Learning Laplacian Matrix in Smooth Graph Signal Representations

Xiaowen Dong, Dorina Thanou, Pascal Frossard, and Pierre Vandergheynst

Abstract—The construction of a meaningful graph plays a crucial role in the success of many graph-based representations and algorithms for handling structured data, especially in the emerging field of graph signal processing. However, a meaningful graph is not always readily available from the data, nor easy to define depending on the application domain. In particular, it is often desirable in graph signal processing applications that a graph is chosen such that the data admit certain regularity or smoothness on the graph. In this paper, we address the problem of learning graph Laplacians, which is equivalent to learning graph topologies, such that the input data form graph signals with smooth variations on the resulting topology. To this end, we adopt a factor analysis model for the graph signals and impose a Gaussian probabilistic prior on the latent variables that control these signals. We show that the Gaussian prior leads to an efficient representation that favors the smoothness property of the graph signals. We then propose an algorithm for learning graphs that enforces such property and is based on minimizing the variations of the signals on the learned graph. Experiments on both synthetic and real world data demonstrate that the proposed graph learning framework can efficiently infer meaningful graph topologies from signal observations under the smoothness prior.

Index Terms—Laplacian matrix learning, graph signal processing, representation theory, factor analysis, Gaussian prior.

I. INTRODUCTION

MODERN data processing tasks often manipulate structured data, where signal values are defined on the vertex set of a weighted and undirected graph. We refer to such data as graph signals, where the vertices of the graph represent the entities and the edge weights reflect the pairwise relationships between these entities. The signal values associated with the vertices carry the information of interest in observations or physical measurements. Numerous examples can be found in real world applications, such as temperatures within a geographical area, transportation capacities at hubs in a transportation network, or human behaviors in a social network. Such data representations have led to the emerging

field of graph signal processing [1], [2], which studies the representation, approximation and processing of such structured signals. Currently, most of the research effort in this field has been devoted to the analysis and processing of the signals, which are defined on a graph that is a priori known or naturally chosen from the application domain, e.g., geographical or social friendship graphs. However, these natural choices of graphs may not necessarily describe well the intrinsic relationships between the entities in the data. Furthermore, a natural graph might not be easy to define at all in some applications. When a good graph is not readily available, it is desirable to learn the graph topology from the observed data such that it captures well the intrinsic relationships between the entities, and permits effective data processing and analysis. This is exactly the main objective of the present paper.

Generally speaking, learning graphs from data samples is an ill-posed problem, and there might be many solutions to associate a structure to the data. We therefore need to define meaningful models for computing the topology such that the relationships between the signals and the graph topology satisfy these pre-defined models. This allows us to define meaningful criteria to describe or estimate structures between the data samples. In this work, we consider the large class of signals that are smooth on a graph structure. A graph signal is smooth if the signal values associated with the two end vertices of edges with large weights in the graph tend to be similar. Such property is often observed in real world graph signals. For instance, consider a geographical graph where the vertices represent different locations and the edges capture their physical proximities. If we define the signal values as temperature records observed in these locations, then the values at close-by vertices tend to be similar in regions without significant terrain variations. Another example would be a social network graph where the vertices and edges respectively represent people and the friendships between them, and signals defined as personal interests. It is often the case that friends, represented by connected vertices, share common interests. In both examples, data have smooth graph signal representations as long as the graphs are appropriately chosen. Smooth signal models on graphs have been used in many problems involving regularization on graphs [3], with applications to signal denoising [1], classification [4] or multi-view clustering [5] to name a few.

To further illustrate the interplay between the graph and the data, we consider a signal given by a set of unordered scalar values, which can potentially be defined on three different graphs G_1, G_2 and G_3, leading to three graph signals shown in Fig. 1. Without any assumption on the properties of the
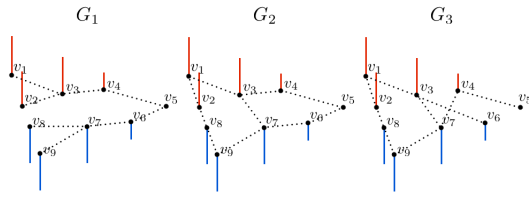
Fig. 1. A given signal can potentially live on different graphs but only one leads to a smooth graph signal (figure inspired by [1]). The red bars pointing upwards and the blue bars pointing downwards represent positive and negative signal values, respectively. The length of the bars reflects the magnitude of the values. In this illustration, while all the graphs are valid a priori, only choosing graph G_1 would favor the smoothness property of the resulting graph signal.

graph signal, the three candidate graphs are all valid choices. However, if we assume that the signal is smooth on the underlying graph, then G_1 is obviously a more reasonable choice than G_2 and G_3. Our objective is exactly to learn a graph similar to G_1 when the signal is expected to be smooth on an "unknown" graph topology. Formally speaking, we consider in this paper the following problem: *Given a set of potentially noisy data $X = \{x_i\}_{i=1}^p$ ($x_i \in \mathbb{R}^n$), we would like to infer an optimal weighted and undirected graph G of n vertices, namely, its edges and the associated weights, such that the elements of X form smooth signals on G.*

In this paper, we build on our previous work in [6] and propose to define the relationships between signals and graphs by revisiting the representation learning theory [7] studied in the traditional signal processing setting. Specifically, we consider a factor analysis model used in the theory of representation learning, where we assume that the observed signals are controlled by a set of unobserved latent variables with some given probabilistic prior. We generalize this model to graph signals, where we assume that the observed graph signals can be represented in terms of a set of latent variables. The difference from the traditional analysis model is that the transformation from the latent variables to the observed signals involves information about the topology of the graph. As a result, we can define joint properties (or a joint model) between the signals and the graph, such that the signal representations are consistent with given statistical priors on the latent variables. Specifically, by imposing a Gaussian prior on the latent variables in our generalized factor analysis model, we obtain a Principal Component Analysis (PCA)-like representation for the graph signals, which turns out to be a smooth signal representation on graphs. The relationship between the Gaussian latent variables and the signal observations is based on the graph Laplacian matrix, which uniquely characterizes the graph topology and has been widely connected in the graph signal processing literature with the notion of smoothness on the graph. Hence, learning a graph from data becomes equivalent to learning its graph Laplacian matrix, which is central to many key concepts in graph signal processing.

Building on the above model, we design a new algorithm for learning a valid graph Laplacian operator from data samples, such that the graph signal representation is smooth and consistent with the Gaussian prior on the latent variables.

Specifically, given potentially noisy signal observations, our algorithm iterates between the learning of a smoothed version of the original signal measurements and the learning of a graph topology such that the variations of the smooth signals on the learned graph are minimized upon convergence. We test our graph learning algorithm on several synthetic and real world experiments, where we show that it is able to efficiently infer the topology of the groundtruth graphs. It further achieves competitive performance with a state-of-the-art structural learning algorithm, which is closely related to the idea of sparse inverse covariance estimation for Gaussian graphical models [8]. More importantly, our results demonstrate the meaningfulness and efficiency of a new learning framework from the graph signal processing perspective.

Our novel approach is one of the first rigorous frameworks in graph signal processing that connects the learning of the graph topology to the representation model of the graph signals and their properties. By extending the classical and generic factor analysis model to graph settings, our framework provides new insights into the understanding of the interactions between signals and graphs. We believe that our new framework can benefit numerous emerging real world applications, such as the analysis of transportation, biomedical, and social networks, where it is critical to infer hidden relationships between data entities.

II. RELATED WORK

Graph signal processing, as an emerging research field, has been attracting an increasing amount of interest from the signal processing community. Most of the research effort so far has been devoted to the efficient representation and processing of signals defined on a given graph. Representative works include studies on the generalizations of the Fourier transform [9], [10], the wavelet transform [11], [12], [13], [14], [15], [16], dictionary learning methods [17], [18], and time-frequency analysis [19], [20] on graphs, with applications to image processing [21], brain network classification [22], mobility inference [23], and community detection [24] to name a few.

In most of these works, the irregular domain of the graph is captured through the graph Laplacian operator, whose importance can be summarized as follows. First, from a graph signal processing perspective, a Laplacian matrix represents a valid graph topology G (i.e., with non-negative edge weights and without self-loops as usually considered in the literature), and supports the definition of graph signals. Furthermore, the Laplacian matrix enables the generalization of the notion of frequency and Fourier transform for graph signals [13], which is the building block of many techniques that have been recently developed in the field of graph signal processing. Therefore, the Laplacian matrix is directly related to the processing of graph signals. Second, the graph Laplacian is an operator central to the definition of various kernels on graphs via its spectral decomposition. More specifically, Smola et al. has shown in [3] that graph Laplacian can be used to define various kernels related to regularization, diffusion, and random walk processes associated with graphs. As a result, the Laplacian matrix permits to process real world

graph signals that represent complex behaviors associated with kernels on graphs. Third, the graph Laplacian, under some conditions, converges to the intrinsic Laplace-Beltrami operator on a Riemannian manifold [25], [26], [27], therefore it may well capture the underlying manifold structure of the observed signals hence benefit the processing tasks. Finally, from an application point of view, Zhang et al. has shown that, when a signal follows a Gaussian Markov Random Field (GMRF) model with the precision matrix (inverse covariance matrix) defined as the graph Laplacian, the eigenvectors of the Laplacian optimally decorrelate the signal and can thus be used for efficient compression [28]. These results show that the knowledge of a valid Laplacian matrix is indeed beneficial to graph-based signal and image processing in real world applications.

While many algorithms are built on the graph Laplacian, they mainly assume that the later is either given a priori, or chosen naturally from the application domain. Relatively less research effort has been devoted to the construction or learning of the graph topologies. A regression framework is proposed in [29] to learn a graph Laplacian matrix based on a fitness metric between the signals and the graph that essentially evaluates the smoothness of the signals on the graph. A similar fitness metric has also been used in [30] to learn a valid graph topology (the adjacency matrix). The main difference between these two works and the method proposed in this paper is that, instead of using a fitness metric, our work relates the graph learning process and the properties of the graph signals in a statistical manner that helps in understanding the data properties. Moreover, the factor analysis model adopted in our new framework implies that the graph signals are generated by a simple linear statistical model, which can be revealed through the learning process.

Two other graph learning works have been described in [31] and [32], where the authors have proposed to use, respectively, correlations between wavelet coefficients of the time series of brain signals, and Principal Component Analysis applied to correlation matrices, to estimate functional connectivities of distinct brain regions. Although these can be considered as methods that learn an activity graph between the brain regions, they are essentially similar to traditional approaches that are based on pairwise correlation analysis rather than considering the global behavior of the signals on the graph.

The problem of constructing a meaningful graph has also been studied implicitly in [33], [34], [35] in the context of multiple kernel learning. The first two works aim at finding an optimal convex combination of Laplacians of some initial graphs that are constructed a priori using some known features. One main difference with our approach is that, in these two papers the basic graphs are constructed using distance metrics and additional features, and a label function is learned by enforcing its smoothness over all the basic graphs through a convex combination. On the other hand, in our work, we do not assume any prior knowledge or additional information to construct such basic graphs and the optimal graph is learned by only using a set of signal observations. The authors in [35] learn an optimal transformation of the spectrum of a predefined graph Laplacian while fixing its eigenvector space

(such a graph is again computed using additional features). In comparison, our work aims at learning both the spectrum and eigenspace (hence the Laplacian matrix) from the mere signal measurements.

In the meantime, there is a large amount of work from the machine learning community that aims at solving similar learning problems. In particular, one topic in the research of learning graphical models is to estimate from the observed data an inverse covariance matrix (precision matrix) for Gaussian graphical models [36], [37], [38], [39], [40], [41], [42], especially in the case when the number of observations is smaller than the sample dimension and the sample covariance becomes singular. Another problem consists in inferring the graph structure for discrete Markov Random Fields [43]. It is known that in case of a Gaussian graphical model, there is an exact correspondence between the location of the non-zero entries in the precision matrix and the existence of partial correlations between the random variables [44]. In this case, a maximum-likelihood estimator turns out to be the solution to a log-determinant program. The estimated precision matrix is therefore considered to carry information about the partial correlations between the random variables.

We would however like to emphasize the difference between this type of approaches and our framework by making the following points. First, it is important to notice that in our framework we aim at learning a valid graph Laplacian, which is not the case in the above mentioned works. Indeed, in those cases the learned precision matrix is a full-rank matrix that usually (i) has both positive and negative off-diagonal entries reflecting both positive and negative correlations, and (ii) does not have rows summed up to zero. Non-positive off-diagonal entries and zero row-sum are however necessary constraints to define a graph Laplacian. As a result, the learning of the precision matrix in those methods is not directly linked to the interpretation of global properties of the input graph signals but rather reflects the partial correlations between the random variables that control the observations. Second, it is not straightforward to project the learned precision matrix in those approaches onto the convex set of the constraints that are needed in order to transform it into a valid Laplacian matrix. Therefore, we rather propose in this work to learn a Laplacian matrix directly through the optimization problem. Finally, the work in [8] learns a valid graph topology (the adjacency matrix) with an optimization problem that is very similar to the classical problem of sparse inverse covariance estimation, but with a regularized full-rank Laplacian matrix. In the specific case of an infinite a priori feature variance in the problem formulation, this approach is similar to ours but with an explicit sparsity-promoting term for the adjacency matrix.

To summarize, an essential difference between our method and the above-mentioned machine learning approaches is that, most of these methods do not learn a valid graph Laplacian matrix; they mainly focus on the pairwise correlation between random variables (including the work in [8]), but do not explore the link between the signal model (i.e., global smoothness) and the graph topology. This is exactly the central focus of this paper, whose objective is to learn a representation that permits effective analysis of the graph signals.

III. Graph signal representation via factor analysis

A. Smooth graph signals

We cast the problem of learning graph topology as a problem of learning the so-called graph Laplacian matrix as it uniquely characterizes the graph. Consider a weighted and undirected graph $G = \{V, E, \omega\}$ of n vertices, where V and E represent the vertex and edge sets, respectively, and ω gives a positive weight for each edge in E. A graph signal can then be represented by a function $f : V \to \mathbb{R}^n$ that assigns a scalar value to each vertex. The unnormalized (or combinatorial) graph Laplacian matrix L, which is an n by n matrix, is defined as:

$$L = D - W, \tag{1}$$

where W is the possibly weighted adjacency matrix of the graph and D is the degree matrix that contains the degrees of the vertices along the diagonal. Since L is a real and symmetric matrix, it has a complete set of orthonormal eigenvectors and associated eigenvalues. In particular, it can be written as:

$$L = \chi \Lambda \chi^T, \tag{2}$$

where χ is the eigenvector matrix that contains the eigenvectors as columns, and Λ is the diagonal eigenvalue matrix where the associated eigenvalues are sorted in increasing order. The smallest eigenvalue is 0 with a multiplicity equal to the number of connected components of the graph G [45]. Therefore, if G has c connected components, then the rank of Λ is $n - c$. The Laplacian matrix L enables, via its spectral decomposition in Eq. (2), the generalization of the notion of frequency and Fourier transform for graph signals [13]. This leads to the extension of classical signal processing tools on graphs (see [1] for a detailed survey of these tools).

Recall that we consider in this paper the class of smooth signals on graphs. Equipped with the Laplacian matrix, the smoothness of a graph signal f on G can be measured in terms of a quadratic form of the graph Laplacian [4]:

$$f^T L f = \frac{1}{2} \sum_{i \sim j} w_{ij} \big(f(i) - f(j)\big)^2, \tag{3}$$

where w_{ij} represents the weight on the edge connecting two adjacent vertices i and j, and $f(i)$ and $f(j)$ are the signal values associated with these two vertices. Intuitively, given that the weights are non-negative, Eq. (3) shows that a graph signal f is considered to be smooth if strongly connected vertices (with a large weight on the edge between them) have similar values. In particular, the smaller the quadratic form in Eq. (3), the smoother the signal on the graph. Smooth graph signal models have been widely used in various learning problems such as regularization and semi-supervised learning on graphs [3], [4].

B. Representation via factor analysis model

We now propose a new representation model for smooth graph signals based on an extension of the traditional factor analysis model. The factor analysis model [46], [47] is a generic linear statistical model that tries to explain observations of a given dimension with a potentially smaller number of unobserved latent variables. Specifically, we consider the following representation model for the input graph signal:

$$x = \chi h + u_x + \epsilon, \tag{4}$$

where $x \in \mathbb{R}^n$ represents the observed graph signal, $h \in \mathbb{R}^n$ represents the latent variable that controls the graph signal x through the eigenvector matrix χ, and $u_x \in \mathbb{R}^n$ is the mean of x. As discussed in [48], we adopt an isotropic noise model, namely, we assume that ϵ follows a multivariate Gaussian distribution with mean zero and covariance $\sigma_\epsilon^2 I_n$. The probability density function of ϵ is thus given by:

$$\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 I_n). \tag{5}$$

The model in Eq. (4) can be considered as a generalization of the classical factor analysis model to graph signals, where the key is the choice of the representation matrix as the eigenvector matrix χ of the graph Laplacian L. This relates the graph signals to the latent variable. The motivation of this definition is twofolds. First, we seek a representation matrix that reflects the topology of the graph, such that the construction of the graph can be directly linked to the representation and properties of the graph signals. The eigenvector matrix of the graph Laplacian is a good candidate, since it provides a spectral embedding of the graph vertices that can subsequently be used for many graph-based analysis tasks such as graph partitioning [49]. Second, it can be interpreted as the graph Fourier basis for representing graph signals [13], which makes it a natural choice as the representation matrix in Eq. (4).

As in the classical factor analysis model, we impose a Gaussian prior on the latent variable h in Eq. (4). Specifically, we assume that the latent variable h follows a degenerate zero-mean multivariate Gaussian distribution with the precision matrix defined as the eigenvalue matrix Λ of the graph Laplacian L, i.e.,

$$h \sim \mathcal{N}(0, \Lambda^\dagger), \tag{6}$$

where $\Lambda^\dagger$ is the Moore-Penrose pseudoinverse of Λ. Together with the definition of χ as the representation matrix and its Fourier interpretation, the above assumption on h implies that the energy of the signal tends to lie mainly in the low frequency components[1], hence it promotes the smoothness of the graph signal. Based on Eqs. (4), (5) and (6), the conditional probability of x given h, and the probability of x, are respectively given as:

$$x|h \sim \mathcal{N}(\chi h + u_x, \sigma_\epsilon^2 I_n), \tag{7}$$
$$x \sim \mathcal{N}(u_x, L^\dagger + \sigma_\epsilon^2 I_n), \tag{8}$$

where we have used in Eq. (8) the fact that the pseudoinverse of L, $L^\dagger$, admits the following eigendecomposition:

$$L^\dagger = \chi \Lambda^\dagger \chi^T. \tag{9}$$

It can be seen from Eq. (8) that, in a noise-free scenario where $\sigma_\epsilon = 0$, x also follows a degenerate multivariate

---

[1]We note that the effect of the first (constant) eigenvector of the graph Laplacian is ignored as the corresponding entry in the diagonal of $\Lambda^\dagger$ is zero.

Gaussian distribution. Moreover, x can be seen as a GMRF with respect to the graph G, where the precision matrix is chosen to be the graph Laplacian L. Notice that the GMRF is a very generic model such that the precision matrix can be defined with much freedom, as long as its non-zero entries encode the partial correlations between the random variables, and as long as their locations correspond exactly to the edges in the graph [44]. In particular, the graph Laplacian L is commonly adopted as the precision matrix of the GMRFs that model images with Gaussian probabilistic priors[2] [28]. By defining the representation matrix in the factor analysis model as the eigenvector matrix χ and assuming that the latent variable follows a degenerate Gaussian distribution with the precision matrix Λ, we can therefore recover a GMRF model with a precision matrix equal to L in a noise-free scenario.

In the presence of noise, we see from Eq. (8) that, under a Gaussian prior on the latent variable h that follows Eq. (6), the representation matrix χ is the eigenvector matrix of the covariance of x. Indeed, the covariance matrix $L^\dagger + \sigma_\epsilon^2 I_n$ admits the following eigendecomposition:

$$L^\dagger + \sigma_\epsilon^2 I_n = \chi(\Lambda^\dagger + \sigma_\epsilon^2 I_n)\chi^T. \qquad (10)$$

This is analogous to the classical factor analysis model where the representation matrix spans the same subspace as the k leading principal components of the covariance of x under a Gaussian prior on h. It has been pointed out in [7] that signal representation with the classical factor analysis model provides a probabilistic interpretation of the highly successful representation learned by the PCA, which was originally presented in [50], [48]. Because of the analogy mentioned above, the representation in Eq. (4) can thus be considered as a PCA-like representation for the graph signal x. More importantly, we show in the next section that this promotes the smoothness properties for the signal on the graph, based on which we propose our novel learning framework.

IV. GRAPH LAPLACIAN LEARNING ALGORITHM

A. Learning framework

Given the observation x and the multivariate Gaussian prior distribution of h in Eq. (6), we are now interested in a MAP estimate of h. Specifically, by applying Bayes' rule and assuming without loss of generality that $u_x = 0$, the MAP estimate of the latent variables h can be written as follows [51]:

$$\begin{aligned} h_{\text{MAP}}(x) &:= \arg \max_h p(h|x) \\ &= \arg \max_h p(x|h)p(h) \qquad (11) \\ &= \arg \min_h \left(-\log p_E(x - \chi h) - \log p_H(h)\right), \end{aligned}$$

where $p_E(\epsilon) = p_E(x - \chi h)$ and $p_H(h)$ represent the probability density function (p.d.f.) of the noise and the latent variable, respectively, and $p(h|x)$ is the conditional p.d.f. of h given a realization of x. Now, from the Gaussian probability

distributions shown in Eq. (5) and Eq. (6), the above MAP estimate of Eq. (11) can be expressed as:

$$\begin{aligned} h_{\text{MAP}}(x) &:= \arg \min_h \left(-\log p_E(x - \chi h) - \log p_H(h)\right) \\ &= \arg \min_h \left(-\log e^{-(x - \chi h)^T(x - \chi h)} - \alpha \log e^{-h^T \Lambda h}\right) \\ &= \arg \min_h ||x - \chi h||_2^2 + \alpha \ h^T \Lambda h, \end{aligned}$$
$$\qquad (12)$$

where α is some constant parameter proportional to the variance of the noise σ_ϵ^2 in Eq. (5). In a noise-free scenario where $x = \chi h$, Eq. (12) corresponds to minimizing the following quantity:

$$h^T \Lambda h = (\chi^T x)^T \Lambda \chi^T x = x^T \chi \Lambda \chi^T x = x^T L x. \qquad (13)$$

The Laplacian quadratic term in Eq. (13) is the same as the one in Eq. (3). Therefore, it confirms that in a factor analysis model in Eq. (4), a Gaussian prior in Eq. (6) imposed on the latent variable h leads to smoothness properties for the graph signal x. Similar observations can be made in a noisy scenario, where the main component of the signal x, namely, χh, is smooth on the graph.

We are now ready to introduce the proposed learning framework. Notice that in Eq. (12) both the representation matrix χ and the precision matrix Λ of the Gaussian prior distribution imposed on h come from the graph Laplacian L. They respectively represent the eigenvector and eigenvalue matrices of L. When the graph is unknown, we can therefore have the following joint optimization problem of χ, Λ and h in order to infer the graph Laplacian:

$$\min_{\chi, \Lambda, h} ||x - \chi h||_2^2 + \alpha \ h^T \Lambda h. \qquad (14)$$

The objective in Eq. (14) can be interpreted as the probability of the latent variable h conditioned on the observations x and the graph Laplacian L, which we consider as another variable. Using a change of variable $y = \chi h$, we thus have:

$$\min_{L, y} ||x - y||_2^2 + \alpha \ y^T L y. \qquad (15)$$

According to the factor analysis model in Eq. (4), y can be considered as a "noiseless" version of the zero-mean observation x. Recall that the Laplacian quadratic form $y^T L y$ in Eq. (15) is usually considered as a measure of smoothness of the signal y on the graph. Solving the problem of Eq. (15) is thus equivalent to finding jointly the graph Laplacian L and y, such that y is close to the observation x, and at the same time y is smooth on the learned graph. As a result, it enforces the smoothness property of the observed signals on the learned graph.

B. Learning algorithm

We propose to solve the problem in Eq. (15) with the following objective function given in a matrix form:

$$\min_{L \in \mathbb{R}^{n \times n}, Y \in \mathbb{R}^{n \times p}} ||X - Y||_F^2 + \alpha \ \text{tr}(Y^T L Y) + \beta ||L||_F^2,$$

$$\begin{aligned} \text{s.t.} \quad &\text{tr}(L) = n, \\ &L_{ij} = L_{ji} \le 0, \ i \ne j, \qquad (16) \\ &L \cdot \mathbf{1} = \mathbf{0}, \end{aligned}$$

---

[2]Notice that this is not the case in most of the state-of-the-art approaches for inverse covariance estimation, where the inverse covariance or precision matrix needs to be a nonsingular and full-rank matrix that cannot be interpreted as a graph Laplacian.

where $X \in \mathbb{R}^{n \times p}$ contains the p input data samples $\{x_i\}_{i=1}^{p}$ as columns, α and β are two positive regularization parameters, and $\mathbf{1}$ and $\mathbf{0}$ denote the constant one and zero vectors. In addition, $\mathrm{tr}(\cdot)$ and $||\cdot||_F$ denote the trace and Frobenius norm of a matrix, respectively. The first constraint (the trace constraint) in Eq. (16) acts as a normalization factor and permits to avoid trivial solutions, and the second and third constraints guarantee that the learned L is a valid Laplacian matrix that is positive semidefinite. Furthermore, the trace constraint essentially fixes the L^1-norm of L, while the Frobenius norm is added as a penalty term in the objective function to control the distribution of the off-diagonal entries in L, namely, the edge weights of the learned graph. When Y is fixed, the optimization problem bears similarity to the linear combination of L^1 and L^2 penalties in an elastic net regularization [52] in the sense that the sparsity term is imposed by the trace constraint.

The optimization problem of Eq. (16) is not jointly convex in L and Y. Therefore, we adopt an alternating minimization scheme where, at each step, we fix one variable and solve for the other variable. The solution therefore corresponds to a local minimum rather than a global minimum. Specifically, we first initialize Y as the signal observations X. Then, at the first step, for a given Y, we solve the following optimization problem with respect to L:

$$
\begin{aligned}
\min_{L} \quad & \alpha \, \mathrm{tr}(Y^T L Y) + \beta ||L||_F^2, \\
\text{s.t.} \quad & \mathrm{tr}(L) = n, \\
& L_{ij} = L_{ji} \le 0, \ i \ne j, \\
& L \cdot \mathbf{1} = \mathbf{0}.
\end{aligned}
\tag{17}
$$

At the second step, L is fixed and we solve the following optimization problem with respect to Y:

$$
\min_{Y} \ ||X - Y||_F^2 + \alpha \, \mathrm{tr}(Y^T L Y).
\tag{18}
$$

Both problems of Eq. (17) and Eq. (18) can be cast as convex optimization problems with unique minimizers. First, the problem of Eq. (17) can be written as a quadratic program (QP). In more details, notice that the matrix $L \in \mathbb{R}^{n \times n}$ in the problem of Eq. (17) is symmetric, which means that we only need to solve for the lower triangular part of L, that is, the $\frac{n(n+1)}{2}$ entries on and below the main diagonal. Therefore, instead of the square matrix form, we solve for the half-vectorization of L that is obtained by vectorizing the lower triangular part of L. We denote the half-vectorization and vectorization of L as $\mathrm{vech}(L) \in \mathbb{R}^{\frac{n(n+1)}{2}}$ and $\mathrm{vec}(L) \in \mathbb{R}^{n^2}$, respectively, and the former can be converted into the latter using the duplication matrix $\mathcal{M}_{\mathrm{dup}}$ [53]:

$$
\mathcal{M}_{\mathrm{dup}} \, \mathrm{vech}(L) = \mathrm{vec}(L).
\tag{19}
$$

Now, by using Eq. (19) together with the fact that:

$$
tr(Y^T L Y) = \mathrm{vec}(YY^T)^T \, \mathrm{vec}(L),
\tag{20}
$$

and

$$
||L||_F^2 = \mathrm{vec}(L)^T \, \mathrm{vec}(L),
\tag{21}
$$

we can rewrite the problem of Eq. (17) as:

$$
\begin{aligned}
\arg \min_{\mathrm{vech}(L)} \ & \alpha \, \mathrm{vec}(YY^T)^T \, \mathcal{M}_{\mathrm{dup}} \, \mathrm{vech}(L) \\
& + \beta \, \mathrm{vech}(L)^T \, \mathcal{M}_{\mathrm{dup}}^T \, \mathcal{M}_{\mathrm{dup}} \, \mathrm{vech}(L), \\
\text{s.t.} \quad & A \, \mathrm{vech}(L) = \mathbf{0}, \\
& B \, \mathrm{vech}(L) \le \mathbf{0},
\end{aligned}
\tag{22}
$$

where A and B are the matrices that handle the equality and inequality constraints in Eq. (17). The problem of Eq. (22) is a quadratic program with respect to the variable $\mathrm{vech}(L)$ subject to linear constraints, and can be solved efficiently via interior point methods [54]. As we can see, the computational complexity scales quadratically with the number of vertices in the graph. With graphs of very large number of vertices, we can instead use operator splitting methods (e.g., alternating direction method of multipliers (ADMM) [55]) to find a solution. Finally, once we solve the problem of Eq. (22), we convert $\mathrm{vech}(L)$ into the square matrix form in order to solve the problem of Eq. (18). Second, the problem of Eq. (18) has the following closed form solution:

$$
Y = (I_n + \alpha L)^{-1} X.
\tag{23}
$$

In practice, since the matrix $I_n + \alpha L$ is Hermitian and positive-definite, we can use the Cholesky factorization to compute Y efficiently in Eq. (23) [54]. We then alternate between these two steps of Eq. (17) and Eq. (18) to get the final solution to the problem of Eq. (16), and we generally observe convergence within a few iterations. The complete algorithm is summarized in Algorithm 1.

Algorithm 1 Graph Learning for Smooth Signal Representation (**GL-SigRep**)

1: **Input:** Input signal X, number of iterations iter, α, β
2: **Output:** Output signal Y, graph Laplacian L
3: **Initialization:** $Y = X$
4: **for** $t = 1, 2, ..., \text{iter}$ **do:**
5:     **Step to update Graph Laplacian L:**
6:     Solve the optimization problem of Eq. (17) to update L.
7:     **Step to update Y:**
8:     Solve the optimization problem of Eq. (18) to update Y.
9: **end for**
10: $L = L^{\mathrm{iter}}, Y = Y^{\mathrm{iter}}$.

V. EXPERIMENTS

In this section, we evaluate the performance of the proposed graph learning algorithm. We first describe the general experimental setting, and then we present experimental results on both synthetic and real world data.

A. Experimental settings

We test the performance of our framework by comparing the graphs learned from sets of synthetic or real world observations to the groundtruth graphs. We provide both visual

and quantitative comparisons, where we compare the existence of edges in the learned graph to the ones of the groundtruth graph. We use four evaluation criteria commonly used in information retrieval, namely, *Precision*, *Recall*, *F-measure* and *Normalized Mutual Information (NMI)* [56], to test the performance of our algorithm. For computing the *NMI*, we first compute a 2-cluster partition of all the vertex pairs using the learned graph, based on whether or not there exists an edge between the two vertices. We then compare this partition with the 2-class partition obtained in the same way using the groundtruth graph.

In our experiments, we solve the optimization problem of Eq. (17) using the convex optimization package CVX [57], [58]. Our algorithm **GL-SigRep** stops when the maximum number of iterations is reached or the absolute change in the objective is smaller than 10^{-4}. In most of the cases, we observed that the algorithm converges within a few iterations. The experiments are carried out on different sets of parameters, namely, for different values of α and β in Eq. (16). Finally, we prune insignificant edges that have a weight smaller than 10^{-4} in the learned graph.

We compare the proposed graph learning framework to a state-of-the-art machine learning approach for estimating graph structure, which is closely related to the idea of sparse inverse covariance estimation for GMRF. Specifically, Lake et al. propose in [8] to solve the following optimization problem:

$$
\begin{aligned}
\min_{L_{\text{pre}} \succ 0, W, \sigma^2} \quad & \text{tr}(L_{\text{pre}} \frac{1}{p} X X^T) - \log|L_{\text{pre}}| + \frac{\lambda}{p}||W||_1, \\
\text{s.t.} \quad & L_{\text{pre}} = \text{diag}(\sum_j w_{ij}) - W + I/\sigma^2, \\
& w_{ii} = 0, \ i = 1, \ldots, n, \\
& w_{ij} \geq 0, \ i = 1, \ldots, n; \ j = 1, \ldots, n, \\
& \sigma^2 > 0,
\end{aligned}
\tag{24}
$$

where W is the adjacency matrix of the target graph with edge weight w_{ij} between vertices i and j, L_{pre} is a precision matrix for the multivariate Gaussian distribution that generates the observations X, λ is a regularization parameter, and $|\cdot|$ and $||\cdot||_1$ denote the determinant and the L^1-norm, respectively. It can be seen that the objective function in Eq. (24) is very similar to the L^1 regularized log-determinant program proposed in [38], [39]; however, by adding the constraints in Eq. (24), Lake et al. essentially defines the precision matrix as a regularized graph Laplacian with positive diagonal loading, where the parameter σ^2 is interpreted as the a priori feature variance [59]. As a consequence, by solving the problem of Eq. (24), they are able to obtain a valid Laplacian matrix, which is not the case for the methods in [38], [39]. Therefore, we choose this method instead of the ones in [38], [39] for a more fair comparison. In our experiments we solve the problem of Eq. (24) using CVX, similarly to what is indicated in [8]. We denote this algorithm as **GL-LogDet**. We test **GL-LogDet** based on different choices of the parameter λ in Eq. (24). After obtaining the Laplacian matrix, we prune insignificant connections that have a weight smaller than 10^{-4}, similarly to the post-processing done with **GL-SigRep**.

B. Results on synthetic data

We first carry out experiments on three different synthetic graphs of 20 vertices, namely, a graph whose edges are determined based on Euclidean distances between vertices, and two graphs that follow the Erdős-Rényi model [60] and the Barabási-Albert model [61], respectively. For the first graph, we generate the coordinates of the vertices uniformly at random in the unit square, and compute the edge weights with a Gaussian radial basis function (RBF), i.e., $\exp\left(-d(i,j)^2/2\sigma^2\right)$ where $d(i,j)$ is the distance between vertices and $\sigma = 0.5$ is a kernel width parameter. We then remove all the edges whose weights are smaller than 0.75. Next, we use the Erdős-Rényi (ER) model with edge probability 0.2 to generate a random graph, that is, each possible edge is included in the graph with probability 0.2 independently of other edges. Finally, we use the Barabási-Albert (BA) model to generate a scale-free random graph. Specifically, the BA graph in our experiments is generated by adding one new vertex to the graph at each time, connecting to one existing vertex in the graph. The probability of the new vertex to be linked to a given existing vertex in the graph is proportional to the ratio of the degree of that existing vertex to the sum of degrees of all the existing vertices. The ER and BA graphs are important random graph models studied in network science. Specifically, the former has power-law (or scale-free) degree distributions similarly to many networks observed in real world, while the latter does not. The ER and BA graphs in our experiments have unitary edge weights.

Given a groundtruth synthetic graph, we compute the graph Laplacian L and normalize the trace according to Eq. (16). Then, for each graph, we generate 100 signals $X = \{x_i\}_{i=1}^{100}$ that follow the distribution shown in Eq. (8) with $u_x = 0$ and $\sigma_\epsilon = 0.5$. We then apply **GL-SigRep** and **GL-LogDet** to learn the graph Laplacian matrices, given only the signals X. Finally, we also compute as a baseline the sample correlation matrix S where S_{ij} represents the Pearson product-moment correlation coefficient between signal observations at vertices i and j.

We first provide visual comparisons in Fig. 2, where we show from the left to the right columns the Laplacian matrices of the groundtruth graph, the graph Laplacians learned by **GL-SigRep**, the graph Laplacians learned by **GL-LogDet**, the sample correlation matrices, and the thresholded sample correlation matrices, for one random instance of each of the three graph models[3]. In each row, the values in the Laplacian and correlation matrices are scaled to be within the same range. First, we see that the sample correlation matrix simply computes pairwise correlations and therefore may lead to undesirable noisy correlation values as shown in Fig. 2(d), (i), (n). The thresholded sample correlation matrices show that the positions of the large positive entries in the sample correlation matrices generally correspond to the positions of the edges in the groundtruth graph; however, the intensities of these

---

[3]These results are obtained based on the parameters, namely, α and β in **GL-SigRep**, λ in **GL-LogDet**, and threshold for the thresholded sample correlation matrix, that lead to the best *F-measure* scores (see quantitative results in the next paragraph). More discussion about the choices of these parameters are presented later.
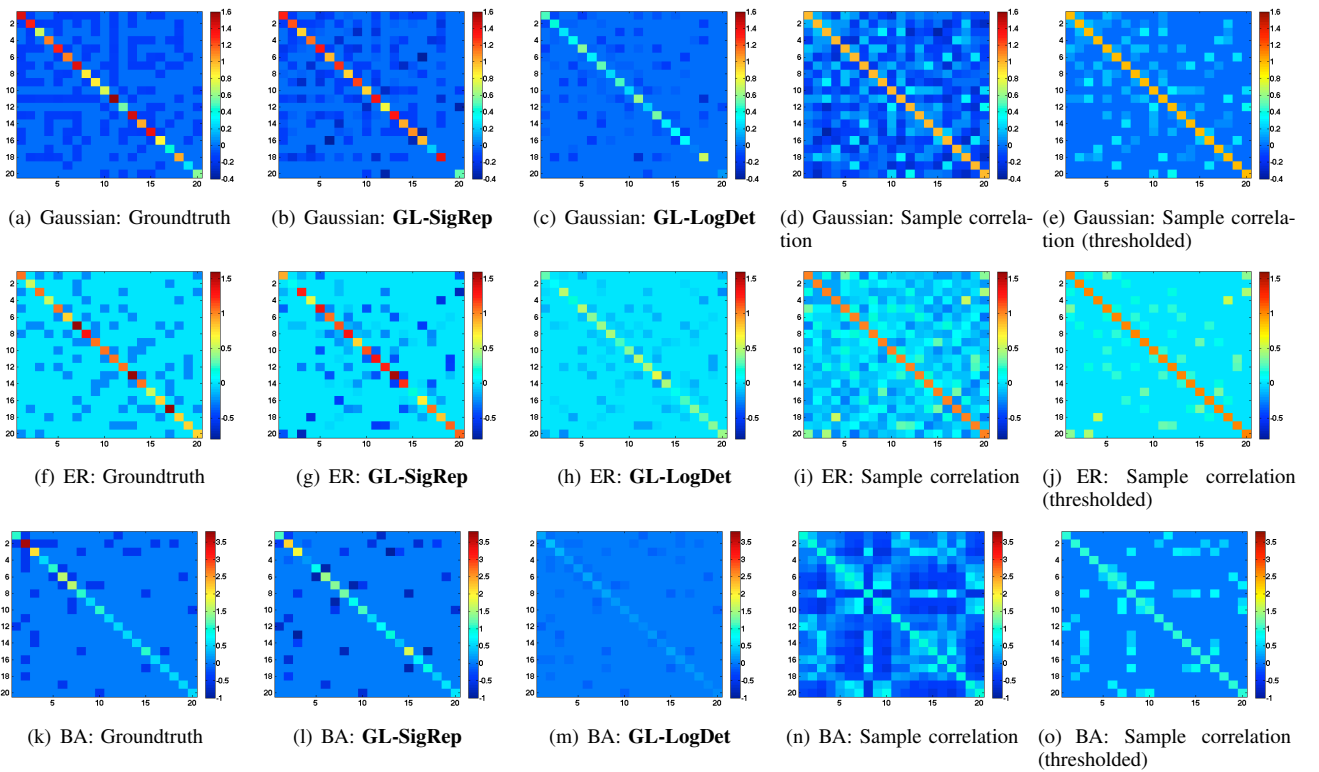
(a) Gaussian: Groundtruth   (b) Gaussian: **GL-SigRep**   (c) Gaussian: **GL-LogDet**   (d) Gaussian: Sample correlation   (e) Gaussian: Sample correlation (thresholded)

(f) ER: Groundtruth   (g) ER: **GL-SigRep**   (h) ER: **GL-LogDet**   (i) ER: Sample correlation   (j) ER: Sample correlation (thresholded)

(k) BA: Groundtruth   (l) BA: **GL-SigRep**   (m) BA: **GL-LogDet**   (n) BA: Sample correlation   (o) BA: Sample correlation (thresholded)

Fig. 2. The learned graph Laplacian matrices as well as sample correlation matrices (original and thresholded). The columns from the left to the right are the groundtruth Laplacians, the Laplacians learned by **GL-SigRep**, the Laplacians learned by **GL-LogDet**, the sample correlation matrices, and the thresholded sample correlation matrices. The rows from the top to the bottom are for the Gaussian RBF graph, the ER graph, and the BA graph.

"edges" are less consistent with the groundtruth than those in the Laplacian matrices learned by our method, as we see later. One possible reason for this is that the sample correlation matrix computes pairwise correlations between (possibly noisy) observations; in comparison, with a global smoothness constraint in our model, we rather learn a graph topology such that the signals are regularized to be globally smooth. Second, we see that for all three types of graphs, the graph Laplacian matrices learned by **GL-SigRep** are visually more consistent with the groundtruth data than the ones learned by **GL-LogDet**, especially in the case of a BA graph.

Next, we quantitatively evaluate the performance of **GL-SigRep**, **GL-LogDet**, and sample correlations, in terms of recovering the position of the edges in the groundtruth graph. For sample correlation matrices, we consider off-diagonal correlations with values greater than a chosen threshold (see Footnote 4 for the exact values) as the "recovered edges". The evaluation metrics we choose are the *F-measure, Precision, Recall* and *Normalized Mutual Information (NMI)* [56]. Specifically, the *Precision* evaluates the percentage of correct edges in the learned graph, that is, the edges that are present in the groundtruth graph. The *Recall* evaluates the percentage of the edges in the groundtruth graph that are present in the learned graph. The *F-measure* thus takes into account both *Precision* and *Recall* to measure the overall accuracy of the obtained edge set. Finally, the *NMI* measures the mutual dependence between the obtained edge set and that of the groundtruth graph from an information theoretic viewpoint.

TABLE I
GRAPH LEARNING PERFORMANCE FOR **GL-SigRep**, **GL-LogDet** AND SAMPLE CORRELATION MATRICES.

(a) The Gaussian RBF graph.

Algorithm	F-measure	Precision	Recall	NMI
GL-SigRep	0.8480	0.8050	0.9007	0.5146
GL-LogDet	0.8159	0.8100	0.8345	0.4632
Samp. Corr.	0.8193	0.8174	0.8361	0.4659

(b) The ER graph.

Algorithm	F-measure	Precision	Recall	NMI
GL-SigRep	0.7236	0.6391	0.8447	0.3695
GL-LogDet	0.7246	0.6273	0.8737	0.3903
Samp. Corr.	0.7019	0.7846	0.6536	0.3725

(c) The BA graph.

Algorithm	F-measure	Precision	Recall	NMI
GL-SigRep	0.9342	0.9514	0.9211	0.8366
GL-LogDet	0.8387	0.8113	0.9158	0.6783
Samp. Corr.	0.6819	0.8149	0.6105	0.4380

Testing on different parameter values in the three methods[4], we show in Table I the best *F-measure, Precision, Recall* and *NMI* scores achieved by the three methods averaged over ten random instances of the three graphs with the associated signals X. Our algorithm **GL-SigRep** provides competitive or

[4] We choose parameters through a grid search. The optimal values for α, β, λ, and threshold for the sample correlation matrix, are 0.012, 0.79, 19.95 and 0.06, respectively, for the Gaussian RBF graph; 0.0032, 0.10, 12.59 and 0.10, respectively, for the ER graph; and 0.0025, 0.050, 125.89 and 0.46, respectively, for the BA graph.
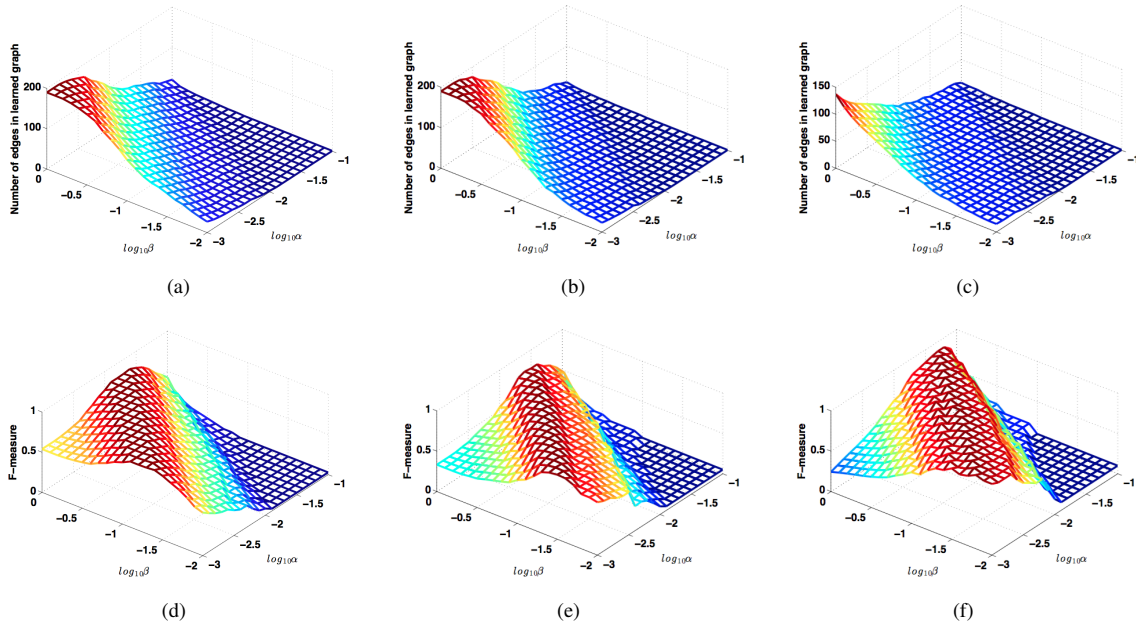
Fig. 3. (a-c) The number of edges in the learned graph, and (d-f) the *F-measure* score, under 441 different combinations of the parameters α and β, for a random instance of the Gaussian RBF graph (left column), ER graph (middle column), and BA graph (right column).

superior performance compared to the other two in terms of all the evaluation criteria. Especially, for the Gaussian RBF and BA graphs, the high average *F-measure* scores achieved by **GL-SigRep** suggest that the learned graphs have topologies that are very similar to the groundtruth ones. The advantage of **GL-SigRep** for the non-manifold structured ER graphs is less obvious, possibly due to the fact that the edges are generated independently in this random graph model. This may lead to a weaker global relationship in the data and therefore affects the efficiency of a global relationship model like ours. Finally, we have also computed the mean squared error (MSE) between the groundtruth and the learned Laplacian matrices in order to study the accuracy of the edge weights. For the sample correlation method, we construct an adjacency matrix where the edges correspond to the correlations above the chosen threshold (the ones in Footnote 4), and then compute accordingly the Laplacian matrix. The MSE is 1.4345 for **GL-SigRep** compared to 2.8443 for **GL-LogDet** and 2.0508 for sample correlation in case of the Gaussian RBF graph, 2.1737 compared to 3.4350 and 3.0117 in case of the ER graph, and 3.0412 compared to 6.1491 and 5.2262 in case of the BA graph. This shows that, in addition to edge positions, the edge weights learned by **GL-SigRep** are more consistent with the groundtruth than those of the other methods.

To better understand the behavior of **GL-SigRep** under different sets of parameters, we plot in Fig. 3(a) the numbers of edges in the learned graph, and in Fig. 3(d) the *F-measure* scores, under $21 \times 21 = 441$ different combinations of the parameters α and β in Eq. (16), for a random instance of the Gaussian RBF graph. First, we see that the number of edges in the learned graph decreases as β decreases and α increases. The intuitions behind this behavior are as follows. When β increases, the Frobenius norm of L in the objective function in Eq. (16) tends to be small. Given a fixed L^1-norm

of L (which is enforced by the trace constraint), this leads to a more uniform distribution of the off-diagonal entries (with similar but smaller values), so that the number of edges tends to increase. Decreasing β leads to the opposite effect. On the other hand, when α increases, the trace of the quadratic term tends to be small. The algorithm thus favors a smaller number of non-zero entries in L, and the number of edges decreases. Therefore, both parameters α and β implicitly affect the sparsity of the learned graph Laplacian. It is interesting to notice in Fig. 3(a) and Fig. 3(b) that, both the number of edges and the *F-measure* scores are similar for the values of α and β that lead to the same ratio $\frac{\beta}{\alpha}$. The same patterns are also observed for the ER graph model (Fig. 3(b)(e)) and the BA graph model (Fig. 3(c)(f)). This suggests that, when the initial observations are relatively smooth, the trace of the quadratic term and the Frobenius norm are the dominating factors in the optimization of Eq. (16), rather than the data fidelity term. This implies that, in practice, we may only need to search for an appropriate ratio $\frac{\beta}{\alpha}$ to maximize the learning performance of the algorithm.

Next, we show in Fig. 4(a) and Fig. 4(b) the number of edges in the graphs learned by **GL-SigRep**, and the learning performance evaluated based on the four criteria, respectively, for the same Gaussian RBF graph as before but under different ratios of β to α. As expected, the number of edges decreases as the ratio of β to α decreases. Looking at Fig. 4(a) and Fig. 4(b) together, we see that, as the number of edges approaches the number of edges in the groundtruth graph (in this case, 70 edges), the *Recall* stays high and the *Precision* increases rapidly, which makes the *F-measure* increase. When the number of edges in the learned graph is close to the one in the groundtruth graph, the curves for the *Precision* and the *Recall* intersect and the *F-measure* reaches its peak. After that, although the *Precision* keeps increasing towards 1, the *Recall*
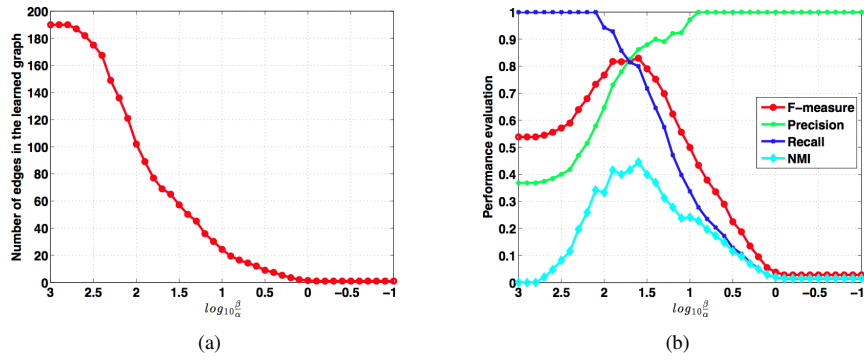
Fig. 4. (a) Number of edges in the graphs learned by **GL-SigRep**, and (b) performance of **GL-SigRep**, for different ratios $\frac{\beta}{\alpha}$ for a random instance of the Gaussian RBF graph.
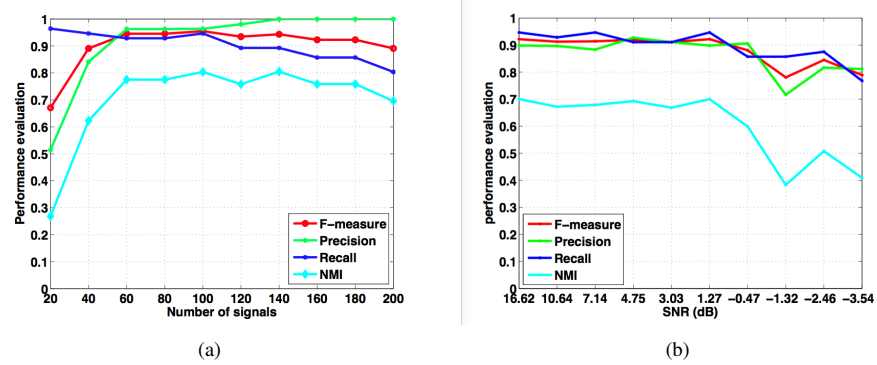

Fig. 5. Performance of **GL-SigRep** for (a) different numbers of signals p and (b) different SNR (dB), for two random instances of the Gaussian RBF graph, respectively.

drops rapidly as fewer and fewer edges are detected, leading to a decreasing trend in the *F-measure*. A similar trend can be observed in the curve for the *NMI* score. The patterns in Fig. 4 are also observed for the ER and BA graph models. These results show that **GL-SigRep** is able to learn a graph that is very close to the groundtruth graph when the number of edges matches the number of edges in the groundtruth graph.

Finally, we investigate the influence of the number of signals available for learning, and the level of noise present in the data, for two random instances of the Gaussian RBF graph, respectively. In Fig. 5(a), we show the performance of **GL-SigRep** for different numbers of signals. As we can see, the performance initially increases as more signals are available to learn the graph Laplacian, but remains quite stable when more than 60 signals are available. In Fig. 5(b), we show the performance of our algorithm for different values of the signal-to-noise-ratio (SNR) given the Gaussian noise σ_ϵ. We see that the performance of **GL-SigRep** again remains stable until the SNR becomes very low.

C. Learning meteorological graph from temperature data

We now test the proposed graph learning framework on real world data. We first consider monthly temperature data collected at 89 measuring stations in Switzerland (shown in Fig. 6(a)) during the period between 1981 and 2010 [62]. Specifically, for each station, we compute the average temperature for each month over the 30 years. This leads

to 12 signals that measure the average temperatures at the 89 measuring stations in the 12 months. By applying the proposed graph learning algorithm, we would like to infer a graph where stations with similar temperature evolutions across the year are connected. In other words, we aim at learning a graph on which the observed temperature signals are smooth. In this case, the natural choice of a geographical graph based on physical distances between the stations does not seem appropriate for representing the similarity of temperature values between these stations. Indeed, Fig. 6(b)-(d) show the average temperatures in Switzerland in February, June and October, and we can see that the evolution of temperatures at most of the stations follows very similar trends, which implies that the stations' values are highly correlated, regardless of the geographical distances between them. On the other hand, it turns out that altitude is a more reliable source of information to determine temperature evolutions. For instance, as we observed from the data, the temperatures at two stations, Jungfraujoch and Piz Corvatsch, follow similar trends that are clearly different from other stations, possibly due to their similar altitudes (both are more than 3000 metres above sea level). Therefore, the goal of our learning experiments is to recover a graph that reflects the altitude relationships between the stations given the observed temperature signals.

We apply the two graph learning algorithms **GL-SigRep** and **GL-LogDet** on the temperature signals to learn two graph Laplacian matrices. For quantitative evaluation, we build a groundtruth graph that reflects the similarity between stations
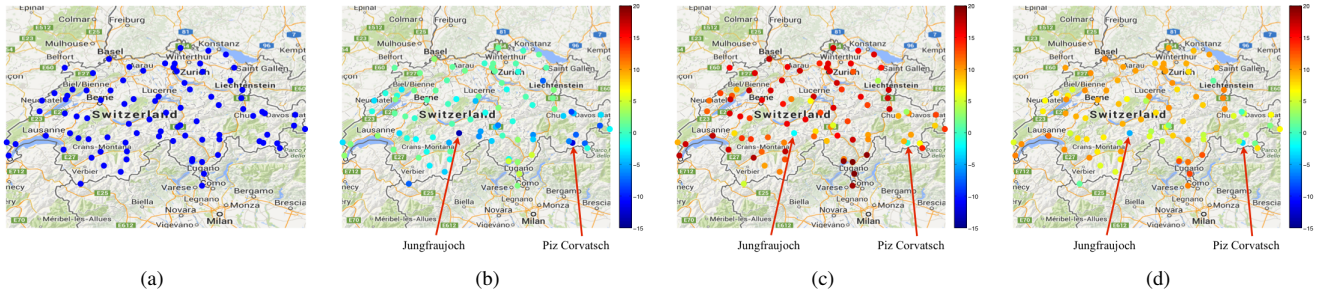
Fig. 6. (a) The locations of 89 measuring stations in Switzerland. (b)-(d) Average monthly temperature in February, June and October, respectively, in Switzerland during the period from 1981 to 2010. The color code in (b)-(d) represents the temperature in Celsius scale.

TABLE II
PERFORMANCE FOR **GL-SIGREP** AND **GL-LOGDET** IN LEARNING THE METEOROLOGICAL GRAPH.

Algorithm	F-measure	Precision	Recall	NMI
GL-SigRep	0.8374	0.7909	0.8896	0.4910
GL-LogDet	0.8052	0.7718	0.8417	0.4249

in terms of their altitudes. More specifically, we connect two stations with a unitary weight if and only if their altitude difference is smaller than 300 metres. We then compare the groundtruth and learned graph Laplacians using the same evaluation metrics as in the synthetic experiments.

We first show visual comparisons between the Laplacian of the groundtruth altitude-based graph and the graph Laplacians learned by **GL-SigRep** and **GL-LogDet**. For a more clear visualization, we focus on the top left part of the three matrices and plot them in Fig. 7. The comparisons between **GL-SigRep** and **GL-LogDet** show that the edges in the graph learned by our algorithm are again more consistent with the groundtruth data. This is confirmed by the results shown in Table II where the best *F-measure*, *Precision*, *Recall* and *NMI* scores achieved by the two algorithms are presented. Overall, both visual and quantitative comparisons show that **GL-SigRep** achieves better performance than **GL-LogDet** in inferring the graph topology from the signal observations in this example.

To further verify our results, we separate the measuring stations into disjoint clusters based on the graph learned by **GL-SigRep**, such that different clusters correspond to different characteristics of the measure stations. In particular, since we obtain a valid Laplacian matrix, we can apply spectral clustering [63] to the learned graph to partition the vertex set into two disjoint clusters. The results are shown in Fig. 8(a), where the red and blue dots represent two different clusters of stations. As we can see, the stations in the red cluster are mainly those located on the mountains, such as those in the Jura Mountains and Alps, while the ones in the blue cluster are mainly stations in flat regions. It is especially interesting to notice that, the blue stations in the Alps region (from centre to the bottom middle of the map) mainly lie in the valleys along main roads (such as those in the canton of Valais) or in the Lugano region. For comparison, we show in Fig. 8(b) a 2-cluster partition of the stations by applying k-means clustering algorithm using directly the observed temperature records. We see that the clusters obtained in this way are less consistent with the altitude information of the stations. For instance,

some of the stations in mountain regions in Jura, Valais and eastern Switzerland (highlighted by the cyan ellipse) have been clustered together with low-altitude stations. One possible explanation could be that the k-means algorithm treats the observations at each vertex of the graph independently and does not emphasize a global relationship during the clustering process, therefore it may not be able to capture a global similarity pattern as required in our graph learning task. These results show that the clusters obtained by **GL-SigRep** capture the altitude information of the measuring stations hence confirms the quality of the learned graph topology.

D. Learning climate graph from evapotranspiration data

In this application, we consider average monthly evapotranspiration (ETo) data recorded at 56 measuring stations in California between January 2012 and December 2014 [64]. Similarly to the previous example, for each station, we compute the average evapotranspiration for each month over the three years. This leads to 12 signals that measure the average evapotranspiration at the 56 measuring stations in the 12 months. By applying our graph learning framework, we would like to infer a graph that captures the similarities between these measuring stations in terms of the monthly variations of evapotranspiration at their locations. In this example, we do not have a readily available groundtruth graph; however, we have a reference ETo Zone Map [65], which classifies each of the 56 stations into one of the 4 ETo zones according to the terrain and climate conditions at its location[5]. This can thus be considered as a groundtruth clustering of the 56 stations, as shown in Fig. 9. Therefore, for performance evaluation, we apply spectral clustering to the learned graphs using **GL-SigRep** and **GL-LogDet**, respectively, and we partition the stations into 4 clusters. We then compare the resulting clusters with the groundtruth information.

In Table III, we show the overall best scores achieved by the two algorithms in terms of three evaluation metrics for clustering, namely, *NMI*, *Purity* and *Rand Index (RI)* [56]. Even though this clustering task is challenging - indeed, according to the descriptions of the ETo Zone Map, there exists only slight difference between zone 12 in green and zone 14 in orange - we nevertheless see from Table III that the

---

[5]In our experiment, we consider stations from the four largest zones that contain at least 10 stations, namely, zone 6, 12, 14 and 18.

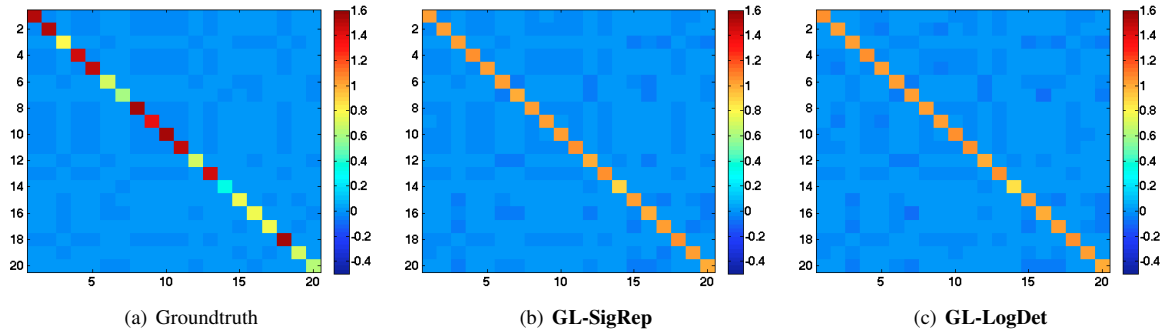(a) Groundtruth   (b) GL-SigRep   (c) GL-LogDet

Fig. 7. Visual comparisons between (a) the groundtruth graph Laplacian, (b) the graph Laplacian learned by **GL-SigRep**, and (c) the graph Laplacian learned by **GL-LogDet**. The results for **GL-SigRep** and **GL-LogDet** are obtained based on the parameters α, β and λ that lead to the best *F-measure* scores. The values in the graph Laplacian learned by **GL-LogDet** is scaled so that it has the same trace as the other two matrices.

(a)   (b)
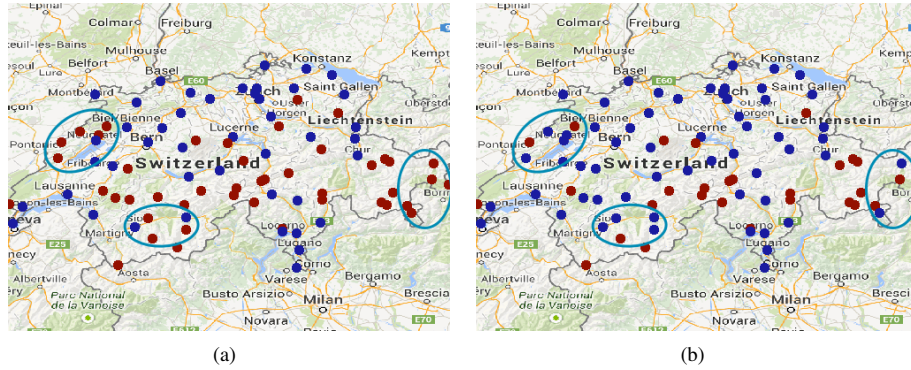
Fig. 8. Two clusterings of the measuring stations obtained by (a) applying spectral clustering to the learned graph, and (b) applying k-means to the raw temperature records. The red cluster includes stations at higher altitudes and the blue cluster includes stations at lower altitudes. Some of the stations with high altitudes, highlighted by the cyan ellipses, are clustered together with low-altitude stations by k-means.
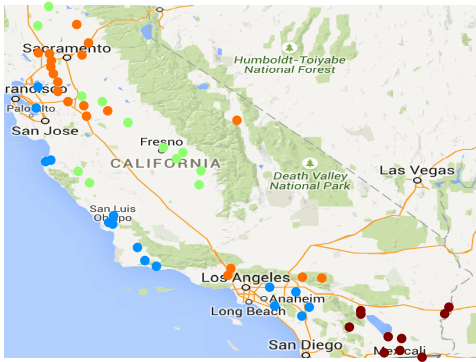

Fig. 9. Locations of the 56 measuring stations in California. The colors indicate 4 groundtruth clusters that correspond to 4 ETo zones.

TABLE III
PERFORMANCE FOR **GL-SigRep** AND **GL-LogDet** IN TERMS OF
RECOVERING GROUDTRUTH CLUSTERS OF MEASURING STATIONS.

Algorithm	NMI	Purity	RI
GL-SigRep	0.5813	0.7321	0.8039
GL-LogDet	0.5093	0.7321	0.7844

clusters obtained based on the graph learned by **GL-SigRep** is more consistent with the groundtruth than that based on **GL-LogDet**.

E. Learning political graph from voting data

We now move onto the final real world example, where we consider voting data from the national referendums for 37 federal popular initiatives in Switzerland between 2008 and 2012 [66]. Specifically, we consider the percentage of votes supporting each initiative in the 26 Swiss cantons as our observed signal. This leads to 37 signals (i.e., one per initiative), each of dimension 26. By applying the proposed graph learning framework, we would like to infer a graph that captures the political relationships between the Swiss cantons in terms of their votes in the national referendums. In this example, we neither have an obvious groundtruth relationship graph nor a groundtruth partitions of the cantons in terms of their political preferences. Therefore, we focus in this example on validating our results, by interpreting the clusters obtained by partitioning the learned graph using **GL-SigRep**.

In Fig. 10(a), we show one 3-cluster partition obtained by applying spectral clustering to the graphs learned by **GL-SigRep**. We can see that the blue cluster contains all the French-speaking cantons, while the yellow clusters contain most of the German-speaking cantons and the Italian-speaking canton Ticino. Then, the five cantons in the red cluster, namely, Uri, Schwyz, Nidwalden, Obwalden and Appenzell Innerrhoden, the first four of which constituting the so-called "primitive" cantons at the origin of Switzerland, are all considered among the most conservative cantons in Switzerland. The cluster membership of the canton Basel-Stadt can be
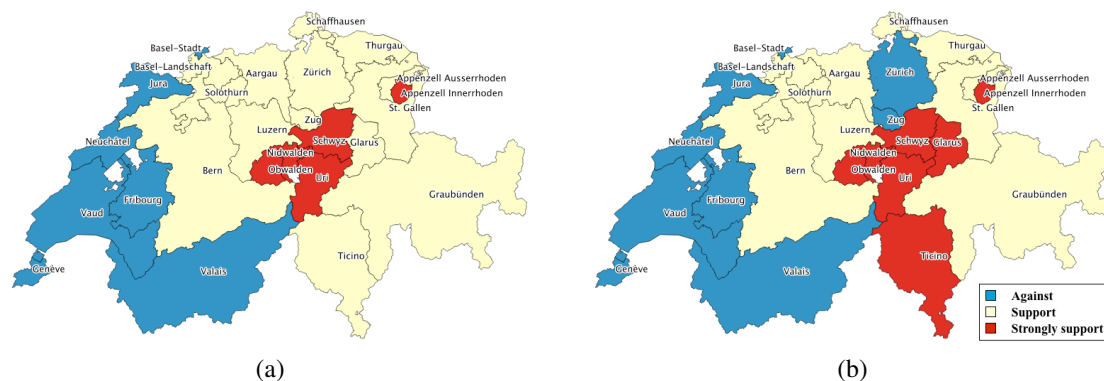
Fig. 10. (a) Clustering of the 26 Swiss cantons based on their votes in the national referendums between 2008 and 2012. (b) The results in the 26 Swiss cantons of the national referendum for the federal popular initiative "Against mass immigration".

explained by the fact that Basel regularly agrees with the cantons in the French-speaking part of Switzerland on referendums for close relations with the European Union. Therefore, this clustering result demonstrates that the graph learned by **GL-SigRep** indeed captures the hidden political relationships between the 26 Swiss cantons, which are consistent with the general understanding of their voting behaviors in the national referendums.

Finally, to confirm the meaningfulness of the clusterings shown in Fig. 10(a), we illustrate in Fig. 10(b) a 3-cluster partition of the cantons based on voting statistics in a recent national referendum for the initiative "Against mass immigration". This referendum, which was held in February 2014, has the largest turnout of 55.8% in recent years. In Fig. 10(b), the cantons in the blue cluster voted against the initiative, while the ones in the yellow and red clusters voted for it. In particular, the seven cantons where the percentage of voters supporting the initiative is greater than 58.2% are grouped in the red cluster. As we can see, this partition is largely consistent with those in Fig. 10(a), with the only exceptions of the cantons of Zürich and Zug agreeing with the French-speaking cantons with small margins. This confirms the clustering in Fig. 10(a) hence demonstrates the quality of the graphs learned by **GL-SigRep**.

VI. Conclusion

In this paper, we have presented a framework for learning graph topologies (graph Laplacians) from signal observations under the assumption that the signals are smooth on the learned graph. Specifically, we have developed a method for learning graphs that enforces the smoothness property of the graph signals, under a Gaussian prior distribution imposed on the latent variables in a generalized factor analysis model. We believe the proposed graph learning framework can benefit numerous signal processing and learning tasks on graphs through the learning of an appropriate graph topology that captures relationships between data entities. Furthermore, it opens new perspectives in the field of graph signal processing. In particular, although we only focus on a Gaussian prior in the present paper, we can also impose other statistical priors on the latent variables in the generalized factor analysis model. We leave such an analysis for future work.

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] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, Apr 2013.

[3] A. Smola and R. Kondor, "Kernels and Regularization on Graphs," in *Proceedings of the 16th Annual Conference on Computational Learning Theory (COLT)*, 2003, pp. 144–158.

[4] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *ICML Workshop on Statistical Relational Learning*, 2004, pp. 132–137.

[5] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov, "Clustering with multi-layer graphs: A spectral perspective," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5820–5831, Nov 2012.

[6] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Laplacian matrix learning for smooth graph signal representation," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 3736–3740.

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug 2013.

[8] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graph," in *Proceedings of the 32nd Annual Meeting of the Cognitive Science Society (CogSci 2010)*, 2010, pp. 778–783.

[9] D. I Shuman, B. Ricaud, and P. Vandergheynst, "A windowed graph Fourier transform," in *Proceedings of the IEEE Statistical Signal Processing Workshop (SSP)*, 2012, pp. 133–136.

[10] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 3921–3924.

[11] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 53–94, Jul 2006.

[12] M. Gavish, B. Nadler, and R. Coifman, "Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2010, pp. 367–374.

[13] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, Mar 2011.

[14] I. Ram, M. Elad, and I. Cohen, "Generalized tree-based wavelet transform," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4199–4209, Sep 2011.

14

[15] S. K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter-banks for graph structured data," *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 2786–2799, Jun 2012.

[16] D. I Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, "Spectrum-adapted tight graph wavelet and vertex-frequency frames," *IEEE Transactions on Signal Processing*, vol. 63, no. 16, pp. 4223–4235, Aug 2015.

[17] X. Zhang, X. Dong, and P. Frossard, "Learning of structured graph dictionaries," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 3373–3376.

[18] D. Thanou, D. I Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," *IEEE Transactions on Signal Processing*, vol. 62, no. 15, pp. 3849–3862, Aug 2014.

[19] D. I Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," *Applied and Computational Harmonic Analysis*, vol. 40, no. 2, pp. 260–291, Mar 2016.

[20] A. Agaskar and Y. M. Lu, "A spectral graph uncertainty principle," *IEEE Transactions on Information Theory*, vol. 59, no. 7, pp. 4338–4356, Jul 2013.

[21] G. Shen, W.-S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *Proceedings of Picture Coding Symposium (PCS)*, 2010, pp. 566–569.

[22] C. Hu, L. Cheng, J. Sepulcre, G. E. Fakhri, Y. M. Lu, and Q. Li, "Matched signal detection on graphs: Theory and application to brain network classification," in *Proceedings of International Conference on Information Processing in Medical Imaging (IPMI)*, 2013, pp. 1–12.

[23] X. Dong, A. Ortega, P. Frossard, and P. Vandergheynst, "Inference of mobility patterns via spectral graph wavelets," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 3118–3122.

[24] N. Tremblay and P. Borgnat, "Graph Wavelets for Multiscale Community Mining," *IEEE Transactions on Signal Processing*, vol. 62, no. 20, pp. 5227–5239, Oct 2014.

[25] M. Hein, J.-Y. Audibert, and U. von Luxburg, "From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians," in *Proceedings of the 18th Annual Conference on Computational Learning Theory (COLT)*, 2005, pp. 470–485.

[26] A. Singer, "From graph to manifold Laplacian: The convergence rate," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 128–134, 2006.

[27] M. Belkin and P. Niyogi, "Towards a theoretical foundation for Laplacian-based manifold methods," *Journal of Computer and System Sciences*, vol. 74, no. 8, pp. 1289–1308, 2008.

[28] C. Zhang and D. Florencio, "Analyzing the optimality of predictive transform coding using graph-based models," *IEEE Signal Processing Letters*, vol. 20, no. 1, pp. 106–109, Jan 2013.

[29] C. Hu, L. Cheng, J. Sepulcre, G. E. Fakhri, Y. M. Lu, and Q. Li, "A graph theoretical regression model for brain connectivity learning of Alzheimer's disease," in *Proceedings of the IEEE International Symposium on Biomedical Imaging (ISBI)*, 2013, pp. 616–619.

[30] S. I. Daitch, J. A. Kelner, and D. A. Spielman, "Fitting a graph to vector data," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2009, pp. 201–208.

[31] N. Leonardi and D. Van De Ville, "Wavelet frames on graphs defined by fMRI functional connectivity," in *Proceedings of the IEEE International Symposium on Biomedical Imaging (ISBI)*, 2011, pp. 2136–2139.

[32] N. Leonardi, J. Richiardi, M. Gschwind, S. Simioni, J.-M. Annoni, M. Schluep, P. Vuilleumier, and D. Van De Ville, "Principal components of functional connectivity: A new approach to study dynamic brain connectivity during rest," *NeuroImage*, vol. 83, pp. 937–950, Dec 2013.

[33] A. Argyriou, M. Herbster, and M. Pontil, "Combining Graph Laplacians for Semi-Supervised Learning," in *Advances in Neural Information Processing Systems 18 (NIPS)*, 2005, pp. 67–74.

[34] G. Dai and D.-Y. Yeung, "Kernel selection for semi-supervised kernel machines," in *Proceedings of the 24th International Conference on Machine Learning*, 2007, pp. 185–192.

[35] P. Shivaswamy and T. Jebara, "Laplacian spectrum learning," in *European Conference on Machine Learning (ECML)*, 2010, pp. 261–276.

[36] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the LASSO," *The Annals of Statistics*, vol. 34, no. 3, pp. 1436–1462, 2006.

[37] M. Yuan and Y. Lin, "Model selection and estimation in the Gaussian graphical model," *Biometrika*, vol. 94, no. 1, pp. 19–35, 2007.

[38] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian

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

[40] A. J. Rothman, P. J. Bickel, E. Levina, and J. Zhu, "Sparse permutation invariant covariance estimation," *Electronic Journal of Statistics*, vol. 2, pp. 494–515, 2008.

[41] P. Ravikumar, G. Raskutti, M. J. Wainwright, and B. Yu, "Model selection in Gaussian graphical models: High-dimensional consistency of ℓ_1-regularized MLE," in *Advances in Neural Information Processing Systems 21 (NIPS)*, 2008, pp. 1329–1336.

[42] C.-J. Hsieh, M. A. Sustik, I. S. Dhillon, and P. Ravikumar, "Sparse inverse covariance matrix estimation using quadratic approximation," in *Advances in Neural Information Processing Systems 24 (NIPS)*, 2011, pp. 2330–2338.

[43] P. Loh and M. J. Wainwright, "Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses," *The Annals of Statistics*, vol. 41, no. 6, pp. 3022–3049, 2013.

[44] H. Rue and L. Held, *Gaussian Markov random fields: Theory and applications*. Chapman and Hall/CRC, 2005.

[45] F. R. K. Chung, *Spectral graph theory*. American Mathematical Society, 1997.

[46] D. J. Bartholomew, M. Knott, and I. Moustaki, *Latent variable models and factor analysis: A unified approach, 3rd Edition*. Wiley, 2011.

[47] A. Basilevsky, *Statistical factor analysis and related methods*. Wiley, 1994.

[48] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[49] U. von Luxburg, "A Tutorial on Spectral Clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec 2007.

[50] S. Roweis, "EM algorithms for PCA and sensible PCA," *CNS Technical Report CNS-TR-97-02, Caltech*, 1997.

[51] R. Gribonval, "Should penalized least squares regression be interpreted as maximum a posteriori estimation?" *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2405–2410, May 2011.

[52] H. Zou and T. Hastie, "Regularization and Variable Selection via the Elastic Net," *Journal of the Royal Statistical Society, Series B*, vol. 67, Part 2, pp. 301–320, 2005.

[53] K. M. Abadir and J. R. Magnus, *Matrix Algebra*. Cambridge University Press, 2005.

[54] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[55] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[56] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge University Press, 2008.

[57] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.0 beta," http://cvxr.com/cvx, September 2013.

[58] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.

[59] X. Zhu, J. Lafferty, and Z. Ghahramani, "Semi-supervised learning: From gaussian fields to gaussian processes," in *Tech. Rep. No. CMU-CS-03-175, Carnegie Mellon University*, 2003.

[60] P. Erdős and A. Rényi, "On the evolution of random graphs," *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.

[61] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct 1999.

[62] http://www.meteoswiss.admin.ch/home/climate/past/climate-normals/climate-diagrams-and-normal-values-per-station.html.

[63] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14 (NIPS)*, 2001, pp. 849–856.

[64] http://www.cimis.water.ca.gov.

[65] http://www.cimis.water.ca.gov/App_Themes/images/etozonemap.jpg.

[66] http://www.swissvotes.ch.