Enhanced Graph-Learning Schemes Driven by Similar Distributions of Motifs

Samuel Rey, *Student Member, IEEE*, T. Mitchell Roddenberry, *Student Member, IEEE*, Santiago Segarra, *Senior Member, IEEE*, and Antonio G. Marques, *Senior Member, IEEE*

***Abstract*—This paper looks at the task of network topology inference, where the goal is to learn an unknown graph from nodal observations. One of the novelties of the approach put forth is the consideration of *prior information* about the *density of motifs* of the unknown graph to enhance the inference of classical Gaussian graphical models. Directly dealing with the density of motifs constitutes a challenging combinatorial task. However, we note that if two graphs have similar motif densities, one can show that the expected value of a polynomial applied to their empirical spectral distributions will be similar. Guided by this, we first assume that we observe a reference graph with a density of motifs similar to that of the sought graph, and then, we exploit this relation by incorporating a similarity constraint and a regularization term in the graph learning optimization problem. The (non-)convexity of the optimization problem is discussed, and a computationally efficient alternating majorization-minimization algorithm is designed. We assess the performance of the proposed method through exhaustive numerical experiments, where different constraints are considered and compared against popular alternatives on both synthetic and real-world datasets.**

***Index Terms*—Network topology inference, graphical models, graph signal processing, motif distribution.**

I. Introduction

HARNESSING graphs to model the underlying structure of signals is gaining relevance due to the rising of data defined over non-Euclidean domains. This graph-based perspective is at the core of graph signal processing (GSP) and machine learning over graphs, fields devoted to the development of methods for processing and learning from signals defined over irregular supports modeled by graphs [1], [2], [3], [4]. Successful applications of these methods are found when processing signals in power, communication, social, geographical, financial, and brain networks, to name a few [1], [5], [6], [7]. While the default approach is to assume that the graph is known and to focus on the processing of the network data, there are many relevant scenarios where the topology of the graph is unknown. To handle this, a preliminary (critical) step is to learn the topology of the graph from a set of nodal observations. The key to this task, which is commonly known as *network topology inference* or *graph learning*, is to leverage models/assumptions relating the properties of the observed signals to the topology of the sought graph [8], [9], [10], [11], [12]. Noteworthy approaches to this task include partial correlations and Gaussian graphical models [13], [14], [15], [16], [17], sparse structural equation models [18], [19], smooth (total variation) models [20], [21], [22], and graph stationary models [8], [23], [24], [25], among others.

All the aforementioned graph-learning approaches share one common characteristic: the focus is placed on the *signals* rather than the graphs. Indeed, most works learn the graph that best explains the observations without considering any prior information about the topology of the graph other than its sparsity. If information about the topological structure of the graph is available, we can harness it to improve the quality of the estimated graphs by promoting desired structural characteristics. An initial step in this direction is taken in joint graph-learning algorithms [15], [26], [27], [28], where several graphs are jointly estimated under the additional assumption that they are close to each other in some sense. This assumption is indeed justified when, e.g., the graphs being estimated proceed from the same distribution. Nonetheless, measuring the distance between two graphs is a non-trivial endeavor, and joint inference works are typically constrained to comparing graphs with a common set of nodes and promoting similar edge support across all graphs.

Some other works are also starting to take into consideration prior information about the graph. A relevant example is found in [17], where the authors propose recovering the graph Laplacian from a set of Gaussian Markov random field (GMRF) observations while considering simple spectral constraints like setting the number of zero eigenvalues. However, as we discuss in Section III-B, these constraints are limited and cannot capture more complex information. Alternatively, [24] introduces a different graph learning method where the unknown graph is assumed to be drawn from a graphon. The main limitations of such an approach are that the graphon is assumed to be known, which may not be trivial in practice since it involves knowing the distribution of the unknown graph, and moreover, that not every graph may be represented as a graphon.

Manuscript received 20 March 2023; revised 1 August 2023; accepted 4 August 2023. Date of publication 10 August 2023; date of current version 5 September 2023. This work was supported in part by the Spanish AEI under Grants PID2019-105032GB-I00, PID2022-136887NB-I00, FPU17/04520, and EST21/00420; in part by the Community of Madrid (Madrid ELLIS Unit); in part by the USA NSF award CCF-2008555; and in part by the USA ARO award W911NF-23-1-0040. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Cihan Tepedelenlioglu. *(Corresponding author: Antonio G. Marques.)*

Samuel Rey and Antonio G. Marques are with the Department of Signal Theory and Communications, King Juan Carlos University, 28933 Madrid, Spain (e-mail: samuel.rey.escudero@urjc.es; antonio.garcia.marques@urjc.es).

T. Mitchell Roddenberry and Santiago Segarra are with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: mitch@rice.edu; segara@rice.edu).

Digital Object Identifier 10.1109/TSP.2023.3303639

In contrast with previous works, this paper proposes a novel graph-learning algorithm that considers prior information about the topology of the graph in a general yet informative way. We lead with the assumption that a reference graph with a *density of motifs similar* to that of the sought graph is known. Note that access to such a reference graph is common in the context of dynamic graphs, graph sampling, or multi-layered graphs [29], [30]. Furthermore, reference graphs have been successfully employed in, e.g., face recognition and genome inference [31], [32]. In this work, to harness the information encoded in the reference graph while avoiding the challenges associated with the combinatorial nature of motifs, we reveal a connection between the spectra of both graphs. Then, we approach the graph learning task as an optimization problem where we exploit the spectral similarity between the reference and the sought graph as a constraint. Because the resulting algorithm is derived from the density of motifs, it is local in nature, which allows us to compare graphs of different sizes (as described in further detail in later sections). Furthermore, the proposed similarity constraints involve the *empirical distribution* of the eigenvalues, which results in constraints that are more informative than the ones considered in previous works. To the best of our knowledge, this is the first graph learning algorithm harnessing the density of motifs to capture prior information about the graph topology.

After reviewing basic ideas in graph signal processing and graph learning in Section II, the structure and main contributions of the paper are summarized next:

1. We relate the structural characteristics of a graph described by the density of motifs to the graph spectrum (Section III-A), and introduce the similarity constraints to propose an optimization program for network topology inference (Section III-B).
2. We analyze the non-convexity of the similarity constraints and introduce a convex relaxation based on Majorization-Minimization techniques (Section IV).
3. We propose a convex computational efficient algorithm and provide guarantees for its convergence (Section V).

Interesting generalizations of the considered graph learning problem are discussed in Section VI, and then, the effectiveness of the proposed approach is demonstrated in Section VII, followed by brief concluding remarks.

II. Preliminaries: Graphs, GSP and GMRFs

We briefly introduce graph-related and GSP-related notation and review the definition of GMRFs.

Graphs: Let $\mathcal{G} := (\mathcal{V}, \mathcal{E})$ denote an undirected and weighted graph with a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$. The graph is composed of $|\mathcal{V}| = N$ nodes and, for every $i, j \in \mathcal{V}$, we have that $(i, j) \in \mathcal{E}$ if and only if the nodes i and j are directly connected. The neighborhood of any node i represents the set of nodes that are connected to i, i.e., $\mathcal{N}_i := \{j \in \mathcal{V} | (i, j) \in \mathcal{E}\}$. The connectivity of $\mathcal{G}$ is captured in the sparse adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ with $A_{ij} = 0$ only if $(i, j) \notin \mathcal{E}$, and whose entry A_{ij} represents the weight of the edge between nodes i and j.

Graph signals and GSP: Together with the graph $\mathcal{G}$, we consider signals defined on (associated with) $\mathcal{V}$, the nodes of $\mathcal{G}$. Formally, a *graph signal* can be modeled as a function from the vertex set to the real field $x : \mathcal{V} \to \mathbb{R}$ or, equivalently, as an N-dimensional vector $\mathbf{x} \in \mathbb{R}^N$, with x_i denoting the signal value at node i. The last key element in the GSP framework is the so-called *graph-shift operator* (GSO), an $N \times N$ matrix denoted as $\mathbf{S}$ [2]. The GSO, whose entries satisfy that S_{ij} can be non-zero only if $i = j$ or $(i, j) \in \mathcal{E}$, captures the topology of the underlying graph $\mathcal{G}$ and can be understood as a topology-aware local operator that can be applied to process graph signals. Typical choices for the GSO include the adjacency matrix $\mathbf{A}$, the graph combinatorial Laplacian $\mathbf{L} := \text{diag}(\mathbf{A1}) - \mathbf{A}$, and its normalized variants [1], [2]. Note that $\text{diag}(\cdot)$ denotes the diagonal operator that transforms a vector into a diagonal matrix and $\mathbf{1}$ denotes the vector of all ones. Since $\mathcal{G}$ is undirected, it follows that $\mathbf{S}$ is symmetric and it can be diagonalized as $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where the orthonormal matrix $\mathbf{V} \in \mathbb{R}^{N \times N}$ collects the eigenvectors of $\mathbf{S}$, and the diagonal matrix $\mathbf{\Lambda} = \text{diag}(\boldsymbol{\lambda})$ collects the eigenvalues $\boldsymbol{\lambda} \in \mathbb{R}^N$.

GMRF: A multivariate normal distribution is said to form a GMRF with respect to a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if the edges not present in $\mathcal{E}$ correspond to zeros on the precision matrix (the inverse covariance matrix). Upon selecting the GSO $\mathbf{S}$ as the positive definite precision matrix, the previous definition implies that if the random graph signal $\mathbf{x}$ follows a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{S}^{-1})$, then $\mathbf{x}$ is a GMRF with respect to $\mathbf{S}$.

As a result, the probability density function (PDF) of a zero-mean GMRF with GSO $\mathbf{S}$ is simply

$$f_{\mathbf{x}}(\mathbf{x}; \mathbf{S}) = (2\pi)^{-N/2} \det(\mathbf{S})^{1/2} \exp\left(-\frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{S}\mathbf{x}\right). \quad (1)$$

The above expression will be critical to postulate an optimization that learns (estimates) the GSO $\mathbf{S}$ (and, hence, the edge set $\mathcal{E}$) from nodal observations, a key question at the core of Gaussian graphical models [13], [14], [33].

III. Graph Learning From Motif Similarity

Suppose now that we have access to a collection of M graph signals $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_M]$. Each of the M signals collects N measurements (one per node) associated with the nodes of a graph $\mathcal{G}$ that is not known. The graph learning problem aims at using $\mathbf{X} \in \mathbb{R}^{N \times M}$ to estimate the GSO $\mathbf{S} \in \mathbb{R}^{N \times N}$ and, as a result, to identify the *unknown* edge set $\mathcal{E}$ that connects the nodes in the graph $\mathcal{G}$. To render this problem tractable, we consider two main assumptions:

- First, we assume that we have prior knowledge about the local properties of the graph $\mathcal{G}$ and, in particular, on the distribution of its motifs. More specifically, we consider that a reference graph $\tilde{\mathcal{G}}$ with a *density of motifs similar* to that of the unknown graph $\mathcal{G}$ is available. Understanding a graph as a composition of motifs is particularly interesting due to the *local* nature of motifs [34]. Intuitively, assuming that two graphs have a similar density of motifs can be interpreted as assuming that both graphs have common "building blocks" or similar patterns.

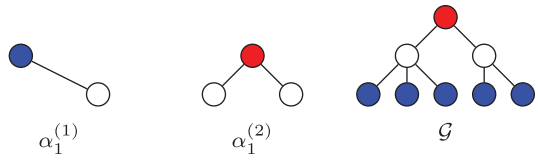


Fig. 1. Two rooted 1-balls $\alpha_1^{(1)}, \alpha_1^{(2)}$ and a graph $\mathcal{G}$. Each node of $\mathcal{G}$ is colored to indicate if the rooted 1-ball centered at that node is isomorphic to $\alpha_1^{(1)}$ (blue), $\alpha_1^{(2)}$ (red), or neither (white). We observe that $\tau_1(\alpha_1^{(1)}, \mathcal{G}) = \frac{5}{8}$ and $\tau_1(\alpha_1^{(2)}, \mathcal{G}) = \frac{1}{8}$.

- The second assumption establishes a relation between the (properties of the) observations in $\mathbf{X}$ and the underlying graph $\mathcal{G}$. In particular, we consider that the columns of $\mathbf{X}$ are (independent) realizations of a GMRF with zero mean and GSO $\mathbf{S}$. While other models relating the graph signals with the unknown supporting graph exist, we focus on GMRF due to its flexibility, solid statistical foundations, and wide adoption within the network science community. Nonetheless, in Section VI we discuss how to generalize our approach to models beyond GMRF.

The goal of this section is to formulate the motif-based graph learning problem rigorously (Problem 1) and postulate an associated constrained optimization problem that leverages the information in $\mathbf{X}$ and the previous assumptions to generate as solution the desired $\mathbf{S}$. To that end, we need to describe in more detail our approach to assess motif similarity (remainder of this section and Section III-A) and then set a formulation combining motif similarity with the GMRF topology estimation framework (Section III-B). The first step is to describe the structural properties of a graph $\mathcal{G}$ in terms of the density of rooted balls, or motifs. A rooted graph is simply a graph with a special labeled node, denoted by a tuple $(\mathcal{G}, \rho)$. If $(\mathcal{G}, \rho)$ is such that each node in $\mathcal{G}$ is in the r-hop neighborhood of the root ρ, we say that it is a rooted r-ball. For a given integer radius $r \geq 0$, a graph $\mathcal{G}$ yields a family of rooted r-balls. For each node $i \in \mathcal{V}$, consider the induced subgraph of the r-hop neighborhood of i. Then, treating i as the root, this yields a rooted r-ball "centered" at i, which we denote as $V_r(\mathcal{G}, i)$.

Then, for a given motif α_r, we define the *rooted motif density* as

$$\tau_r(\alpha_r, \mathcal{G}) = \frac{1}{N} \left| \{ i = 1, ..., N : V_r(\mathcal{G}, i) \cong \alpha_r \} \right|, \qquad (2)$$

where $V_r(\mathcal{G}, i) \cong \alpha_r$ denotes isomorphism of rooted r-balls, i.e., graph isomorphism with the extra condition that the roots coincide. Simply put, the quantity $\tau_r(\alpha_r, \mathcal{G})$ measures the frequency with which a specific motif α_r appears in the graph $\mathcal{G}$ by computing the proportion of rooted r-balls in $\mathcal{G}$ that are isomorphic to α_r. We illustrate this in Fig. 1.

Based on (2), we can compare the similarity between two graphs in terms of their motif densities. With this notation at hand, we formalize the graph learning problem introduced at the beginning of the section.

Problem 1: Let $\mathcal{G}$ be an unknown graph with node set $\mathcal{V}$, $N = |\mathcal{V}|$ and GSO $\mathbf{S} \in \mathbb{R}^{N \times N}$. Furthermore, i) let $\tilde{\mathcal{G}}$ be a reference graph with node set $\tilde{\mathcal{V}}$, $\tilde{N} = |\tilde{\mathcal{V}}|$ and GSO $\tilde{\mathbf{S}} \in \mathbb{R}^{\tilde{N} \times \tilde{N}}$ and ii) let $\mathbf{X} \in \mathbb{R}^{N \times M}$ be a set of M graph signals defined over $\mathcal{G}$. Our goal is to use $\mathbf{X}$ and $\tilde{\mathcal{G}}$ to find the underlying graph structure encoded in $\mathbf{S}$ under the assumptions that:

(AS1a) Graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$ have nodes with degree at most D.

(AS1b) Graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$ present a similar density of motifs, so that $|\tau_r(\alpha_r^{(k)}, \mathcal{G}) - \tau_r(\alpha_r^{(k)}, \tilde{\mathcal{G}})| \leq \epsilon$ for every k, with ϵ being a small positive number, $r \in (0, R]$, and $\{\alpha_r^{(k)}\}_{k=1}^K$ being the set of all (isomorphisms of) rooted r-balls inside the graph $\mathcal{G}$.

(AS2) The columns of $\mathbf{X}$ are M independent realizations of a GMRF with zero mean and GSO $\mathbf{S}$ [cf. (1)].

Examining the proposed motif-related assumptions, we note that **(AS1a)** ensures that there are finitely many possible r-balls on a given graph, which will be used in the derivation presented in the following section. On the other hand, **(AS1b)** provides prior information about the density of motifs of the sought graph based on a structurally similar reference graph. From the definition of rooted motif density in (2), we can observe that $\tau_r(\alpha_r, \mathcal{G})$ is an expectation of the frequency with which the motif α_r appears in the graph $\mathcal{G}$. Moreover, since this expectation is computed locally at each node, **(AS1b)** endows the inference problem with some interesting properties. First, it allows us to compare graphs of different sizes, something that was non-trivial in other works where the graph similarity promoted graphs with similar supports [15], [28]. Also, note that assuming that two graphs have similar densities of motifs is a laxer requirement than assuming they have similar supports. Second, we do not require to know the whole graph $\tilde{\mathcal{G}}$ since we can approximate its associated motif density through a smaller subgraph, so knowing a sampled version of $\tilde{\mathcal{G}}$ suffices.

Unfortunately, despite its attractive properties, the rooted motif density is intrinsically a combinatorial metric that leads to an NP-hard problem when directly incorporated into an optimization framework. In the next section, we present a way to overcome this issue.

A. From Similar Densities of Motifs to Spectral Distributions

We are interested in finding an alternative approach to take advantage of the graph similarity specified in **(AS1b)** without falling into an NP-hard combinatorial problem. To that end, we start by noting that, due to the nature of the GSO, the diagonal entries of $\mathbf{S}^r$ are strictly dictated by the r-balls centered at each node. Furthermore, since $\text{tr}(\mathbf{S}^r) = \text{tr}(\mathbf{\Lambda}^r)$, it seems evident that the density of motifs is closely related to the eigenvalues of the GSO, collected in the $N \times N$ diagonal matrix $\mathbf{\Lambda} = \text{diag}(\boldsymbol{\lambda})$. This suggests that the spectra of two graphs with similar densities of motifs should be similar.

Motivated by the previous discussion, we encode the similar density of motifs between two graphs by means of test functions applied to the spectral distribution of the graphs. Let $\boldsymbol{\lambda} \in \mathbb{R}^N$ denote the vector containing the eigenvalues of $\mathbf{S}$, and denote its associated empirical spectral density function as $\mu_{\boldsymbol{\lambda}}$, with $\mu_{\boldsymbol{\lambda}}(\lambda_i)$ quantifying the multiplicity of the ith eigenvalue normalized by the number of nodes in $\mathbf{S}$. Indeed, $\mu_{\boldsymbol{\lambda}}$ is (formally) a probability distribution on $\mathbb{R}$. Then, for any continuous function, $g : \mathbb{R} \to \mathbb{R}$ compute the Lebesgue integral

$$c_g(\boldsymbol{\lambda}) = \int g(\lambda)\, d\mu_{\boldsymbol{\lambda}}(\lambda) = \frac{1}{N} \sum_{i=1}^{N} g(\lambda_i), \qquad (3)$$

where the last equality follows from $\mathbf{S}$ having a discrete spectrum. With these definitions in place, the following result shows that if $\mathbf{S}$ and $\tilde{\mathbf{S}}$ have similar densities of motifs, then $c_g(\boldsymbol{\lambda})$ and $c_g(\tilde{\boldsymbol{\lambda}})$ are close.

Theorem 1: Let $\boldsymbol{\lambda} \in \mathbb{R}^N$ and $\tilde{\boldsymbol{\lambda}} \in \mathbb{R}^{\tilde{N}}$ denote the eigenvalues of the GSOs of the graphs $\mathcal{G}$ and $\tilde{\mathcal{G}}$. For any continuous test function g, under **(AS1a)** and **(AS1b)**, it follows that

$$|c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})| \leq \delta_\epsilon + \delta_r, \tag{4}$$

where $c_g(\cdot)$ is given in (3), $\delta_\epsilon \geq 0$ is a constant dependent only on g, D, and ϵ such that $\delta_\epsilon \to 0$ as $\epsilon \to 0$, and $\delta_r \geq 0$ is a constant dependent only on g, D, and r such that $\delta_r \to 0$ as $r \to \max\{N, \tilde{N}\}$.

The proof of the theorem is provided in Appendix A. In a nutshell, the proof shows that for any continuous test function g, the quantity $c_g(\boldsymbol{\lambda})$ can be approximated by the expected value of some continuous function depending only on rooted r-balls.

Theorem 1 reduces the similarity of motif densities between two graphs to a comparison of an appropriate test function g applied to their empirical spectral densities. This quantity $c_g(\boldsymbol{\lambda})$ is less expressive than the motif densities $\tau_r(\alpha_r^{(k)}, \mathcal{G})$ in describing the structure of the graph, but it bypasses the combinatorial difficulties in computing the precise motif densities. In the next section, we show that this trade-off is beneficial, as it enables easy integration into network topology inference methods. Moreover, it is worth mentioning that the connection revealed in Theorem 1 can be leveraged in other graph-related problems beyond the graph learning task considered here.

B. Graph Motif-Enhanced Optimization for GMRF Learning

Suppose for now that we ignore the assumptions **(AS1a)** and **(AS1b)**. Leveraging **(AS2)** and the PDF in (1), we have that the likelihood of the joint observation of the M signals in $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_M]$ is $\prod_{m=1}^{M} (2\pi)^{-N/2} \det(\mathbf{S})^{1/2} \exp\left(-\frac{1}{2}\mathbf{x}_m^{\mathrm{T}} \mathbf{S} \mathbf{x}_m\right)$. Upon adopting a maximum likelihood (ML) approach, exploiting the monotonicity of the log function, and using the observations in $\mathbf{X}$ to build the empirical covariance matrix $\hat{\mathbf{C}} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{x}_m \mathbf{x}_m^\top$, the matrix $\mathbf{S}$ can be estimated as

$$\begin{aligned} \min_{\mathbf{S}} \ & \operatorname{tr}(\hat{\mathbf{C}}\mathbf{S}) - \log\det(\mathbf{S}) \\ \text{s.t}: \ & \mathbf{S} \succeq 0, \end{aligned} \tag{5}$$

with the constraint $\mathbf{S} \succeq 0$ guaranteeing that the precision matrix is positive semidefinite and that the $\log\det$ function in the objective is well defined. In the context of GMRF, a widely adopted approach is to augment the objective in (5) with a sparsity promoting regularizer $\lambda\|\mathbf{S}\|_1$, giving rise to the celebrated graphical lasso algorithm [13], [14], [33]. In the previous, $\lambda > 0$ controls the level of sparsity and $\|\mathbf{S}\|_1$ denotes the ℓ_1 norm of the vectorization of the matrix $\mathbf{S}$. On top of augmenting the ML formulation with an ℓ_1 norm, other graph learning approaches incorporate topological conditions by considering a set of feasible GSOs $\mathcal{S}$ and augmenting the formulation in (5) with the constraint $\mathbf{S} \in \mathcal{S}$ [35], [36].

Hence, the key to our approach is to formulate a modified version of the ML estimation in (5) capable of exploiting the availability of the reference graph $\tilde{\mathcal{G}}$ and the results in Theorem 1. More specifically, we encode the fact of $\mathcal{G}$ and $\tilde{\mathcal{G}}$ having similar densities of motifs by leveraging (4) and, as a result, approach Problem 1 through the following non-convex optimization program:

$$\begin{aligned} \min_{\mathbf{S},\mathbf{V},\boldsymbol{\lambda}} \ & \operatorname{tr}(\hat{\mathbf{C}}\mathbf{S}) - \log\det(\operatorname{diag}(\boldsymbol{\lambda})) + \alpha\|\mathbf{S}\|_1 \\ & + \frac{\beta}{2}\|\mathbf{S} - \mathbf{V}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{V}^\top\|_F^2 \\ \text{s.t}: \ & |c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})| \leq \delta, \ \ \mathbf{S} \in \mathcal{S}, \ \ \mathbf{V}^\top\mathbf{V} = \mathbf{I}. \end{aligned} \tag{6}$$

Note that this alternative formulation for learning GMRFs is amenable to constraints involving the spectrum of $\mathbf{S}$. In addition, $\tilde{\boldsymbol{\lambda}}$ is a known constant since it can be obtained from the eigendecomposition of the reference graph $c_g(\tilde{\boldsymbol{\lambda}})$.

We refer to the first constraint in (6) as the similarity constraint because, as stated in Theorem 1, it stems from the assumption that $\mathcal{G}$ and $\tilde{\mathcal{G}}$ have similar motif densities. Intuitively, this constraint promotes desirable properties on the eigenvalues of $\mathbf{S}$ by requiring that evaluating the *empirical spectral distribution* of $\mathbf{S}$ and $\tilde{\mathbf{S}}$ using a common test function g results in a similar value. If we are interested in further reducing the size of the feasible set, it is possible to simultaneously employ several test functions $\{g_j\}_{j=1}^J$ resulting in the associated set of functions $\{c_{g_j}\}_{j=1}^J$. We can trivially modify the program in (6) to include a similarity constraint for each function c_{g_j}. When several constraints are included, we face a trade-off between the improvement in the estimation of $\mathbf{S}$ and the additional complexity of enlarging the set of constraints. In the remainder of the paper, we assume that a single similarity constraint is used, and leave the (optimal) combination of multiple constraints as a future research direction.

The optimization framework introduced in (6) estimates separately the GSO $\mathbf{S}$ from its eigendecomposition $\mathbf{V}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{V}^\top$, including a Frobenius-norm penalty in the objective function to encourage that $\mathbf{S}$ and $\mathbf{V}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{V}^\top$ stay close. Dealing with $\mathbf{V}$ and $\boldsymbol{\lambda}$ as explicitly separated optimization variables allows us to incorporate constraints involving the spectrum of the graph. While this sacrifices convexity, the selected approach is amenable to designing an efficient iterative algorithm, as detailed in Section V. Consideration of graph eigenvalues as explicit optimization variables in the context of graph learning has been explored in, e.g., [8] and [17]. In [8], the eigenvectors were considered to be given. Meanwhile, in [17], they consider that $\mathbf{S} = \mathbf{L}$ and the (convex) spectral constraints are mainly concerned with relatively simple conditions, such as bounding the minimum and maximum value of non-zero elements in $\boldsymbol{\lambda}$ or selecting the number of connected components (number of zero eigenvalues). In contrast, the similarity constraints considered in this paper are more involved and lead to non-convex formulations. Moreover, they stem from the assumption that two graphs have similar motif densities. We explore these differences in more detail through the numerical experiments presented in Section VII.

Capturing more complex prior information about (the spectrum of) $\mathbf{S}$ comes at the cost of employing non-convex constraints. However, since the optimization in (6) was already

non-convex, it does not fundamentally change the complexity of the problem. This is further discussed in the following section, where a convex-approximation approach to handle the similarity constraints is introduced.

IV. Convex Relaxation for the Similarity Constraints

Solving the optimization problem introduced in (6) is a challenge due to its non-convexity, stemming from the bilinear terms involving $\mathbf{V}$ and $\boldsymbol{\lambda}$, the orthogonality of $\mathbf{V}$, and the similarity constraint. The bilinear terms and the orthogonality constraint can be dealt with by implementing an alternating optimization scheme and leveraging results from optimization over manifolds [37], respectively. However, dealing with the *similarity constraint* requires further elaboration.

To analyze the curvature of the similarity constraint, we start by noting that $|c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})| \leq \delta$ is a composition of functions, an operation that is non-convex in general [38]. We also observe that the convexity of c_g is determined by the convexity of the test function g. Then, due to the presence of the absolute value, the similarity constraint will only be convex when the considered test function g is affine.

According to the definition of the function $c_g(\cdot)$ provided in (3), it follows that any affine function $g(x) = ax + b$ with a, $b \in \mathbb{R}$ delimits the same feasible set independently of the values of a and b. Thus, we select the affine function $g(x) = x$, which results in the similarity constraint

$$\left| \frac{1}{N} \sum_{i=1}^{N} \lambda_i - C \right| = \left| \frac{1}{N} \text{tr}(\mathbf{S}) - C \right| \leq \delta, \tag{7}$$

where the constant $C := c_g(\tilde{\boldsymbol{\lambda}})$ encodes the value of the test function evaluated over the known reference graph. A closer inspection reveals that, when $C = 1$ and $\delta = 0$, (7) is equivalent to $\text{tr}(\mathbf{S}) = N$, a common constraint used to fix the scale of the GSO when learning the graph topology [21]. That is to say, the constraint $\text{tr}(\mathbf{S}) = N$ represents a particular case of the similarity constraints put forth in this paper. Moreover, using (7) as a constraint incorporates information about the true scale of the graph, avoiding the scale ambiguity inherent to most network topology inference approaches. Indeed, we observe in Section VII that this general approach reduces the scale ambiguity of the estimated GSO.

Nonetheless, using a linear test function might not be enough to capture more complex relations between the spectral distributions of $\mathbf{S}$ and $\tilde{\mathbf{S}}$. We tackle this issue below by discussing a convex alternative to leverage more general classes of test functions.

A. Convex Relaxation for Convex or Concave Test Functions

Since our goal is to develop a convex relaxation for the similarity constraint defined in (4), we can focus on either convex or concave test functions g without loss of generality. Therefore, we start our discussion by proposing a convex relaxation under the assumption that g is *concave*.

We already discussed that the similarity constraint $|c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})| \leq \delta$ is non-convex due to the composition of the absolute value and the function $c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})$. Then, the first step towards obtaining a convex surrogate consists of decomposing the similarity constraint into the inequalities

$$c_g(\boldsymbol{\lambda}) \leq c_g(\tilde{\boldsymbol{\lambda}}) + \delta \qquad c_g(\boldsymbol{\lambda}) \geq c_g(\tilde{\boldsymbol{\lambda}}) - \delta, \tag{8}$$

where the left and the right constraints are respectively concave and convex due to the concavity of g.

The pair of constraints in (8) determines a feasible set equivalent to the one determined by our original similarity constraint based on the composition of functions. Hence, we replace the optimization problem in (6) with its equivalent form

$$\begin{aligned} \min_{\mathbf{S},\mathbf{V},\boldsymbol{\lambda}} \ & \text{tr}(\hat{\mathbf{C}}\mathbf{S}) - \log\det(\text{diag}\,(\boldsymbol{\lambda})) + \alpha\|\mathbf{S}\|_1 \\ & + \frac{\beta}{2}\|\mathbf{S} - \mathbf{V}\text{diag}(\boldsymbol{\lambda})\mathbf{V}^\top\|_F^2 + \gamma c_g(\boldsymbol{\lambda}) \\ \text{s.t} : \ & c_g(\boldsymbol{\lambda}) \geq c_g(\tilde{\boldsymbol{\lambda}}) - \delta, \ \ \mathbf{S} \in \mathcal{S}, \ \ \mathbf{V}^\top\mathbf{V} = \mathbf{I}. \end{aligned} \tag{9}$$

Here, the key difference is that we kept the convex inequality from (8) as a constraint while the concave inequality is used to augment the objective function. Note that, from the perspective of duality theory, any constraint can be equivalently expressed as a regularization term in the objective function with a non-negative parameter (here denoted as γ) playing the role of the dual variable.

Even though the objective function of (9) is still non-convex due to the presence of convex and concave terms, now the optimization problem can be efficiently solved by an MM approach [39]. Based on the MM framework, we consider an iterative linear upper bound to the function $c_g(\boldsymbol{\lambda})$ leading to a *convex iterative* algorithm that approximates the similarity constraint. Because $c_g(\boldsymbol{\lambda})$ is concave, a suitable upper bound is provided by

$$u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t-1)}) = \nabla c_g(\boldsymbol{\lambda}^{(t-1)})^\top \boldsymbol{\lambda}, \tag{10}$$

which is the first-order approximation of the Taylor series of c_g centered at the solution of the previous iteration $\boldsymbol{\lambda}^{(t-1)}$. Note that we have omitted the terms that do not involve the variable $\boldsymbol{\lambda}$ since they are constants in the optimization problem.

Intuitively, the original non-convex similarity constraint $|c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})| \leq \delta$ ensured that

$$c_g(\boldsymbol{\lambda}) \in [c_g(\tilde{\boldsymbol{\lambda}}) - \delta, \ c_g(\tilde{\boldsymbol{\lambda}}) + \delta] \tag{11}$$

for any feasible $\boldsymbol{\lambda}$. Now, with the proposed convex relaxation based on the MM algorithm, the feasible set is modified as follows. First, the convex constraint $c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}}) \geq \delta$ in (9) ensures that

$$c_g(\boldsymbol{\lambda}) \in [c_g(\tilde{\boldsymbol{\lambda}}) - \delta, \infty]. \tag{12}$$

Then, successively minimizing the upper bound $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t-1)})$ brings the value of $c_g(\boldsymbol{\lambda})$ closer to $c_g(\tilde{\boldsymbol{\lambda}}) - \delta$, the minimum value inside the feasible set. Thus, the value of γ is chosen to promote that $c_g(\tilde{\boldsymbol{\lambda}})$ is inside the interval defined in (11). This process can be interpreted as starting with a loose constraint for the maximum value of $c_g(\tilde{\boldsymbol{\lambda}})$ that gets tightened as the iterative algorithm converges. All the details about the specific implementation of the convex iterative algorithm that solves (9) are provided in Section V.

The last step is to discuss the formulation for *convex* functions g, which lead to a convex $c_g(\boldsymbol{\lambda})$. Using an approach analogous to that for the concave case, from the two constraints in (8), we incorporate the convex one into the graph-related optimization. This entails replacing $c_g(\boldsymbol{\lambda}) \geq c_g(\tilde{\boldsymbol{\lambda}}) - \delta$ with $c_g(\boldsymbol{\lambda}) \leq c_g(\tilde{\boldsymbol{\lambda}}) + \delta$ in (9). Additionally, since for the convex case we are interested in maximizing $c_g(\boldsymbol{\lambda})$, we replace $\gamma c_g(\boldsymbol{\lambda})$ with $-\gamma c_g(\boldsymbol{\lambda})$ in the objective of (9) and employ an MM approach to minimize a linear upper bound of $-\gamma g(\boldsymbol{\lambda})$.

To summarize, following an MM approach, we obtain a convex relaxation for the similarity constraint for every test function g that is differentiable and either convex or concave. Next, we present the specific iterative algorithm that simultaneously deals with the MM relaxation, the bilinear terms, and the orthogonality constraints.

V. Algorithmic Implementation

We solve the network topology inference task presented in Problem 1 by developing an iterative algorithm that solves (9). To that end, we combine an alternating optimization approach that decouples the bilinear terms involving $\boldsymbol{\lambda}$ and $\mathbf{V}$ via MM while incorporating the convex relaxation of the similarity constraint. The resulting algorithm falls into the family of Block Successive Upper bound Minimization (BSUM) [40]. This class of algorithms blend techniques from MM and alternating optimization, and they converge to a stationary point under mild conditions.

Our proposed BSUM algorithm solves (9) by updating the optimization variables $\mathbf{S}$, $\mathbf{V}$, and $\boldsymbol{\lambda}$ in *three* separated *steps*. At each step, we optimize over one of the optimization variables while the rest remain fixed, procuring simpler problems that can be solved efficiently. Then, for a maximum number of T iterations, the following steps are computed at each iteration $t = 0, 1, ..., T$.

Step 1. The first step estimates the block of variables represented by $\mathbf{S}$ while the rest remain fixed. This results in the convex optimization problem given by

$$\begin{aligned} \mathbf{S}^{(t+1)} = \underset{\mathbf{S}}{\operatorname{argmin}} \ & \operatorname{tr}(\hat{\mathbf{C}}\mathbf{S}) + \alpha\|\mathbf{S}\|_1 + \frac{\beta}{2}\|\mathbf{S} - \mathbf{V}^{(t)}\boldsymbol{\Lambda}^{(t)}{\mathbf{V}^{(t)}}^{\top}\|_F^2 \\ \text{s.t} : \ & \mathbf{S} \in \mathcal{S}, \end{aligned} \tag{13}$$

where $\boldsymbol{\Lambda}^{(t)} = \operatorname{diag}(\boldsymbol{\lambda}^{(t)})$. The resulting problem is a combination of linear and (convex) quadratic terms, that can be handled by a number of projected algorithms. First, let $\mathbf{H}$ be a matrix of signed ones matching the sign of the entries of $\mathbf{S}$ such that $\|\mathbf{S}\|_1 = \operatorname{tr}(\mathbf{S}\mathbf{H})$, and hence, $\operatorname{tr}(\hat{\mathbf{C}}\mathbf{S}) + \alpha\|\mathbf{S}\|_1 = \operatorname{tr}(\mathbf{K}\mathbf{S})$, where $\mathbf{K} = \hat{\mathbf{C}} + \mathbf{H}$. Also, define the linear operator $\boldsymbol{\mathcal{S}} : \mathbf{s} \in \mathbb{R}_+^{N(N-1)/2} \rightarrow \boldsymbol{\mathcal{S}}\mathbf{s} \in \mathbb{R}^{N \times N}$ that maps the vector $\mathbf{s}$ into the matrix $\mathbf{S} = \boldsymbol{\mathcal{S}}\mathbf{s}$ satisfying the constraints in $\mathcal{S}$, and denote the adjoint linear operator of $\boldsymbol{\mathcal{S}}$ as $\boldsymbol{\mathcal{S}}^* : \mathbf{Y} \in \mathbb{R}^{N \times N} \rightarrow \boldsymbol{\mathcal{S}}^*\mathbf{Y} \in \mathbb{R}^{N(N-1)/2}$. Then, we efficiently approximate (13) by solving

$$\mathbf{s}^{(t+1)} = \left(\mathbf{s}^{(t)} - \frac{1}{\|\boldsymbol{\mathcal{S}}\|_2^2}(\boldsymbol{\mathcal{S}}^*(\boldsymbol{\mathcal{S}}\mathbf{s}^{(t)}) - \mathbf{z})\right)^+, \tag{14}$$

Algorithm 1: Graph learning from similarity constraints.

Input: $\hat{\mathbf{C}}$, $c_g(\tilde{\boldsymbol{\lambda}})$
Output: $\hat{\mathbf{S}}$.

```
Initialize S^(0), s^(0), λ^(0), and V^(0).
for t = 1 to T do
    Set s^(t+1) as in (14).
    S^(t+1) = S s^(t+1).
    Set V^(t+1) as the eigenvectors of S^(t+1).
    Set λ^(t+1) as the solution to (17).
end
Ŝ = S^(T)
```

where $\mathbf{z} = \boldsymbol{\mathcal{S}}^*(\mathbf{V}^{(t)}\boldsymbol{\Lambda}^{(t)}){\mathbf{V}^{(t)}}^{\top} - \beta^{-1}\mathbf{K})$, $(a)^+ = \max(a, 0)$, and $\|\boldsymbol{\mathcal{S}}\|_2$ denotes the operator norm. Finally, we update $\mathbf{S}^{(t+1)}$ as $\mathbf{S}^{(t+1)} = \boldsymbol{\mathcal{S}}\mathbf{s}^{(t+1)}$.

The derivation of the solution presented in (14) from the initial problem (13) is provided in Appendix B for completeness.

Step 2. The second step estimates the block of variables $\mathbf{V}$ while the others remain fixed. Ignoring the constant terms, the resulting optimization problem is given by

$$\begin{aligned} \mathbf{V}^{(t+1)} = \ & \underset{\mathbf{V}}{\operatorname{argmin}} \ \frac{\beta}{2}\|\mathbf{S}^{(t+1)} - \mathbf{V}\boldsymbol{\Lambda}^{(t)}\mathbf{V}^{\top}\|_F^2 \\ & \text{s.t} : \mathbf{V}^{\top}\mathbf{V} = \mathbf{I}, \end{aligned} \tag{15}$$

which can be equivalently rewritten as

$$\begin{aligned} \mathbf{V}^{(t+1)} = \ & \underset{\mathbf{V}}{\operatorname{argmax}} \ \operatorname{tr}(\mathbf{V}^{\top}\mathbf{S}^{(t+1)}\mathbf{V}\boldsymbol{\Lambda}^{(t)}) \\ & \text{s.t} : \mathbf{V}^{\top}\mathbf{V} = \mathbf{I}. \end{aligned} \tag{16}$$

We note that the orthogonality constraint implies that the optimization variables $\mathbf{V}$ belong to the Stiefel manifold. This is a well-known optimization problem and, as explained in [37, Chapter 4.8], it follows that the solution to (16) is setting $\mathbf{V}^{(t+1)}$ to the eigenvectors of $\mathbf{S}^{(t+1)}$.

Step 3. The last step estimates the block of variables $\boldsymbol{\lambda}$ while the others remain fixed. The resulting optimization problem after ignoring the constant terms can be compactly written as

$$\begin{aligned} \boldsymbol{\lambda}^{(t+1)} = \ & \underset{\boldsymbol{\lambda}}{\operatorname{argmin}} -\sum_{j=1}^{N}\log(\lambda_j) + \frac{\beta}{2}\|\boldsymbol{\lambda} - \hat{\boldsymbol{\lambda}}\|_2^2 + \gamma u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t)}) \\ & \text{s.t} : c_g(\boldsymbol{\lambda}) \geq c_g(\tilde{\boldsymbol{\lambda}}) - \delta, \end{aligned} \tag{17}$$

where $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t)})$ denotes the linear majorization of $c_g(\boldsymbol{\lambda})$ at $\boldsymbol{\lambda}^{(t)}$, and the vector $\hat{\boldsymbol{\lambda}}$ collects the elements on the diagonal of ${\mathbf{V}^{(t+1)}}^{\top}\mathbf{S}^{(t+1)}\mathbf{V}^{(t+1)}$, which are the eigenvalues of $\mathbf{S}^{(t+1)}$. Recall that combining the inequality constraint and the minimization of the upper bound $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t)})$ incorporates the prior information about the distribution of the graph spectrum. Moreover, (17) assumes that the test function g is concave, but, as explained in Section IV, the formulation can be easily modified to account for a convex g.

The overall procedure is summarized in Algorithm 1. Analyzing its computational complexity, we observe that Step 1 requires a moderate number of operations while the complexity

of Step 2 is $\mathcal{O}(N^3)$ because it computes the eigendecomposition of $\mathbf{S}$. Regarding Step 3, directly solving the optimization problem in (17) would result in a computational complexity of $\mathcal{O}(N^{3.5})$. However, because the problem is strictly convex and separable for each optimization variable λ_j, it can be solved efficiently, resulting in a much smaller computational complexity. As a result, the most expensive operation in practice is the eigendecomposition performed in the second step, and hence, the complexity of the overall algorithm is $\mathcal{O}(N^3)$. We stress that this is considerably efficient since learning the graph topology involves $\mathcal{O}(N^2)$ variables. Furthermore, recall that the original setting in Problem 1 involves comparing the density of motifs of two graphs, which is a challenging NP-hard combinatorial problem. Then, regarding the number of iterations, we observed that small values of T are enough for the solution to converge. This is further studied in Fig. 4.

Another key aspect of the proposed BSUM algorithm is its convergence to a stationary point, which is formally stated in the following proposition.

Proposition 1: Let $\mathcal{Y}^*$ denote the set of stationary points of (9). Then, the sequence $(\mathbf{S}^{(t)}, \mathbf{V}^{(t)}, \boldsymbol{\lambda}^{(t)})$ generated by Algorithm 1 converges to a stationary point in $\mathcal{Y}^*$ as $t \to \infty$.

To prove the convergence of our algorithm, we leverage the results in [40] and [41]. To be more specific, conditions under which BSUM algorithms converge to a stationary point were identified in [40, Th. 1b]. However, the original result in [40] did not consider formulations with non-convex constraints, and this is relevant in our setup because the optimization problem in Step 2 includes the non-convex orthogonality constraint $\mathbf{V}^\top \mathbf{V} = \mathbf{I}$. Fortunately, in the context of tensor decompositions, [41] proved that the sequence generated by BSUM algorithms still converges when considering orthogonality constraints like the one in Step 2. As a result, leveraging [41], we can prove the claim in Proposition 1 by showing that our problem satisfies the original conditions identified in [40, Th. 1b]. To be precise, upon denoting the objective function in (9) as $\phi(\mathbf{S}, \mathbf{V}, \boldsymbol{\lambda})$, we have that: (i) the objective functions in (13), (16), and (17) are upper bounds of $\phi(\mathbf{S}, \mathbf{V}, \boldsymbol{\lambda})$[1]; (ii) the level set $\{(\mathbf{S}, \mathbf{V}, \boldsymbol{\lambda}) \,|\phi(\mathbf{S}, \mathbf{V}, \boldsymbol{\lambda}) \leq \phi(\mathbf{S}^{(0)}, \mathbf{V}^{(0)}, \boldsymbol{\lambda}^{(0)})\}$ is compact; (iii) the optimization problems in Step 1 and Step 3 are strictly convex; and (iv) the non-smooth components of $\phi(\mathbf{S}, \mathbf{V}, \boldsymbol{\lambda})$ only involve the variables in $\mathbf{S}$. As a result, the conditions specified in [40, Th. 1b] are met and, invoking [40, Th. 1b] and [41], it follows that the solution of our algorithm converges to a stationary point.

VI. Beyond GMRFs

To simplify exposition and promote clarity, our discussion has been focused on addressing the motif-similarity graph-learning design for the conditions outlined in Problem 1. However, as pointed out at different points of the manuscript, our approach can be used under more general circumstances than those considered so far, including non-Gaussian graphical models. Three generalizations particularly appealing are: (i) having access to more than one reference graph $\tilde{\mathcal{G}}_r$; (ii) having access to the actual spectral density function as $\mu_{\boldsymbol{\lambda}}$ in lieu of $\tilde{\mathcal{G}}$; and (iii) considering more general models than a GMRF to represent the relation between the signals $\mathbf{X}$ and the GSO $\mathbf{S}$. Next, we briefly discuss the modifications to the optimization in (9) required to account for these generalizations.

Starting with the first generalization, let us suppose that we have access to R reference graphs, denoted as $\{\tilde{\mathcal{G}}_r\}_{r=1}^R$. Assuming that the sought graph $\mathcal{G}$ is similar to the graphs in $\{\tilde{\mathcal{G}}_r\}_{r=1}^R$ requires only considering the set of constraints [cf. (8)]

$$c_g(\boldsymbol{\lambda}) \leq c_g(\tilde{\boldsymbol{\lambda}}_r) + \delta_r \qquad c_g(\boldsymbol{\lambda}) \geq c_g(\tilde{\boldsymbol{\lambda}}_r) - \delta_r, \tag{18}$$

for all r. Here, $\tilde{\boldsymbol{\lambda}}_r$ denotes the eigenvalues of the r-th reference graph and the value of δ_r can be selected based on prior information on the similarity between $\mathcal{G}$ and $\tilde{\mathcal{G}}_r$. If such information does not exist, then δ_r is set to δ for all r. Moreover, while all the constraints in (18) can be incorporated into (9), a more prudent approach is to identify first the most restrictive ones and then augment the constraints (objective) of (9) only with those.

We might encounter several reference graphs with similar densities of motifs if, e.g., they are samples drawn from a common random graph model. This leads us to the second generalization, which consists in having access to the desired (true) spectral density function $\mu_{\boldsymbol{\lambda}}$ associated with the random graph model at hand. With an eye on real-world applications, the paper has mostly focused on the case where the prior information on the distribution of motifs comes from a reference graph $\tilde{\mathcal{G}}$ and its empirical spectral density function. However, there may be cases where the actual spectral density function $\mu_{\boldsymbol{\lambda}}$ is known or, alternatively, where promoting some desired properties over the spectral density is of interest. The key to designing graph-learning algorithms that handle the knowledge of $\mu_{\boldsymbol{\lambda}}$ efficiently is to leverage (3), which relates the evaluation of the test functions over the ensemble and the sample distribution. More specifically, it suffices with replacing the sample estimate $c_g(\tilde{\boldsymbol{\lambda}})$ in constraint (8) with the ensemble estimate $\int g(\lambda)\, d\mu_{\boldsymbol{\lambda}}(\lambda)$ computed based on $\mu_{\boldsymbol{\lambda}}$, with no additional changes being required in the optimization.

The third generalization deals with more encompassing models to represent the relation between the observed signals and the sought graph. A meaningful and tractable alternative is to consider that the signals are Gaussian and graph stationary [42], [43]. Basically, a zero-mean random graph signal $\mathbf{x}$ is said to be stationary in a GSO $\mathbf{S}$ if its covariance matrix $\mathbf{C}_x = \mathbb{E}[\mathbf{x}\mathbf{x}^T]$ can be written as a polynomial of $\mathbf{S}$ [43]. Clearly, GMRFs are a particular instance of graph stationary models, since we have that $\mathbf{C}_x = \mathbf{S}^{-1}$. As a result, graph stationarity has been recently used in a number of graph-learning-related problems [8], [23], [44]. For the setup at hand, considering that the signals are both Gaussian and graph stationary implies that the eigenvectors of $\mathbf{S}$ and those of the precision matrix $\boldsymbol{\Theta} \in \mathbb{R}^{N \times N}$ are the same and, as a result, that the product $\mathbf{S}\boldsymbol{\Theta}$ is the same as the product $\boldsymbol{\Theta}\mathbf{S}$. Then, a tractable way to adapt our formulation in (9) to

[1]To be rigorous, when stating that the objective functions of the steps 1, 2 and 3 are upper bounds of $\phi(\mathbf{S}, \mathbf{V}, \boldsymbol{\lambda})$ we are also considering the constant terms omitted in the optimization problems (13), (16), and (17).

deal with stationary GMRF signals is to consider the constraint $\mathbf{S}\boldsymbol{\Theta} = \boldsymbol{\Theta}\mathbf{S}$, which results in the following optimization problem

$$\begin{aligned}\min_{\boldsymbol{\Theta},\mathbf{S},\mathbf{V},\boldsymbol{\lambda}} \; & \operatorname{tr}(\hat{\mathbf{C}}\boldsymbol{\Theta}) - \log\det(\boldsymbol{\Theta}) + \alpha\|\mathbf{S}\|_1 \\ & + \frac{\beta}{2}\|\mathbf{S} - \mathbf{V}\operatorname{diag}(\boldsymbol{\lambda})\mathbf{V}^\top\|_F^2 + \gamma c_g(\boldsymbol{\lambda}) \\ \text{s.t} : \; & c_g(\boldsymbol{\lambda}) \geq c_g(\tilde{\boldsymbol{\lambda}}) - \delta, \;\; \mathbf{S} \in \mathcal{S}, \;\; \mathbf{V}^\top\mathbf{V} = \mathbf{I}, \\ & \boldsymbol{\Theta}\mathbf{S} = \mathbf{S}\boldsymbol{\Theta}. \end{aligned} \tag{19}$$

Intuitively, rather than promoting a sparse precision matrix such that $\boldsymbol{\Theta} = \mathbf{S}$, (19) learns a precision matrix $\boldsymbol{\Theta}$ that is a polynomial of the sparse GSO. This less restrictive assumption results in a more flexible graph-learning algorithm capable of handling a larger range of scenarios. Even though the resulting optimization problem is non-convex, it is amenable to an iterative approach similar to the one presented in Section V, but with an additional step for estimating the new optimization variable $\boldsymbol{\Theta}$.

VII. Numerical Results

We now present numerical experiments to gain intuition about the proposed graph-learning algorithm and to assess its performance. We consider different test functions and compare the results achieved with popular graph-learning algorithms over a range of scenarios. The code implementing the proposed algorithm and the experiments is available on GitHub[2].

Upon proper selection of the test functions, the method proposed in this paper is robust to the graph scale ambiguity. Since, in general, this is not the case for most graph-learning algorithms, to provide a fairer comparison, we set the true GSO $\mathbf{S}^*$ and its estimate $\hat{\mathbf{S}}$ to have unit Frobenius norm before computing the error. The resulting error metric is given by

$$\operatorname{err}(\hat{\mathbf{S}}, \mathbf{S}^*) = \left\| \frac{\hat{\mathbf{S}}}{\|\hat{\mathbf{S}}\|_F} - \frac{\mathbf{S}^*}{\|\mathbf{S}^*\|_F} \right\|_F^2. \tag{20}$$

In addition, in the numerical experiments, we focus on estimating the combinatorial Laplacian $\mathbf{L}$, so we solve the optimization problem in (9) by setting the set of feasible GSOs to $\mathcal{L} := \{L_{ij} \leq 0 \text{ for } i \neq j;\ \mathbf{L} = \mathbf{L}^\top;\ \mathbf{L}\mathbf{1} = \mathbf{0}\}$. While our algorithms work for any type of GSO, most of the literature focuses on learning Laplacians, so setting $\mathbf{S} = \mathbf{L}$ here facilitates the comparisons with the state of the art.

A. Proposed Test Functions

The test functions g are at the core of the similarity constraints proposed in this paper. Hence, before presenting the numerical results, we provide the different test functions considered in the experiments and the associated upper bounds.

Linear test function. Considering $g(x) = x$ results in the similarity constraint (7). Since it involves the $\operatorname{tr}(\mathbf{S})$, we denote it as "Tr" in the experiments. This function renders the similarity constraint convex, so no upper bound is required.

Heat kernel test function. Setting $g(x) = e^{-x}$ results in a convex function c_g with an associated upper bound $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t-1)}) = \frac{1}{N}\sum_{i=1}^N \lambda_i e^{-\lambda^{(t-1)}}$. This is denoted as "Heat" in the experiments.

Square root test function. Setting $g(x) = \sqrt{x}$ results in a concave function c_g with an associated upper bound $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t)})) = \frac{1}{2N}\sum_{i=1}^N \frac{\lambda_i}{\lambda_i^{(t-1)}}$. This is denoted as "Sqrt" in the experiments.

Quadratic test function. Setting $g(x) = x^2$ results in a convex function c_g with an associated upper bound $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t-1)}) = -\frac{2}{N}\sum_{i=1}^N \lambda_i^{(t-1)}\lambda_i$. This is denoted as "Sq" in the experiments.

Band-rejection test function. Setting $g(x) = (x - 1.5)^2/4$ results in a convex function c_g with an associated upper bound given by $u(\boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t-1)}) = \frac{1}{N}\sum_{i=1}^N (0.75 - 0.5\lambda_i^{(t-1)})\lambda_i$. This test function concentrates around small and large values of $\boldsymbol{\lambda}$, resembling a band-rejection filter. This is denoted as "BR" in the experiments.

B. Results on Synthetic Graphs

By using synthetic data, we can test the algorithms in a wider range of settings, facilitating getting insights. In the following experiments, the graph signals $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_M]$ are sampled from a GMRF where the covariance matrix is given by the pseudo-inverse of the true Laplacian denoted as $(\mathbf{L}^*)^\dagger$. The reported error corresponds to the mean error averaged across 100 realizations of random graphs and graph signals.

Test case 1. The first experiment probes how the test functions in Section VII-A influence the spectrum of the estimated graphs. We generate the target graph $\mathcal{G}$ and the reference graph $\tilde{\mathcal{G}}$ as two lattice graphs with 4 neighbors and $N = 200$ and $\tilde{N} = 150$ nodes, respectively. The histograms of their eigenvalues $\boldsymbol{\lambda}$ and $\tilde{\boldsymbol{\lambda}}$ are depicted in Fig. 2(a) and 2(b), where we can observe that the spectra of both graphs are clearly similar. Then, the remaining panels show the spectrum of the estimated GSOs, $\hat{\boldsymbol{\lambda}}$, obtained following Algorithm 1 when no similarity constraint is employed (Fig. 2(c)), as well as for the different test functions. It can be seen that employing any of the selected similarity constraints renders the empirical distribution of $\hat{\boldsymbol{\lambda}}$ closer to the ground truth than not using any constraint. It is also worth noting that "Heat" and "Sqrt" test functions (Fig. 2(e) and 2(f)) properly capture the distribution of low-valued eigenvalues but struggle with high-valued eigenvalues, resulting in longer tails. On the other hand, "BR" and "Sq" test functions (Fig. 2(g) and 2(h)) are better suited for capturing the shape of the distribution associated with medium and large eigenvalues, but are less precise with the smaller ones. This interesting behavior could help in designing specific test functions that efficiently capture the shape of the spectral distribution of the graph, a worth-looking problem that is considered as a future research direction.

In addition to visually comparing the spectral distribution of the estimated graphs, Fig. 3(a) shows the error of the estimated eigenvalues as the number of signal observations M increases. The error is measured as $\operatorname{err}(\hat{\boldsymbol{\lambda}}, \boldsymbol{\lambda}^*)$, where the Frobenius norm is replaced by the ℓ_2 norm of the vectors. Once again, we observe that the worst performance is obtained when no similarity constraint is used ("Unc" in the legend), clearly illustrating the benefit of accounting for the similarity of $\mathcal{G}$ and $\tilde{\mathcal{G}}$ based on their

[2]https://github.com/reysam93/motif_nti

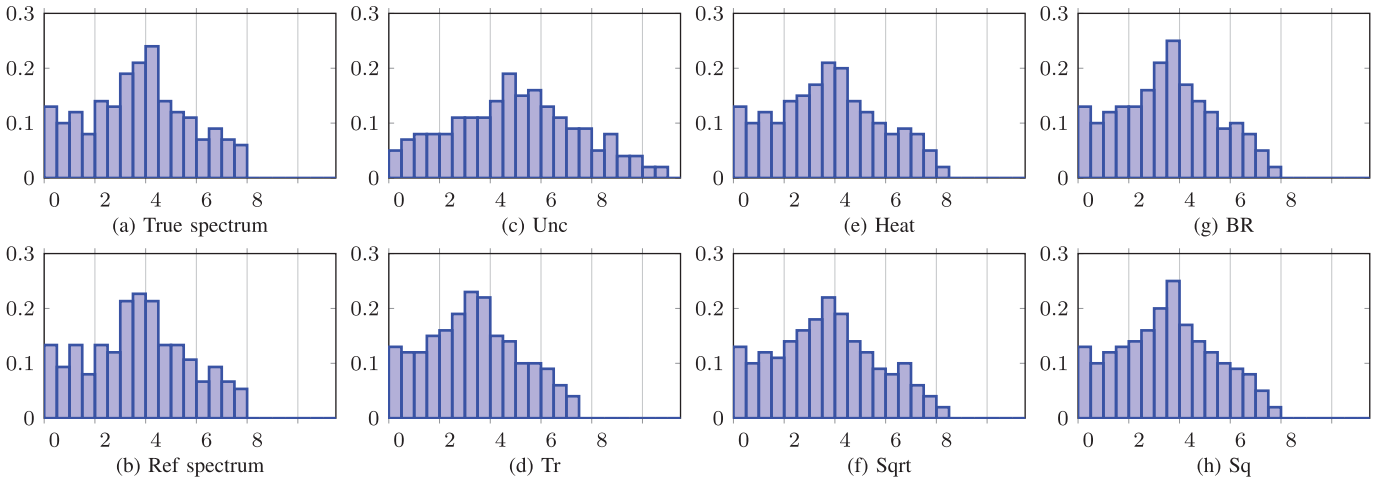


Fig. 2. Histograms representing the empirical spectral distribution of different Laplacian matrices. (a) and (b) Histograms of the true Laplacian $\mathbf{L}^*$ and the reference Laplacian $\tilde{\mathbf{L}}$. (c) Histogram of the estimated $\hat{\mathbf{L}}$ when no similarity constraint is used, and (d)–(h) histogram of $\hat{\mathbf{L}}$ when the considered constraints are linear, heat kernel, square root, band-rejection, and quadratic, respectively.

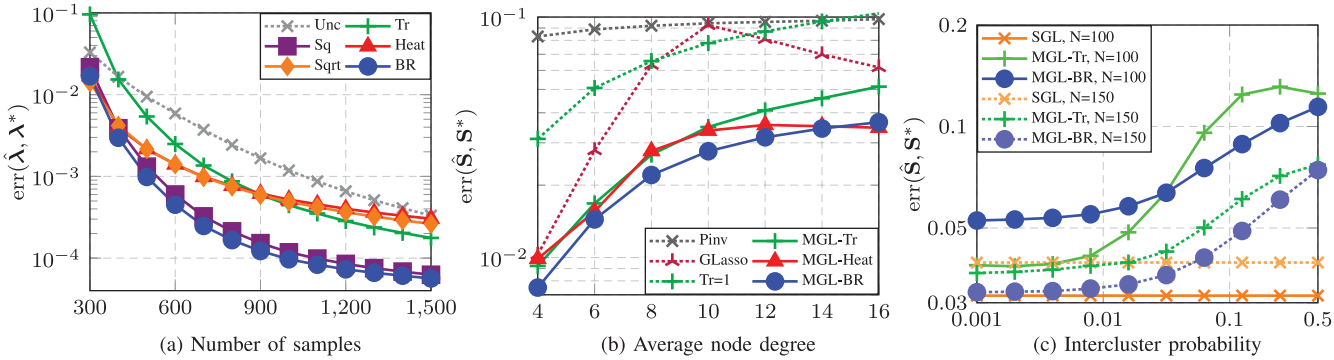


Fig. 3. Mean error of the estimated GSOs when using synthetic data and different types of graphs. (a) $\mathcal{G}$ and $\tilde{\mathcal{G}}$ are generated as lattice graphs with 4 neighbors per node; (b) $\mathcal{G}$ and $\tilde{\mathcal{G}}$ are sampled from a small world model; (c) $\mathcal{G}$ and $\tilde{\mathcal{G}}$ are sampled from an SBM model. All figures show errors averaged over 100 realizations of graphs and signals.

local structures. Furthermore, we observe that the quadratic ("Sq") and band-rejection ("BR") test functions consistently outperform the linear constraint ("Tr"). This supports our previous hypothesis that more sophisticated test functions are more capable of capturing the relationship between the reference and the sought graph.

Test case 2. We continue by evaluating the error of the estimated $\hat{\mathbf{S}}$ when graphs are sampled from the small world (SW) random graph model [45] as edge density increases. True GSOs $\mathbf{S}^*$ have $N = 100$ nodes while the reference graphs have $\tilde{N} = 150$ nodes. In both cases, the number of neighbors of each node increases as reflected in the x-axis of Fig. 3(b). The edge rewiring probability is 0.1, and the number of observations is $M = 1,000$. The results illustrated in Fig. 3(b) compare the performance of our proposed approach with that of the following baselines: (i) "Pinv", which considers the naive solution given by the pseudo-inverse of the sample covariance matrix $\hat{\mathbf{C}}$; (ii) "GLasso", which estimates $\hat{\mathbf{S}}$ by means of the graphical Lasso algorithm [14]; and (iii) "Tr=N", which solves problem (6) replacing the similarity constraint by the fixed constraint $\text{tr}(\mathbf{S}) = N$ employed in [21]. Our graph-learning algorithm based on similar motif densities is denoted as "MGL" followed by an additional label indicating the similarity constraint considered. Looking at the results, we observe that the proposed MGL approach outperforms the other baselines independently of the selected similarity constraint. Of special interest is the comparison between "Tr=N" and "MGL-Tr" since the two constraints are intimately related, as discussed in Section IV. The results show that "MGL-Tr" clearly outperforms "Tr=N", which was expected because the first case employs information about the true value of the $\text{tr}(\mathbf{S})$. Moreover, since the experiments are conducted with $\mathbf{S} = \mathbf{L}$, the value of $\text{tr}(\mathbf{S})$ represents the sum of the degrees across nodes, so the trace constraint can be interpreted as approximately fixing the value of $\|\mathbf{S}\|_1$ to its true value.

Test case 3. This experiment evaluates the robustness of the similarity constraints to discrepancies between $\mathcal{G}$ and $\tilde{\mathcal{G}}$. Furthermore, we compare the performance of the proposed method with the spectral graph learning (SGL) algorithm in [17], which yields state-of-the-art performance when dealing with graphs with multiple connected components. We draw the true $\mathbf{S}^*$ from a stochastic block model (SBM) [46] with $K = 5$ communities and edge probabilities of $p = 0.3$ and $q = 0$ for nodes within

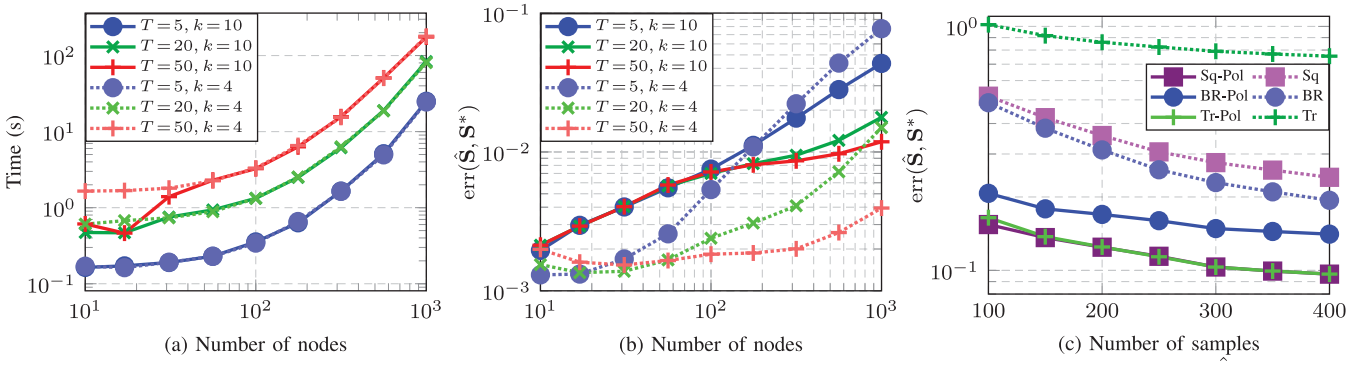


Fig. 4. Evaluation of the MGL algorithm in different settings. (a) and (b) Report the running time and the error of $\hat{\mathbf{S}}$ as the size of SW graphs increases. (c) Compares the performance of the MGL algorithm as described in Algorithm 1 and the alternative model introduced in (19) when the precision matrix of the GMRF is a polynomial of the GSO. Signals are sampled from a GMRF distribution and the reported error is the average over 100 realizations.

the same community and nodes of different communities. That is, the SBM graphs have 5 separate connected components, a setting for which the SGL algorithm is tailored. On the other hand, the reference GSO $\tilde{\mathbf{S}}$ is drawn from an SBM with 150 nodes and the same values of p and K, but the value of q increases progressively as indicated in the x-axis of Fig. 3(c). The number of samples is $M = 1{,}000$ independently of the number of nodes and, as indicated in the legend, we consider the error of the estimated $\hat{\mathbf{S}}$ for graphs with 100 and 150 nodes. In other words, larger values in the x-axis imply that the discrepancy between $\tilde{\mathbf{S}}$ and $\mathbf{S}^*$ increases. Also, note that the error lines associated with the SGL algorithm remain constant since they do not depend on the reference graph.

A first look at the results from Fig. 3(c) reveals that our MGL algorithm is surprisingly robust to the proposed perturbation on the reference graph. The error remains below 0.1 even for values of q that are comparable to the values of p. Indeed, this phenomenon suggests that the similarity constraints are capturing information about the spectrum that goes beyond unveiling the number of zero eigenvalues. Next, focusing on the graphs with 100 nodes (solid lines), the best performance is achieved by the SGL algorithm. This was expected since the spectral constraints of SGL exactly capture the number of disconnected communities. More illuminating are the results of graphs with 150 nodes (dashed lines), where it can be observed that the MGL outperforms the SGL algorithm for the two selected similarity constraints. This change of behavior is caused because the error of SGL increases with the number of nodes, while the error of the MGL decreases as the graph grows. We stress that this behavior is counter-intuitive because the number of samples remains constant independently of the number of nodes, and hence, a higher N should carry a higher error. Nonetheless, the rationale behind this result is as follows. The functions $c_g(\boldsymbol{\lambda})$ described in (3) may be interpreted as estimating the expectation of some test function g *across all the nodes of the graph*, and hence, as the number of nodes increases the estimation of this expectations improves. As a result, the similarity constraint carries more information when the graph has $N = 150$ nodes, compensating the additional error derived from estimating a larger number of edges, and hence resulting in a better estimate. Finally, note that information about the number of zero eigenvalues can be incorporated into our proposed model seamlessly.

Test case 4. The next experiment assesses the performance of the MGL algorithm with moderately large graphs. Fig. 4(a) and 4(b) respectively shows the running time and the error of the estimated GSO as the number of nodes increases. We consider SW graphs with either $k = 4$ or $k = 10$ neighbors and fix the number of observed signals to $M = 10^4$. First, Fig. 4(a) shows that the proposed algorithm takes between 20 and 120 seconds (considering $T = 5$ or $T = 50$ iterations) to estimate graphs with 10^3 nodes, which is a reasonable running time. Then, in Fig. 4(b), we observe that the error is consistently below 0.1, showcasing that small values of T are enough to obtain a valid estimate. More interestingly, focusing on the error of the setting "$T = 50$, $k = 4$", we observe that the quality of the estimated GSO barely degrades when N increases if T is large enough. This is aligned with the results from Test case 3, and it shows that the similarity constraint carries more information when graphs are larger.

Test case 5. The last experiment involving synthetic graphs investigates the benefits of considering a more general model for the graph signals. To that end, we sample the graph signals from a zero-mean multivariate Gaussian distribution whose covariance matrix is given by a polynomial of the GSO (in contrast with previous experiments, where the covariance matrix was the inverse of the GSO). The reference and the target graphs are lattice graphs with 4 neighbors, and 50 and 20 nodes, respectively. Then, Fig. 4(c), compares the performance of Algorithm 1, which assumes that the covariance of the observed signals is $\mathbf{S}^{-1}$, with the model introduced in (19) ("Pol" in the legend), which assumes that the covariance is a polynomial of $\mathbf{S}$. As expected, we observe that Algorithm 1 has a worse performance because the observed signals do not comply with the assumed model. In contrast, the better performance of the "Pol" model showcases the potential benefits of considering more lenient assumptions.

C. Results on Real-World Graphs

We close the numerical experiments by validating our proposed algorithm over two datasets with real-world graphs.

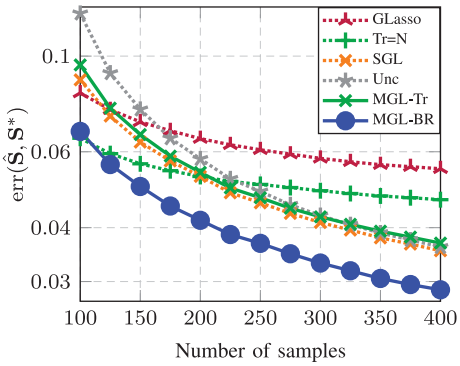


Fig. 5. Error of the estimated GSO using the graphs from the Ljubljana student network dataset. Different baselines are considered and signals are sampled from a GMRF distribution.

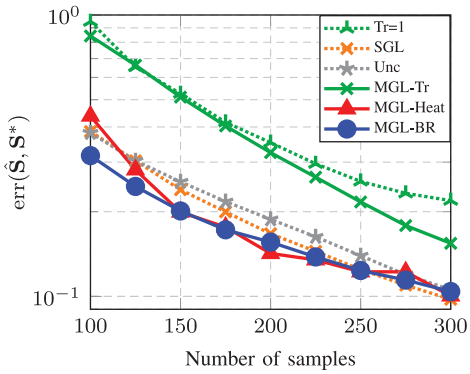


Fig. 6. Error estimating the GSO from the roll-call votes of the U.S. congress dataset. Signals are randomly sampled among all signals, and the error is the average over 100 realizations.

Student network dataset. In this experiment, we consider two graphs with 32 nodes from the Ljubljana student network dataset[3]. In these graphs, nodes represent students from the University of Ljubljana, and the edges of the different networks capture different types of interactions among the students. Because the same students (nodes) are represented across both selected networks, it is expected that the topology of the graphs will be related, allowing us to further assess the value of the method in this paper. This dataset does not contain graph signals, which are created as a GMRF using $(\mathbf{L}^*)^\dagger$ as the covariance. The combination of real graphs and synthetic data brings us the opportunity of evaluating the performance of the MGL algorithm on real graphs while ensuring that the observed signals comply with the assumed model.

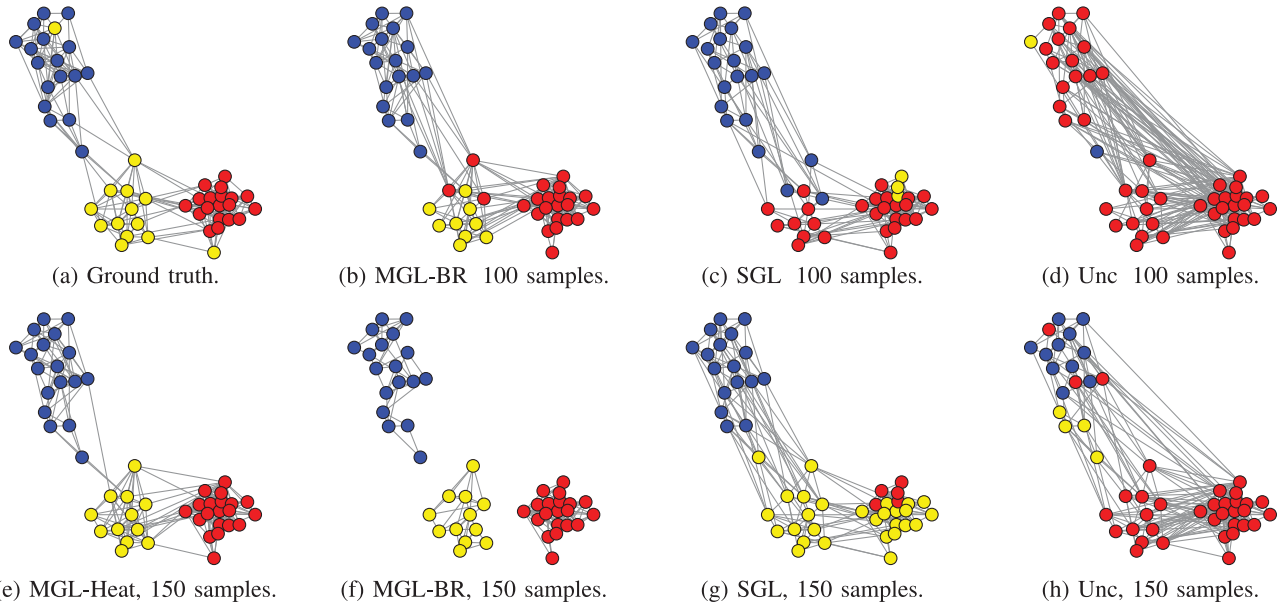
The results are depicted in Fig. 5, where we can observe the error of the estimated graph $\hat{\mathbf{L}}$ as the number of samples increases (represented in the x-axis). It can be seen that the MGL based on the band-rejection test function ("MGL-BR") consistently outperforms the other alternatives. We also note that, for the first values of the number of samples, using the fixed constraint $\mathrm{tr}(\mathbf{S}) = N$ renders a smaller error than using the graph similarity constraint based on the linear test function ("MGL-Tr"). This contrast with the behavior previously observed can be explained because the number of nodes is small ($N = 32$), and hence, as commented in Test case 3, the benefit of the similarity constraints is more limited. Nonetheless, as the number of samples increases, the performance of "MGL-Tr" quickly surpasses that of "Tr=N". We also observe that, for the largest values of M, the errors of "SGL", "Unc", and "MGL-Tr", seem to converge to the same value. We recall that the "Unc" model is a particular implementation of the Laplacian estimation proposed in [16].

Senate votes dataset. Lastly, we consider a dataset containing the roll-call votes of the U.S. Senate [47]. As done in [26], we represent the congresses as networks with 50 nodes (one per U.S. state) that encode the ideological representation of each state. Signals $\mathbf{x}_1, ..., \mathbf{x}_M$ correspond to the votes on different laws and proposals. When voting on a proposal (say the m-th one), we codify the vote of each senator as 1 for a yea, -1 for a nay, and 0 for abstention. We then obtain the value of $\mathbf{x}_m$ for the i-th node as the sum of the votes of the two senators representing the i-th U.S. state, and repeat this process for $i = 1, ..., 50$. The resultant graph signals are categorical and, thus, do not follow the assumption of being sampled from a GMRF. As a result, this experiment will help to illustrate that the MGL algorithm may be employed even when the observed graph signals do not follow a Gaussian distribution.

Modeling the dataset at hand as a dynamic graph, we set the graph corresponding to the 114th congress (years 2015 and 2016) as the known reference graph $\tilde{\mathcal{G}}$, and our goal is to estimate $\mathcal{G}$, the graph corresponding to the 115th congress (years 2017 and 2018). It is worth mentioning that the MGL algorithm is not employing specific information about the number of nonzero eigenvalues. We have access to 499 and 591 observed signals for each of the graphs. Since there are no evident ground-truth graphs, we consider as the true underlying graphs those inferred using the unconstrained solution of problem (6) when all the signals are available. The error of the estimated $\hat{\mathbf{L}}$ is reported in Fig. 6, where the x-axis denotes the M observed signals considered. For low values of M, the "MGL-BR" and "MGL-Heat" outperform the alternatives, even though "Unc" is the algorithm used to generate the ground-truth graph. Moreover, in additional experiments, we observe that considering the median error instead of the mean, the heat test function outperforms the band-rejection test function. Recalling that the heat test function learns small eigenvalues better than the larger ones, the superior performance of the heat test function suggests that in these networks the small eigenvalues play a more fundamental role than in previous settings. On the other hand, as M increases, the error of the different models converges towards the same value, except for "Tr=N" and "MGL-Tr", showing that the trace-based constraints struggle to capture the topological properties of this graph.

To further assess the performance of the proposed algorithm, we employ the estimated graphs to predict the node labels via spectral graph clustering. The results are portrayed in Fig. 7, where we compare the topology and the labels of the true graph (Fig. 7(a)) with the estimates obtained with different graph learning algorithms. The labels (colors) of the nodes represent the ideological representation of each state with red, blue,

[3]The original data can be found at http://vladowiki.fmf.uni-lj.si/doku.php?id=pajek:data:pajek:students



(a) Ground truth. (b) MGL-BR 100 samples. (c) SGL 100 samples. (d) Unc 100 samples.

(e) MGL-Heat, 150 samples. (f) MGL-BR, 150 samples. (g) SGL, 150 samples. (h) Unc, 150 samples.

Fig. 7. Representation of the senate network for the 115th congress. (a) Shows the true graph, (b)–(d) show the estimates obtained with "MGL-BR", "SGL" and "Unc" algorithms when 100 samples are available, while (e)–(h) show the estimates obtained with "MGL-Heat", "MGL-BR", "SGL", and "Unc" when 150 samples are available. The color of the nodes in (a) represents the true node labels, and the colors from (b)–(h) represent the node labels estimated through a spectral graph clustering algorithm considering the graph estimated with each method.

and yellow nodes corresponding to states with two Republican senators, two Democratic senators, and senators from different parties, respectively. When only 100 samples are employed, we observe that using the "MGL-BR" algorithm (Fig. 7(b)) is the alternative that correctly predicts more labels, obtaining an accuracy of 0.9. In contrast, most of the nodes show the same label in the "Unc" solution (Fig. 7(d)), and the "SGL" solution (Fig. 7(c)) misses most of the yellow nodes, resulting in accuracies of 0.4 and 0.66, respectively. When 150 signals are employed, it can be seen that the estimate "MGL-Heat" (Fig. 7(e)) is the alternative that learns more labels while maintaining a single connected component. The resulting accuracy is 0.96, which is also the accuracy obtained when the ground truth graph is employed. Similarly, the "MGL-BR" estimate with 150 samples (Fig. 7(f)) also achieves an accuracy of 0.96, but it segregates the nodes in three connected components, which is useful in this context of node clustering but might be undesirable in other applications. Finally, the "SGL" and "Unc" alternatives have an accuracy of 0.58 and 0.68.

VIII. Conclusion

In this paper, we faced the relevant problem of learning the topology of a graph from a set of GMRF nodal observations. The novel framework proposed herein departs from the maximum likelihood estimator of the sought graph $\mathcal{G}$ and then exploits the assumption that the motif density of a known graph $\tilde{\mathcal{G}}$ is similar to that of $\mathcal{G}$. Indeed, comparing the density of motifs of two graphs is a non-trivial combinatorial task that we addressed by leveraging a relation between the distribution of the spectra of both graphs. More precisely, we showed that, when two graphs have similar motif densities, evaluating a continuous test function over their respective empirical distribution of eigenvalues renders a similar value. This observation was exploited as a constraint in an optimization problem. The resulting similarity constraints were non-convex for most test functions, so we also developed a convex relaxation by proposing an efficient iterative algorithm capable of handling any differentiable convex or concave test function. The proposed algorithm blends techniques from MM algorithms and alternating optimization, it is guaranteed to converge to a stationary point, and its computational complexity is cubic in the number of nodes. Then, we evaluated the proposed algorithm through different numerical experiments involving synthetic and real-world data, where we assessed the influence of several test functions and showed that the proposed algorithm outperforms other popular alternatives. Finally, relying on information about the density of motifs is a promising approach that can be extended to a gamut of applications, a task considered as a future research direction.

Appendix A
Proof of Theorem 1

Let $\mathbf{S}$ and $\tilde{\mathbf{S}}$ denote the GSOs of $\mathcal{G}$ and $\tilde{\mathcal{G}}$, and let $\boldsymbol{\lambda} \in \mathbb{R}^N$ and $\tilde{\boldsymbol{\lambda}} \in \mathbb{R}^{\tilde{N}}$ denote their respective eigenvalues. From **(AS1a)**, it follows that λ_i is contained in a bounded interval of $\mathbb{R}$ for every i, so the spectrum of $\mathbf{S}$ has compact support. The same holds for $\tilde{\mathbf{S}}$. Denote the union of the supports of both empirical spectral densities by A. According to the Stone-Weierstrass theorem [48], any continuous function defined over a compact domain can be approximated arbitrarily and uniformly well by polynomials. That is to say, there is some polynomial of degree r, which we denote by g_r, such that for all $\lambda \in A$, it holds that $|g(\lambda) - g_r(\lambda)| \leq \delta_1$ for some $\delta_1 \geq 0$. Moreover, $\delta_1 \to 0$ as $r \to \infty$. One can then see that

$$|c_g(\boldsymbol{\lambda}) - c_{g_r}(\boldsymbol{\lambda})| \leq \delta_1, \tag{21}$$

with the same bound holding for $\tilde{\boldsymbol{\lambda}}$.

Let $\{\alpha_r^{(k)}\}_{k=1}^K$ be an enumeration of all isomorphism classes of rooted r-balls whose underlying graph satisfies **(AS1a)**. Define the function h on these rooted r-balls so that for each $\alpha_r^{(k)}$, $h(\alpha_r^{(k)})$ yields the diagonal entry at the root of the polynomial g_r applied to the GSO of $\alpha_r^{(k)}$. Since there are only finitely many such rooted r-balls, the magnitude of h is bounded by some constant $C \geq 0$.

Let ρ be a node in $\mathcal{G}$. Then, if the rooted ball $V_r(\mathcal{G}, \rho)$ is isomorphic to $\alpha_r^{(k)}$ for some k, we have that

$$[g_r(\mathbf{S})]_{ii} = h(\alpha_r^{(k)}) = h(V_r(\mathcal{G}, \rho)). \tag{22}$$

Since $\mathcal{G}$ satisfies **(AS1a)**, every rooted r-ball $V_r(\mathcal{G}, \rho)$ satisfies **(AS1a)**, so that we can write

$$c_{g_r}(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{i=1}^{N} h(V_r(\mathcal{G}, i)) = \sum_{k=1}^{K} h(\alpha_r^{(k)})\tau_r(\alpha_r^{(k)}, \mathcal{G}), \tag{23}$$

with a similar equality holding for $c_g(\tilde{\boldsymbol{\lambda}})$ and $\tilde{\mathcal{G}}$. By **(AS1b)**, we have

$$\begin{aligned} |c_{g_r}(\boldsymbol{\lambda}) - c_{g_r}(\tilde{\boldsymbol{\lambda}})| &\leq \sum_{k=1}^{K} |h(\alpha_r^{(k)})| \cdot |\tau_r(\alpha_r^{(k)}, \mathcal{G}) - \tau_r(\alpha_r^{(k)}, \tilde{\mathcal{G}})| \\ &\leq \min\{K, \max\{N, \tilde{N}\}\} \cdot C\epsilon. \end{aligned} \tag{24}$$

We conclude the proof via a simple application of the triangle inequality.

$$\begin{aligned} |c_g(\boldsymbol{\lambda}) - c_g(\tilde{\boldsymbol{\lambda}})| &\leq |c_g(\boldsymbol{\lambda}) - c_{g_r}(\boldsymbol{\lambda})| + |c_{g_r}(\boldsymbol{\lambda}) - c_{g_r}(\tilde{\boldsymbol{\lambda}})| \\ &\quad + |c_{g_r}(\tilde{\boldsymbol{\lambda}}) - c_g(\tilde{\boldsymbol{\lambda}})| \\ &\leq 2\delta_1 + \min\{K, \max\{N, \tilde{N}\}\} \cdot C\epsilon =: \delta. \end{aligned} \tag{25}$$

Appendix B
Efficient Approximation for Step 1

We now provide the details to develop the efficient solution for (13). We start by exploiting the symmetry of the GSO. To that end, recall that $\boldsymbol{\mathcal{S}} : \mathbf{s} \in \mathbb{R}_+^{N(N-1)/2} \to \boldsymbol{\mathcal{S}}\mathbf{s} \in \mathbb{R}^{N\times N}$ denotes the linear operator mapping the non-negative vector $\mathbf{s}$ into the matrix $\mathbf{S} = \boldsymbol{\mathcal{S}}\mathbf{s}$ while ensuring that the constraints in $\mathcal{S}$ are satisfied. Also, recall that $\|\mathbf{S}\|_1 = \text{tr}(\mathbf{SH})$, where $\mathbf{H}$ is an $N \times N$ matrix of signed ones with the sign of its entries matching the sign of the entries of $\mathbf{S}$, so we have that $\text{tr}(\hat{\mathbf{C}}\mathbf{S}) + \alpha\|\mathbf{S}\|_1 = \text{tr}(\mathbf{KS})$, where $\mathbf{K} = \hat{\mathbf{C}} + \mathbf{H}$.

Then, we rewrite the problem in (13) as

$$\begin{aligned} \mathbf{s}^{(t+1)} = \underset{\mathbf{s}}{\text{argmin}}\ &\text{tr}(\mathbf{K}\boldsymbol{\mathcal{S}}\mathbf{s}) + \frac{\beta}{2}\|\boldsymbol{\mathcal{S}}\mathbf{s} - \mathbf{V}^{(t)}\boldsymbol{\Lambda}^{(t)}\mathbf{V}^{(t)^\top}\|_F^2 \\ &\text{s.t} : \mathbf{s} \geq 0, \end{aligned} \tag{26}$$

where the number of optimization variables has been reduced to less than half. Moreover, we denote as $\boldsymbol{\mathcal{S}}^* : \mathbf{Y} \in \mathbb{R}^{N\times N} \to \boldsymbol{\mathcal{S}}^*\mathbf{Y} \in \mathbb{R}^{N(N-1)/2}$ the adjoint linear operator of $\boldsymbol{\mathcal{S}}$ such that $\langle \boldsymbol{\mathcal{S}}\mathbf{s}, \mathbf{Y}\rangle = \langle \mathbf{s}, \boldsymbol{\mathcal{S}}^*\mathbf{Y}\rangle$. Then, we reformulate (26) as the following equivalent quadratic problem

$$\min_{\mathbf{s}\geq 0}\ \frac{1}{2}\|\boldsymbol{\mathcal{S}}\mathbf{s}\|_F^2 - \mathbf{z}^\top\mathbf{s}, \tag{27}$$

with $\mathbf{z} = \boldsymbol{\mathcal{S}}^*(\mathbf{V}^{(t)}\boldsymbol{\Lambda}^{(t)})(\mathbf{V}^{(t)})^\top - \beta^{-1}\mathbf{K})$. Although the problem in (27) is strictly convex, the non-negativity constraint prevents us from obtaining a closed-form solution. To circumvent this issue, we replace the objective function of (27) with an upper bound centered at $\mathbf{s}^{(t)}$, resulting in the optimization

$$\min_{\mathbf{s}\geq 0}\frac{1}{2}\mathbf{s}^\top\mathbf{s} - \mathbf{s}^\top\left(\mathbf{s}^{(t)} - \frac{1}{\|\boldsymbol{\mathcal{S}}\|_2^2}\nabla f(\mathbf{s}^{(t)})\right). \tag{28}$$

The term $\nabla f(\mathbf{s}^{(t)}) = \boldsymbol{\mathcal{S}}^*(\boldsymbol{\mathcal{S}}\mathbf{s}^{(t)}) - \mathbf{z}$ denotes the gradient of the objective function in (27) and $\|\boldsymbol{\mathcal{S}}\|_2^2$ denotes the operator norm given by $\|\boldsymbol{\mathcal{S}}\|_2^2 = \sup_{\|\mathbf{x}\|=1}\|\boldsymbol{\mathcal{S}}\mathbf{x}\|_F^2$.

Finally, the closed-form solution from the KKT optimality conditions of (28) is given by

$$\mathbf{s}^{(t+1)} = \left(\mathbf{s}^{(t)} - \frac{1}{\|\boldsymbol{\mathcal{S}}\|_2^2}\nabla f(\mathbf{s}^{(t)})\right)^+, \tag{29}$$

which is the update for the first step provided in (14).

References

[1] D. Shuman, S. 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 Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

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

[3] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.

[4] P. Djuric and C. Richard, *Cooperative and Graph Signal Processing: Principles and Applications*. Cambridge, MA, USA: Academic Press, 2018.

[5] E. D. Kolaczyk, *Statistical Analysis of Network Data: Methods and Models*. New York, NY, USA: Springer, 2009.

[6] O. Sporns, *Discovering the Human Connectome*. Boston, MA, USA: MIT Press, 2012.

[7] K. Nodop, R. Connolly, and F. Girardi, "The field campaigns of the European tracer experiment (ETEX): Overview and results," *Atmos. Environ.*, vol. 32, no. 24, pp. 4095–4108, 1998.

[8] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 467–483, Sep. 2017.

[9] S. Segarra, A. G. Marques, M. Goyal, and S. Rey, "Network topology inference from input-output diffusion pairs," in *Proc. IEEE Workshop Stat. Signal Process. (SSP)*. Piscataway, NJ, USA: IEEE, 2018, pp. 508–512.

[10] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 16–43, May 2019.

[11] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo, "Graph topology inference based on sparsifying transform learning," *IEEE Trans. Signal Process.*, vol. 67, no. 7, pp. 1712–1727, Apr. 2019.

[12] A. Buciulea, S. Rey, C. Cabrera, and A. G. Marques, "Network reconstruction from graph-stationary signals with hidden variables," in *Proc. 53rd Asilomar Conf. Signals, Syst., Comput.* Piscataway, NJ, USA: IEEE, 2019, pp. 56–60.

[13] S. L. Lauritzen, *Graphical Models*, vol. 17. Oxford, U.K.: Clarendon, 1996.

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

[15] P. Danaher, P. Wang, and D. M. Witten, "The joint graphical lasso for inverse covariance estimation across multiple classes," *J. R. Statist. Soc. Ser. B (Stat. Methodol.)*, vol. 76, no. 2, pp. 373–397, 2014.

[16] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 825–841, Sep. 2017.

[17] S. Kumar, J. Ying, J. Cardoso, and D. Palomar, "Structured graph learning via Laplacian spectral constraints," *Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[18] X. Cai, J. A. Bazerque, and G. B. Giannakis, "Sparse structural equation modeling for inference of gene regulatory networks exploiting genetic perturbations," *PLoS, Comput. Biol.*, Jun. 2013.
[19] B. Baingana, G. Mateos, and G. B. Giannakis, "Proximal-gradient algorithms for tracking cascades over social networks," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 4, pp. 563–575, Aug. 2014.
[20] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Intl. Conf. Artif. Intel. Statist. (AISTATS),* 2016, pp. 920–929.
[21] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning Laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160–6173, Dec. 2016.
[22] S. S. Saboksayr and G. Mateos, "Accelerated graph learning from smooth signals," *IEEE Signal Process. Lett.*, vol. 28, pp. 2192–2196, 2021.
[23] R. Shafipour and G. Mateos, "Online topology inference from streaming stationary graph signals with partial connectivity information," *Algorithms*, vol. 13, no. 9, p. 228, 2020.
[24] T. M. Roddenberry, M. Navarro, and S. Segarra, "Network topology inference with graphon spectral penalties," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Piscataway, NJ, USA: IEEE, 2021, pp. 5390–5394.
[25] A. Buciulea, S. Rey, and A. G. Marques, "Learning graphs from smooth and graph-stationary signals with hidden variables," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 8, pp. 273–287, 2022.
[26] M. Navarro, Y. Wang, A. G. Marques, C. Uhler, and S. Segarra, "Joint inference of multiple graphs from matrix polynomials," *J. Mach. Learn. Res.*, vol. 23, no. 76, pp. 1–35, 2022.
[27] X. Yang, M. Sheng, Y. Yuan, and T. Q. Quek, "Network topology inference from heterogeneous incomplete graph signals," *IEEE Trans. Signal Process.*, vol. 69, pp. 314–327, 2020.
[28] S. Rey, A. Buciulea, M. Navarro, S. Segarra, and A. G. Marques, "Joint inference of multiple graphs with hidden variables from stationary graph signals," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 5817–5821.
[29] C. Cortes, D. Pregibon, and C. Volinsky, "Computational methods for dynamic graphs," *J. Comput. Graphical Statist.*, vol. 12, no. 4, pp. 950–970, 2003.
[30] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proc. 12th ACM SIGKDD Intl. Conf. Knowl. Discovery Data Mining*, 2006, pp. 631–636.
[31] M. Kafai, L. An, and B. Bhanu, "Reference face graph for face recognition," *IEEE Trans. Info. Forensics Secur.*, vol. 9, no. 12, pp. 2132–2143, Dec. 2014.
[32] A. Dilthey, C. Cox, Z. Iqbal, M. R. Nelson, and G. McVean, "Improved genome inference in the MHC using a population reference graph," *Nat. Genet.*, vol. 47, no. 6, pp. 682–688, 2015.
[33] P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu, "High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence," *Electron. J. Statist.*, vol. 5, pp. 935–980, Sep. 2011.
[34] T. M. Roddenberry, F. Gama, R. G. Baraniuk, and S. Segarra, "On local distributions in graph signal processing," *IEEE Trans. Signal Process.*, vol. 70, pp. 5564–5577, 2022.
[35] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graphs," in *Proc. 32nd Annu. Conf. Cogn. Sci. Soc.*, 2010, pp. 778–784.
[36] L. Zhao, Y. Wang, S. Kumar, and D. P. Palomar, "Optimization algorithms for graph Laplacian estimation via ADMM and MM," *IEEE Trans. Signal Process.*, vol. 67, no. 16, pp. 4231–4244, Aug. 2019.
[37] P. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ, USA: Princeton Univ. Press, 2009.
[38] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[39] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 794–816, Feb. 2017.
[40] M. Hong, M. Razaviyayn, Z. Luo, and J. Pang, "A unified algorithmic framework for block-structured optimization involving big data: With applications in machine learning and signal processing," *IEEE Signal Process. Mag.*, vol. 33, no. 1, pp. 57–77, Jan. 2016.
[41] X. Fu, K. Huang, M. Hong, N. D. Sidiropoulos, and A. M. C. So, "Scalable and flexible multiview MAX-VAR canonical correlation analysis," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4150–4165, Aug. 2017.
[42] N. Perraudin and P. Vandergheynst, "Stationary signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3462–3477, Jul. 2017.
[43] A. G. Marques, S. Segarra, G. Leus, and A. Ribeiro, "Stationary graph processes and spectral estimation," *IEEE Trans. Signal Process.*, vol. 65, no. 22, pp. 5911–5926, Nov. 2017.
[44] S. Rey and A. G. Marques, "Robust graph-filter identification with graph denoising regularization," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* Piscataway, NJ, USA: IEEE, 2021, pp. 5300–5304.
[45] M. E. J. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," *Phys. Lett. A*, vol. 263, no. 4–6, pp. 341–346, 1999.
[46] M. Newman, *Networks*. Oxford, U.K.: Oxford Univ. Press, 2018.
[47] J. B. Lewis, K. Poole, H. Rosenthal, A. Boche, A. Rudkin, and L. Sonnet. (2019). "Voteview: Congressional roll-call votes database." Accessed: Jul. 27, 2018. [Online]. Available: https://voteview.com
[48] K. Weierstrass, "Über die analytische darstellbarkeit sogenannter willkürlicher functionen einer reellen veränderlichen," *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, vol. 2, pp. 633–639, 1885.