pub/data/gso/>