Multiview Graph Learning With Consensus Graph

Abdullah Karaaslanli and Selin Aviyente, *Senior Member, IEEE*

***Abstract*—Graph topology inference is a significant task in many application domains. Existing approaches are mostly limited to learning a single graph assuming that the observed data is homogeneous. This is problematic because many modern datasets are heterogeneous and involve multiple related graphs, i.e., multiview graphs. Prior work in multiview graph learning ensures the similarity of learned view graphs through pairwise regularization, which has several limitations. First, most of the existing work focuses on the Gaussian Graphical Models (GGM) which learns precision matrices rather than the actual graph structures. Second, these methods do not infer the consensus structure across views, which may be useful in certain applications for summarizing the group level connectivity patterns. Finally, the number of pairwise constraints increases quadratically with the number of views. To address these issues, we propose a consensus graph-based multiview graph model, where each view is assumed to be a perturbed version of an underlying consensus graph. The proposed framework assumes that the observed graph data is smooth over the multiview graph and learns the graph Laplacians. A generalized optimization framework to jointly learn the views and the consensus graph is proposed, where different regularization functions can be incorporated into the formulation based on the structure of the underlying consensus graph and the perturbation model. Experiments with simulated data show that the proposed method has better performance than existing GGM-based methods and requires less run time than pairwise regularization-based methods. The proposed framework is also employed to infer the functional brain connectivity networks of multiple subjects from their electroencephalogram (EEG) recordings, revealing both the consensus structure and the individual variation across subjects.**

***Index Terms*—Multiview graphs, graph inference, graph signal processing.**

I. Introduction

MANY real-world data are represented through the relations between data samples or features, i.e., a graph structure [1]. Although many datasets, including social and traffic networks, come with a known graph that helps in their interpretation, there is still a large number of applications where a graph is not readily available. For instance, functional brain networks [2] and gene regulatory networks [3] are not directly observable. In such cases, inferring the topology of the graph is an essential task to be able to effectively analyze the data.

Received 24 January 2024; revised 5 August 2024 and 2 November 2024; accepted 28 December 2024. Date of publication 8 January 2025; date of current version 25 February 2025. This work was supported by the National Science Foundation under Grant CCF-2312546 and Grant CCF-2211645. The associate editor coordinating the review of this article and approving it for publication was Prof. Xiaowen Dong. *(Corresponding author: Abdullah Karaaslanli.)*

The authors are with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: karaasl1@msu.edu; aviyente@egr.msu.edu).

Digital Object Identifier 10.1109/TSIPN.2025.3527888

In order to address this problem, various *graph learning* techniques have been developed by using observed nodal data, also known as *graph signals* [4], to learn the unknown graph topology. These methods are developed using approaches from different domains including statistical methods [5], graph signal processing (GSP) [6], [7] and recently graph neural networks (GNN) [8]. Statistical methods, such as graphical lasso [9], are usually based on GGMs, where the aim is to learn the precision matrix representing conditional dependencies between nodes. Methodologies using GSP defines the relation between graph signals and graph topology using signal processing concepts, such as stationarity [10], [11], [12], non-stationarity [13] and smoothness [14], [15], [16]. Finally, GNN-based techniques use recent graph convolutional networks and the more general message passing networks for relational inference [8], [17], [18].

Although aforementioned methods have shown to be effective, they are limited to homogeneous datasets, where observed graph signals are assumed to be identically distributed and defined on a single graph. However, in many applications, the data may be heterogeneous and come from multiple related graphs, also known as *multiview graphs*. For example, gene expression measurements are often collected across different cell types each with their own regulatory mechanism [19], [20]. Similarly, neuroimaging data from multiple subjects can be considered as being defined on a multiview graph where each view is a subject's brain connectome [21], [22]. In these situations, the views of the multiview graphs are closely related to each other. Therefore, learning the topology of views jointly by incorporating the relationships among views can improve the performance [23], [24], [25].

In the setting of joint learning, one needs to choose the model that relates the views of the multiview graph. The commonly utilized approach is pairwise similarity, where each pair of views is assumed to be similar [24], [25], [26]. Although this approach is effective in joint learning setup, it requires quadratic number of terms to impose similarity. Moreover, it cannot reveal the common structure shared by views. To this end, an alternative model where each view is the sum of common and individual structures has been proposed [27], [28]. Compared to pairwise similarity, this alternative approach can learn the common structure and the views within the same framework. Moreover, this approach requires linear number of terms to impose similarity across views, which can reduce computational complexity. However, the methods in [27], [28] are developed for GGMs and they learn the precision matrix reflecting conditional dependency rather than a valid graph structure.

In this paper, we introduce a multiview graph learning approach where views are modeled to be generated from a common

graph structure, referred to as *consensus graph*. In consensus-based multiview graph model, each view is generated from an underlying consensus graph through a perturbation function that modifies the input graph to generate a new one. An example where this model can be employed is recovering gene regulatory networks using gene expressions collected from heterogeneous samples, such as different disease states or cell types. In this case, each view may correspond to a perturbation of an underlying unknown graph. We next develop an optimization problem that learns such multiview graphs from a given set of smooth signals. The proposed optimization problem employs a consensus-based regularization where both the views and the consensus graph are learned. The optimization problem is written in a general form that allows utilizing different regularization functions.

The proposed method addresses limitations of the previous work on multiview graph learning. Compared to pairwise similarity-based methods, it can learn the consensus graph and views within a single framework. Although one can learn the consensus graph of views within the pairwise regularization framework by taking the median or mean of the learned views, this is suboptimal. Namely, this does not allow imposing a structure to the consensus graph, while the proposed optimization problem regularizes the learned consensus graph to have desired graph properties. Second, computational complexity of the proposed approach is linear in number of views, while that of pairwise regularization is quadratic. Our approach also addresses the limitations of prior work where a common structure is learned [27], [28]. This work is focused on GGMs, where the aim is to learn precision matrices reflecting partial correlations, which are hard to interpret as graphs. For example, they include negative off-diagonal entries which do not relate to the edges of a graph directly [14]. Compared to this, our work focuses on smooth graph signals and learns a valid graph topology by learning the graph Laplacian, whose off-diagonal entries are associated with the edges and diagonal entries represent node degrees. Moreover, in GGM-based approach, each view is modeled as the sum of common and individual structures where the common structure is not constrained to be a valid graph. Our consensus graph, on the other hand, is modeled as a valid graph structure, which can be regularized to impose different graph properties. This modeling also allows us to relate views to consensus graph through a general graph perturbation mapping rather a simple additive model employed in previous work. Finally, by using smoothness assumption, our framework is not restricted to data from GGMs and can handle other graph signals generated with smooth graph filters.

To summarize, the contributions of our work are:

- Extending consensus-based multiview graph learning framework from GGM to smooth graph signals, where a valid graph topology instead of precision matrices are learned. By using smoothness, our framework is not restricted to GGMs and can handle different smooth graph signal types.
- Extending smoothness-based graph learning to multiview graph setting through consensus-based regularization, where both the view graphs and the consensus graph are learned. This regularization provides a computationally efficient alternative to pairwise regularization-based multiview graph learning frameworks [25], [29].
- Introducing a consensus-based multiview graph model, where each view is generated from an underlying consensus graph through a perturbation function. To accommodate different perturbation functions, a general optimization framework that allows using different regularization functions is developed.

The remainder of the paper is organized as follows. In Section II, a review of related work is provided. Section III provides a background on graphs and graph learning. Section IV describes the proposed multiview graph learning. Section V includes results on different simulated data as well as on functional brain connectivity network inference from EEG recordings of multiple subjects.

II. Related Work

Most previous methods for learning a multiview graph are based on statistical models. These methods extend graphical lasso [9] to joint learning setup, where they learn the precision matrices of multiple related GGMs and use various penalties in the likelihood framework to exploit the common characteristics shared by different views [24], [28], [30], [31], [32], [33]. Most notable work using this approach is the joint graphical lasso [24], where fused or group lasso penalties are used to encourage topological similarity across views. However, these methods are limited by the assumption that the observed graph signals are Gaussian, suffer from increased computational complexity in the case of pairwise penalties and cannot learn the shared structure across views. [27], [28] address the latter by modeling each view as the sum of common and individual components assuming an additive decomposition model for each view. However, they still focus on learning the precision matrices without imposing any graph structure constraints on the individual views or the common structure. Therefore, they do not infer the graph structure but rather learn conditional dependencies, which may not be suitable for interpreting the structure of data in some contexts. These joint learning approaches have been extended to jointly learn multiple graph Laplacian matrices instead of precision matrices [26]. However, this approach is still limited to Gaussian data.

Recently, GSP community has addressed the problem of learning multiview graphs from heterogeneous data. This work can be split into two categories depending on whether one knows the association of the observed signals to the views *a priori*. In the first setup, multiple datasets are given and each dataset is defined on a view [25], [34]. On the other hand, the second setup deals with the mixture of graph signals, where one is given a single dataset and the association of graph signals to the views is not known [35], [36], [37]. The focus of the present paper is the first category. This problem setting has been most widely studied for inferring the topology of time-varying networks [38], [39], [40], where the aim is to learn graphs at multiple time points and to track changes in the graph structure across time.

This problem can be posed as multiview graph learning with a regularization term that promotes pre-specified changes between consecutive graphs. More recently, the problem of multiview graph learning has been formulated with the assumption of graph stationarity [25]. In this formulation, the signals are assumed to be stationary and pairwise similarity between all graphs is used to regularize the optimization. While this formulation has the same goal as the present paper, it is based on stationary graph signals; while we focus on smoothness. Moreover, it does not learn a consensus graph. In [29], the authors propose a multi-view graph learning method based on smoothness assumption. However, it focuses on decentralized optimization by relying on pairwise regularization, which does not allow for the inference of the shared structure across views.

III. Background

A. Notations

In this paper, lowercase and uppercase letters, e.g., n or N, are used to represent scalars. Vectors and matrices are shown as lowercase and uppercase bold letters, e.g., $\mathbf{x}$ and $\mathbf{X}$, respectively. For a vector $\mathbf{x}$, x_i is its ith entry. For a matrix $\mathbf{X}$, we use X_{ij}, $\mathbf{X}_{i\cdot}$ and $\mathbf{X}_{\cdot i}$ to show its ijth entry, ith row and ith column, respectively. Vectors with all entries equal to 1 or 0, and identity matrix are shown as $\mathbf{1}$, $\mathbf{0}$, and $\mathbf{I}$, respectively. $\mathrm{tr}(\cdot)$ and † refer to the trace and the pseudo-inverse of a matrix, respectively. $\odot$ represents Hadamard product. For a vector $\mathbf{x} \in \mathbb{R}^N$, $\mathrm{diag}(\mathbf{x})$ is a diagonal matrix $\mathbf{X} \in \mathbb{R}^{N\times N}$ with $X_{ii} = x_i$. For a matrix $\mathbf{X} \in \mathbb{R}^{N\times N}$, $\mathrm{diag}(\mathbf{X})$ is a vector $\mathbf{x}$ with $x_i = X_{ii}$. The operator $\mathrm{upper}(\cdot)$ takes a symmetric matrix $\mathbf{X} \in \mathbb{R}^{n\times n}$ as input and returns its strictly upper triangular part as a vector $\mathbf{x} \in \mathbb{R}^{n(n-1)/2}$ constructed in row-major order. We define the matrix $\mathbf{S} \in \mathbb{R}^{n\times n(n-1)/2}$ such that $\mathrm{Supper}(\mathbf{X}) = \mathbf{X}\mathbf{1} - \mathrm{diag}(\mathbf{X})$ where $\mathbf{X}$ is a symmetric matrix.

B. Graphs and Graph Signals

An undirected weighted graph is represented as $G = (V, E, \mathbf{W})$ where V is the node set with cardinality $|V| = n$, E is the edge set [1]. $\mathbf{W} \in \mathbb{R}^{n\times n}$ is the adjacency matrix of G, where $W_{ij} = W_{ji}$ is the weight of the edge between nodes i and j and $W_{ij} = W_{ji} = 0$ if there is no edge between nodes i and j. $\mathbf{d} = \mathbf{W}\mathbf{1}$ is the degree vector and $\mathbf{D} = \mathrm{diag}(\mathbf{d})$ is the diagonal degree matrix. The Laplacian matrix of G is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. Its eigendecomposition is $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where columns of $\mathbf{V}$ are eigenvectors and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues with $0 = \Lambda_{11} \leq \Lambda_{22} \leq \cdots \leq \Lambda_{nn}$.

A graph signal defined on G is a function $x : V \to \mathbb{R}$ and can be represented as a vector $\mathbf{x} \in \mathbb{R}^n$ where x_i is the signal value on node i [4]. Eigenvectors and eigenvalues of the Laplacian of G can be used to define the graph Fourier transform (GFT), where small eigenvalues correspond to low frequencies. Thus, the graph Fourier transform of $\mathbf{x}$ is $\hat{\mathbf{x}} = \mathbf{V}^\top\mathbf{x}$ where $\hat{x}_i$ is the Fourier coefficient at the ith frequency component Λ_{ii} [41]. Similar to Fourier transform of temporal signals, GFT can be utilized to define a notion of signal variability over the graph such that $\mathbf{x}$ is a smooth graph signal, i.e., $\mathbf{x}$ has low variation over the graph, if most of the energy of $\hat{\mathbf{x}}$ lies in the low frequency components. The smoothness of $\mathbf{x}$ can then be calculated using the total variation (also called the Dirichlet energy) of $\mathbf{x}$ measured in terms of the spectral density of its Fourier transform as:

$$\mathrm{tr}(\hat{\mathbf{x}}^\top\mathbf{\Lambda}\hat{\mathbf{x}}) = \mathrm{tr}(\mathbf{x}^\top\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\mathbf{x}) = \mathrm{tr}(\mathbf{x}^\top\mathbf{L}\mathbf{x}). \tag{1}$$

The quadratic term $\mathrm{tr}(\mathbf{x}^\top\mathbf{L}\mathbf{x})$ on the right hand side of (1) is equal to $\sum_{i\neq j} W_{ij}(x_i - x_j)^2$, whose smaller values indicate the graph signal is smooth. In particular, for a smooth signal $\mathbf{x}$, signal values on strongly connected nodes, i.e., large W_{ij}, are similar to each other [41].

C. Single Graph Learning

An unknown graph G can be learned from a set of graph signals defined on it based on some assumptions about the relation between the observed graph signals and the underlying graph structure. One such assumption is the smoothness of the observations with respect to G, which can be quantified using total variation defined in (1). Total variation offers a natural criterion to search for the best topology over which the observed signals have the desired smoothness property. This graph learning paradigm based on smoothness is motivated by the fact that smooth signals admit low-pass and sparse representation with respect to the GFT basis [6], [7]. Thus, the graph learning problem can be equivalently viewed as one of finding efficient information-processing transforms for graph signals. Moreover, smoothness is at the foundation of several graph-based statistical learning tasks including nearest neighbors, denoising, semi-supervised learning, and spectral clustering. The success of these methods hinges on the fact that many real-world graph signals are smooth as graphs are constructed based on similarities between nodal attributes, or when the network formation process is driven by mechanisms such as homophily or proximity in some latent space [7].

Dong et al. [14] proposed to learn G by assuming the graph signals are smooth with respect to G. Given $\mathbf{X} \in \mathbb{R}^{n\times p}$ as the data matrix with the columns corresponding to the observed graph signals, G can be learned by minimizing smoothness, defined in (1), with respect to the Laplacian matrix of G:

$$\begin{aligned} \underset{\mathbf{L}}{\text{minimize}} \;& \mathrm{tr}(\mathbf{X}^\top\mathbf{L}\mathbf{X}) + \alpha\,\|\mathbf{L}\|_F^2 \\ \text{s.t.} \quad & \mathbf{L} \in \mathbb{L} \text{ and } \mathrm{tr}(\mathbf{L}) = 2n, \end{aligned} \tag{2}$$

where the first term quantifies the total variation of graph signals and the second term controls the density of the learned graph such that larger values of hyperparameter α result in a denser graph. $\mathbf{L}$ is constrained to be in $\mathbb{L} = \{\mathbf{L} : L_{ij} = L_{ji} \leq 0\ \forall i \neq j,\ \mathbf{L}\mathbf{1} = \mathbf{0}\}$, which is the set of valid Laplacians. The second constraint is added to prevent the trivial solution $\mathbf{L} = \mathbf{0}$.

IV. Multiview Graph Learning

A. Consensus Graph-Based Multiview Graph Modelling

A multiview graph, $\mathcal{G} = \{G^1, \ldots, G^N\}$, is a set of N graphs, where each $G^i = (V, E^i, \mathbf{W}^i)$ is defined on the same node set V with $|V| = n$. While edge sets, E^i's, and associated edge

weights are different from each other, they are closely related to each other in most practical applications. In this work, the relation between views is modeled through a consensus graph $G_0 = (V, E, \mathbf{W})$, where each G^i is assumed to be a perturbed version of G_0 as defined below.

Definition 1 (Consensus Graph-based Multiview Graph Model): Consider a multiview graph, $\mathcal{G} = \{G^1, \ldots, G^N\}$, where views are closely related to each other. $\mathcal{G}$ is said to follow a *consensus graph-based multiview graph model* if G^i is generated from a consensus graph $G_0 = (V, E, \mathbf{W})$ through a mapping q_i, i.e., $G^i = q_i(G_0)$ which perturbs the input graph's topology and/or edge weights to generate a new graph.

This definition provides the most general class of consensus graph based multiview graph model, where no restrictions are placed on the form of q_i. The transformation, q_i, may be linear or nonlinear and can be independent or dependent across views resulting in the same edges being perturbed across views. One commonly used model for q_i is edge removal and addition defined by $\mathbf{W}^i = \mathbf{W} - \mathbf{E}_1^i \odot \mathbf{W} + \mathbf{E}_2^i \odot (\mathbf{1} - \mathbf{W})$ where $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ are symmetric binary perturbation matrices [42]. In this model, edges corresponding to non-zero entries of $\mathbf{E}_1^i$ are removed, while those corresponding to non-zero entries of $\mathbf{E}_2^i$ are added. Based on the definitions of $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$, one can obtain different perturbation models considered in the prior work. For example, the additive model considered in [27], [28] is obtained if $\mathbf{E}_1^i = \mathbf{0}$ and $\mathbf{E}_2^i$ is a sparse binary matrix. If $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ have block structure, views are related to the consensus graph through perturbation of community structure [43]. If $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ are row-sparse, only a couple of nodes' connectivities differ between views [31]. In this paper, we consider the most general form of perturbations without any constraints on $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$. Finally, $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ can be dependent across i, allowing the same edges being added or removed across views.

B. Problem Formulation

Assume that we are given a collection of data matrices, $\mathcal{X} = \{\mathbf{X}^1, \ldots, \mathbf{X}^N\}$. The columns of $\mathbf{X}^i \in \mathbb{R}^{n \times p_i}$ are assumed to be smooth graph signals defined on the view graph G^i of an unknown multiview graph $\mathcal{G}$. We assume that $\mathcal{G}$ follows the consensus graph-based multiview graph model given in Definition 1. Our aim is to learn $\mathcal{G}$ and the consensus graph G_0 from given $\mathcal{X}$. To this end, we propose the following optimization problem:

$$\begin{aligned} \underset{\{\mathbf{L}^i\}_{i=1}^N, \mathbf{L}}{\text{minimize}} \; & \sum_{i=1}^{N} \left\{ \text{tr}(\mathbf{X}^{i\top} \mathbf{L}^i \mathbf{X}^i) + \alpha \left\| \mathbf{L}^i \right\|_F^2 \right\} \\ & + \beta c(\{\mathbf{L}^i - \mathbf{L}\}_{i=1}^N) + \gamma r(\mathbf{L}) \\ \text{s.t.} \quad & \mathbf{L}^i \in \mathbb{L}, \text{tr}(\mathbf{L}^i) = 2n, \forall i, \text{ and } \mathbf{L} \in \mathbb{L}, \end{aligned} \tag{3}$$

where $\mathbf{L}^i$'s and $\mathbf{L}$ are the graph Laplacians of G^i and G_0, respectively. The first sum and constraints are analogous to (2). Since G^i's are generated from G_0 through a perturbation, learned $\mathbf{L}^i$'s are encouraged to be similar to $\mathbf{L}$ through $c(\{\mathbf{L}^i - \mathbf{L}\}_{i=1}^N)$, which penalizes the difference between $\mathbf{L}^i$'s and $\mathbf{L}$. $r(\mathbf{L})$ is added to regularize the learned $\mathbf{L}$ to have the desired properties. Finally, α, β and γ are hyperparameters that scale different terms of the objective function.

The above optimization problem is written in a general form where different regularization terms can be used for $c(\cdot)$ and $r(\cdot)$. $c(\cdot)$ encourages the similarity of learned G^i's and G_0 and its functional form depends on the assumptions made about the perturbation function q_i in Definition 1. If q_i corresponds to a perturbation that is sparse and independent across views, $c(\cdot)$ can be imposed to each $\mathbf{L}^i - \mathbf{L}$ separately and set to a sparsity imposing function, such as ℓ_1-norm [44]. If the perturbations have some dependency across the views, then structured sparsity inducing norms, such as $\ell_{p,q}$-norm [45], can be used to capture this dependency. If q_i modifies G_0 locally, such as only a couple of nodes' edges are perturbed, group sparsity [46] can be imposed where the connection of all nodes except those perturbed are enforced to be the same in G^i and G_0. Similarly, the term $r(\cdot)$ can be chosen based on the desired topological properties of G_0. Edge density of G_0 can be controlled by sparsity promoting functions. G_0 can be enforced to have a community structure by promoting sparsity in its spectrum. Its degree distribution can be regularized, e.g., a log-barrier term applied to diagonal of $\mathbf{L}$ can ensure every node has at least one connection. Along with these choices, various regularization terms proposed in graph learning and joint graph learning literature [23] can be modified to be used for $c(\cdot)$ and $r(\cdot)$. In Section IV-D, we discuss two models for $c(\cdot)$ and $r(\cdot)$, which are employed to illustrate the applicability of the proposed optimization framework. Future work will focus on other choices for $c(\cdot)$ and $r(\cdot)$ and the development of new regularization functions within the proposed optimization framework.

C. Optimization

In order to solve the optimization problem in (3), we first vectorize it such that the upper triangular parts of the graph Laplacians are learned. Let $\mathbf{k}^i = \text{upper}(\mathbf{X}^i \mathbf{X}^{i\top}) \in \mathbb{R}^m$, $\mathbf{d}^i = \text{diag}(\mathbf{X}^i \mathbf{X}^{i\top}) \in \mathbb{R}^n$, $\boldsymbol{\ell}^i = \text{upper}(\mathbf{L}^i) \in \mathbb{R}^m$ and $\boldsymbol{\ell} = \text{upper}(\mathbf{L}) \in \mathbb{R}^m$, where $m = n(n-1)/2$. Also, define functions $c_v(\{\boldsymbol{\ell}^i - \boldsymbol{\ell}\}_{i=1}^N) = c(\{\mathbf{L}^i - \mathbf{L}\}_{i=1}^N)$ and $r_v(\boldsymbol{\ell}) = r(\mathbf{L})$. The problem in (3) can then be vectorized as follows (see Appendix A for details):

$$\begin{aligned} \underset{\{\boldsymbol{\ell}^i\}_{i=1}^N, \boldsymbol{\ell}}{\text{minimize}} \; & \sum_{i=1}^{N} \left\{ (2\mathbf{k}^i - \mathbf{S}^\top \mathbf{d}^i)^\top \boldsymbol{\ell}^i + \alpha \boldsymbol{\ell}^{i\top} (\mathbf{S}^\top \mathbf{S} + 2\mathbf{I}) \boldsymbol{\ell}^i \right\} \\ & + \beta c_v(\{\boldsymbol{\ell}^i - \boldsymbol{\ell}\}_{i=1}^N) + \gamma r_v(\boldsymbol{\ell}) \\ \text{s.t.} \quad & \boldsymbol{\ell}^i \leq 0, \mathbf{1}^\top \boldsymbol{\ell}^i = -n, \forall i, \text{ and } \boldsymbol{\ell} \leq 0, \end{aligned} \tag{4}$$

which can be solved using Alternating Direction Method of Multipliers (ADMM) [47]. Let $f(\boldsymbol{\ell}^i) = (2\mathbf{k}^i - \mathbf{S}^\top \mathbf{d}^i)^\top \boldsymbol{\ell}^i + \alpha \boldsymbol{\ell}^{i\top} (\mathbf{S}^\top \mathbf{S} + 2\mathbf{I}) \boldsymbol{\ell}^i$, and let $\imath_1(\cdot)$ and $\imath_2(\cdot)$ be the indicator functions for the sets $\{\boldsymbol{\ell} \in \mathbb{R}^m | \mathbf{1}^\top \boldsymbol{\ell} = -n\}$ and $\{\boldsymbol{\ell} \in \mathbb{R}^m | \boldsymbol{\ell} \leq 0\}$, respectively. We introduce the slack variables $\mathbf{z} = \boldsymbol{\ell}$, $\mathbf{z}^i = \boldsymbol{\ell}^i$, $\mathbf{v}^i = \mathbf{z}^i - \mathbf{z}$ $\forall i$ and rewrite the problem in its standard

ADMM form as follows:

$$\begin{aligned}\underset{\{\boldsymbol{\ell}^i,\mathbf{z}^i,\mathbf{v}^i\}_{i=1}^N,\boldsymbol{\ell},\mathbf{z}}{\text{minimize}} & \sum_{i=1}^{N}\left\{f(\boldsymbol{\ell}^i)+\imath_1(\boldsymbol{\ell}^i)+\imath_2(\mathbf{z}^i)\right\} \\ & +\beta c_v(\{\mathbf{v}^i\}_{i=1}^N)+\gamma r_v(\boldsymbol{\ell})+\imath_2(\mathbf{z}) \\ \text{s.t.} \quad & \boldsymbol{\ell}^i=\mathbf{z}^i,\mathbf{v}^i=\mathbf{z}^i-\mathbf{z}\ \forall i,\ \text{and}\ \boldsymbol{\ell}=\mathbf{z}, \end{aligned} \tag{5}$$

where the first and last constraints in (4) are imposed onto $\mathbf{z}^i$'s and $\mathbf{z}$, respectively and all constraints are included in the objective function through the indicator functions. The scaled augmented Lagrangian can then be written as:

$$\begin{aligned}\mathcal{L}(\{\boldsymbol{\ell}^i,\mathbf{z}^i,\mathbf{v}^i,\mathbf{y}^i,\mathbf{w}^i\}_{i=1}^N,\boldsymbol{\ell},\mathbf{z},\mathbf{y}) &= \sum_{i=1}^{N}\Bigg\{f(\boldsymbol{\ell}^i)+\imath_1(\boldsymbol{\ell}^i) \\ &+\imath_2(\mathbf{z}^i)+\frac{\rho}{2}\left\|\mathbf{z}^i-\boldsymbol{\ell}^i+\frac{1}{\rho}\mathbf{y}^i\right\|_2^2-\frac{1}{2\rho}\left\|\mathbf{y}^i\right\|_2^2 \\ &+\frac{\rho}{2}\left\|\mathbf{v}^i-\mathbf{z}^i+\mathbf{z}+\frac{1}{\rho}\mathbf{w}^i\right\|_2^2-\frac{1}{2\rho}\left\|\mathbf{w}^i\right\|_2^2\Bigg\}+\beta c_v(\{\mathbf{v}^i\}_{i=1}^N) \\ &+\gamma r_v(\boldsymbol{\ell})+\imath_2(\mathbf{z})+\frac{\rho}{2}\left\|\mathbf{z}-\boldsymbol{\ell}+\frac{1}{\rho}\mathbf{y}\right\|_2^2-\frac{1}{2\rho}\left\|\mathbf{y}\right\|_2^2, \end{aligned} \tag{6}$$

where $\mathbf{y}^i$'s, $\mathbf{w}^i$'s and $\mathbf{y}$ are the Lagrangian multipliers of the equality constraints in (5) and ρ is the penalty parameter. Standard ADMM steps at the kth iteration are:

$$\{\hat{\boldsymbol{\ell}}^i,\hat{\mathbf{v}}^i\}_{i=1}^N,\hat{\boldsymbol{\ell}}=\underset{\{\boldsymbol{\ell}^i,\mathbf{v}^i\}_{i=1}^N,\boldsymbol{\ell}}{\text{argmin}}\ \mathcal{L}(\{\boldsymbol{\ell}^i,\hat{\mathbf{z}}^i,\mathbf{v}^i,\hat{\mathbf{y}}^i,\hat{\mathbf{w}}^i\}_{i=1}^N,\boldsymbol{\ell},\hat{\mathbf{z}},\hat{\mathbf{y}}) \tag{7}$$

$$\{\hat{\mathbf{z}}^i\}_{i=1}^N,\hat{\mathbf{z}}=\underset{\{\mathbf{z}^i\}_{i=1}^N,\mathbf{z}}{\text{argmin}}\ \mathcal{L}(\{\hat{\boldsymbol{\ell}}^i,\mathbf{z}^i,\hat{\mathbf{v}}^i,\hat{\mathbf{y}}^i,\hat{\mathbf{w}}^i\}_{i=1}^N,\hat{\boldsymbol{\ell}},\mathbf{z},\hat{\mathbf{y}}) \tag{8}$$

$$\hat{\mathbf{y}}^i=\hat{\mathbf{y}}^i+\rho(\hat{\mathbf{z}}^i-\hat{\boldsymbol{\ell}}^i),\ \forall i \tag{9}$$

$$\hat{\mathbf{w}}^i=\hat{\hat{\mathbf{w}}}^i+\rho(\hat{\mathbf{v}}^i-\hat{\mathbf{z}}^i+\hat{\mathbf{z}}),\ \forall i \tag{10}$$

$$\hat{\mathbf{y}}=\hat{\hat{\mathbf{y}}}+\rho(\hat{\mathbf{z}}-\hat{\boldsymbol{\ell}}), \tag{11}$$

where variables with $\hat{}$ and $\hat{\hat{}}$ represent the values of those variables at kth and $(k-1)$th iteration, respectively. The problem in (7) is separable across its variables, thus it can be solved with respect to $\boldsymbol{\ell}^i$'s, $\mathbf{v}^i$'s and $\boldsymbol{\ell}$ separately. Its optimization with respect to each $\boldsymbol{\ell}^i$ leads to a quadratic problem with equality constraint, which has a closed form solution derived from the corresponding Karush–Kuhn–Tucker (KKT) conditions. Optimizing (7) with respect to $\mathbf{v}^i$'s results in the proximal operator of c_v. Similarly, solution of (7) with respect to $\boldsymbol{\ell}$ is the proximal operator of r_v. The variables of (8) are coupled, therefore it cannot be solved separately for each $\mathbf{z}^i$ and $\mathbf{z}$. We solve it using block coordinate descent (BCD) as it's a smooth convex problem with separable inequality constraints [48]. The details of the solutions for (7) and (8) are given in Appendix B.

D. Selection of $c(\cdot)$ and $r(\cdot)$

Different forms for $c(\cdot)$ and $r(\cdot)$ can be selected depending on what perturbation model is used to generate G^i's from G_0 and what properties are desired for G_0. In this section, we propose two models in order to show that the proposed optimization problem and its solution are generalizable to different scenarios. For ease of interpretation, the models are defined using the vectorized form of the regularization terms, i.e., $c_v(\cdot)$ and $r_v(\cdot)$.

For the first model, we consider a case where q_i is a sparse perturbation and independent across views. In particular, q_i is considered to be edge addition and deletion, i.e., $\mathbf{W}^i=\mathbf{W}-\mathbf{E}_1^i\odot\mathbf{W}+\mathbf{E}_2^i\odot(\mathbf{1}-\mathbf{W})$, where $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ are sparse binary matrices that are independent across views. The model, referred as mvGL$_{\ell_1}$, is defined as follows:

$$c_v(\{\boldsymbol{\ell}^i-\boldsymbol{\ell}\}_{i=1}^N)=\sum_{i=1}^{N}\left\|\boldsymbol{\ell}^i-\boldsymbol{\ell}\right\|_1 \tag{12}$$

$$r_v(\boldsymbol{\ell})=0, \tag{13}$$

where $\|\cdot\|_1$ is the ℓ_1-norm. Since $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ are assumed to be sparse and independent across views, the model imposes sparsity *separately* to each $\boldsymbol{\ell}^i-\boldsymbol{\ell}$. Independence and sparsity of $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ imply that only a few views can differ from the consensus graph for any given edge, i.e., if $\mathbf{E}_{1,kl}^i\neq 0$ ($\mathbf{E}_{2,kl}^i\neq 0$), then $\mathbf{E}_{1,kl}^j=0$ ($\mathbf{E}_{2,kl}^j=0$) for most $j\neq i$. This can be restrictive in some situations; thus, we define a second model, referred to as mvGL$_{\ell_2}$. In particular, assume $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ are again sparse binary matrices, but they are allowed to be dependent in the sense that if $\mathbf{E}_{1,kl}^i\neq 0$ ($\mathbf{E}_{2,kl}^i\neq 0$), then $\mathbf{E}_{1,kl}^j$ ($\mathbf{E}_{2,kl}^j$) is allowed to be non-zero for $j\neq i$. mvGL$_{\ell_2}$ is defined as:

$$c_v(\{\boldsymbol{\ell}^i-\boldsymbol{\ell}\}_{i=1}^N)=\sum_{a=1}^{m}\sqrt{\sum_{i=1}^{N}(\ell_a^i-\ell_a)^2}=\|\Delta\|_{2,1}, \tag{14}$$

$$r_v(\boldsymbol{\ell})=\|\boldsymbol{\ell}\|_1, \tag{15}$$

where $\Delta\in\mathbb{R}^{m\times N}$ with $\Delta_{\cdot i}=\boldsymbol{\ell}^i-\boldsymbol{\ell}$, $m=n(n-1)/2$ and $\|\cdot\|_{2,1}$ is the $\ell_{2,1}$-norm of the matrix. $\ell_{2,1}$ imposes structural sparsity to Δ where only a few rows are allowed to be non-zero, however non-zero rows are not forced to be sparse, which fits to the modeling of $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ described above. Moreover, $\boldsymbol{\ell}$ is regularized with ℓ_1 norm to impose sparsity. Since without this regularization $\boldsymbol{\ell}$ is equivalent to the element-wise mean of the views, which results in a dense consensus graph. Therefore, we employ ℓ_1 norm regularization to impose sparsity on the learned $\boldsymbol{\ell}$. In mvGL$_{\ell_1}$, $\boldsymbol{\ell}$ is the element-wise median of the views which ensures that the learned $\boldsymbol{\ell}$ has a sparsity level similar to the individual views. Thus, a sparsity imposing regularization is not necessary.

E. Convergence and Computational Complexity

The proposed optimization procedure is derived by rewriting the vectorized problem (4) as in (5) using slack variables. This new form of the problem follows the standard two-block ADMM form. If $c(\cdot)$ and $r(\cdot)$ are convex functions, the proposed optimization is guaranteed to converge (see [47], [49]).

Using this fact, we terminate the optimization if the difference between objective function at consecutive iterations is smaller than a pre-defined value ϵ (10^{-4} is used in our implementation). Moreover, it is known that ADMM can converge to a local minimum for optimization problems with non-convex regularization terms [47], [50]. Therefore, the proposed optimization problem can converge for some non-convex choices of $c(\cdot)$ and $r(\cdot)$. Future work will focus on empirical and theoretical analysis of non-convex regularization for the proposed method.

The computational complexity of the procedure can be calculated by considering the ADMM steps. Namely, the solution of (7) with respect to each $\boldsymbol{\ell}^i$ is $\mathcal{O}(n^2)$ where n is the number of nodes. Complexity of solving (7) with respect to $\mathbf{v}^i$'s and $\boldsymbol{\ell}$ depends on the proximal operator of c_v and r_v. For the two models proposed in Section IV-D, the time complexity is $\mathcal{O}(n^2N)$ where N is the number of views. Each subproblem of the proposed BCD procedure for solving (8) are projections onto the non-positive orthant with time complexity of $\mathcal{O}(n^2)$. Since ADMM converges when steps are solved inexactly, we employed an inexact BCD, where number of iterations is 1. Thus, solving (8) costs $\mathcal{O}(n^2N)$. In conclusion, assuming I is the number of ADMM iterations, the time complexity of our optimization is $\mathcal{O}(n^2NI)$ when c_v and r_v are selected as described in section IV-D.

Remark: The most common approach in multiview graph learning is to use pairwise regularization, where pairs of views are imposed to be similar (see Section II). This approach has higher computational complexity than the proposed approach due to higher number of terms in the objective function. In particular, if the formulation in (3) with consensus regularization, $c(\mathbf{L}^i - \mathbf{L})$, is replaced by pairwise similarity, i.e., $c(\mathbf{L}^i - \mathbf{L}^j)$, the computational complexity of each iteration would become $\mathcal{O}(n^2N^2)$. Thus, while the proposed consensus-based multiview graph learning scales linearly with the number of views, multiview graph learning based on pairwise regularization scales quadratically with the number of views.

V. Results

In this section, the proposed multiview graph learning approach is tested on both simulated data and applied to a real world application of inferring the functional connectivity networks of multiple subjects from EEG recordings.[1]

A. Simulated Datasets

The proposed algorithms are applied to simulated data where $\mathcal{X}$ is generated from multiple related graphs whose structures are known. The learned graphs are then compared to the ground truth graphs to evaluate the performance. The proposed methods are compared against the following algorithms. First, we compare it to single view graph learning (svGL, see (2)), which learns each view independently. Since this method is also based on the assumption of smoothness of $\mathcal{X}$ with respect to the underlying graph, it provides a baseline to compare mvGL against. Second, we study how smoothness-based multiview graph learning compares to widely studied joint graphical lasso modeling. Since our focus is learning Laplacian matrices rather than precision matrices, we benchmark against Laplacian constrained joint graphical lasso (JEMGL) proposed in [26]. We compare against group JEMGL (JEMGL$_{\mathrm{G}}$), where group lasso is employed for regularization, and Laplacian shrinkage JEMGL (JEMGL$_{\mathrm{LS}}$), where pairwise difference between views is employed for regularization. Finally, we implement pairwise versions of our methods, mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ (indicated as mvGL$^{\mathrm{pw}}_{\ell_1}$ and mvGL$^{\mathrm{pw}}_{\ell_2}$), where we change $c_v(\boldsymbol{\ell}^i - \boldsymbol{\ell})$ as $c_v(\boldsymbol{\ell}^i - \boldsymbol{\ell}^j)$. The proposed consensus-based method is then compared to the pairwise version to investigate the difference between consensus and pairwise regularization for multiview graph learning. Since none of the methods considered for comparison learn a consensus graph, we compute the mean and median of the learned views for each method and consider them as the corresponding "consensus graph" for comparison against our learned consensus graph. In particular, for svGL, JEMGL$_{\mathrm{G}}$ and JEMGL$_{\mathrm{LS}}$, the median of the learned views is reported as their consensus graph.[2] Based on the discussion in Section IV-D, the median is used for mvGL$^{\mathrm{pw}}_{\ell_1}$ and the mean is used for mvGL$^{\mathrm{pw}}_{\ell_2}$.

Simulated data are generated from known graphs that are constructed as follows. We first generate a consensus graph G_0 with n nodes from different random graph models, such as Erdős–Rényi (ER) and Barabási–Albert (BA). Next, N view graphs are generated, where each G^i is generated from G_0 as described in Section IV-A where q_i is selected edge removal and addition model. Given the view graphs, each $\mathbf{X}^i \in \mathbb{R}^{n\times p}$ in $\mathcal{X}$ is generated from G^i using the smooth graph filter $h(\mathbf{L}^i)$. Namely, each column of $\mathbf{X}^i$ is generated as $\mathbf{X}^i_{\cdot j} = h(\mathbf{L}^i)\mathrm{x}_0$; where $\mathrm{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. We finally add $\eta\%$ noise (in ℓ_2 norm sense) to each generated $\mathbf{X}^i$. For each simulation, average performance over 50 realizations is reported.

In most experiments, we employ F1 score to measure the performance of methods. F1 score measures how well the topology of learned graphs matches the ground truth graph. For views, F1 score is calculated for each one separately and the average is reported. For the consensus graph, the learned topology is compared to the ground truth to calculate the consensus F1. In order to ensure that the comparison is valid across different performance metrics, Section V-A3 benchmarks the methods using additional metrics: area under precision and recall curve (AUPRC), accuracy and normalized mutual information (NMI). AUPRC quantifies the performance by comparing learned edge weights to the ground truth topology such that it is higher if large weights are learned for true edges. Both Accuracy and NMI compare the topology of the learned graph to the ground truth and higher values indicate better performance.

All methods require the selection of hyperparameters that weigh the different terms in the loss functions. For svGL, mvGL$_{\ell_1}$,mvGL$_{\ell_2}$, mvGL$^{\mathrm{pw}}_{\ell_1}$, and mvGL$^{\mathrm{pw}}_{\ell_2}$; α determines the sparsity of the learned views. For mvGL$_{\ell_2}$, γ determines the sparsity of the learned consensus graph. mvGL$_{\ell_1}$, mvGL$_{\ell_2}$,

[1]Codes for the proposed methods and all experiment scripts can be found at https://github.com/SPLab-aviyente/mvgl.

[2]The mean of the views is observed to not perform as well as the median in studied simulations.

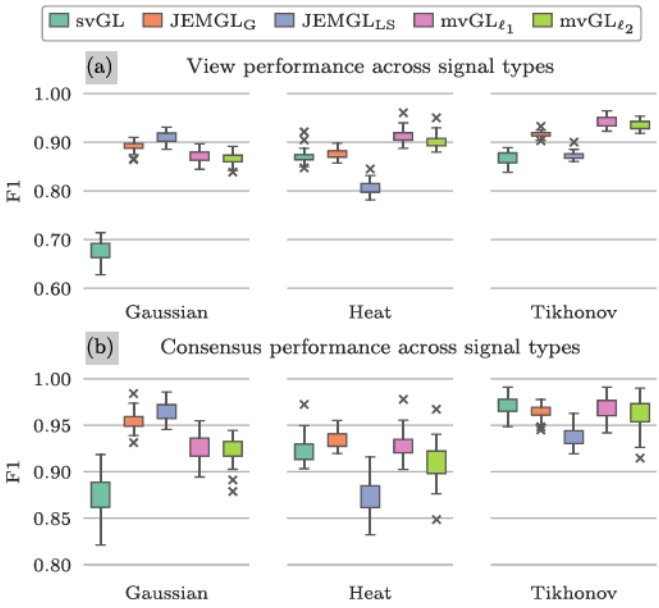

Fig. 1. Comparison of the proposed methods to two JEMGL models across different smooth graph signals types. Performance of svGL is also shown as baseline.

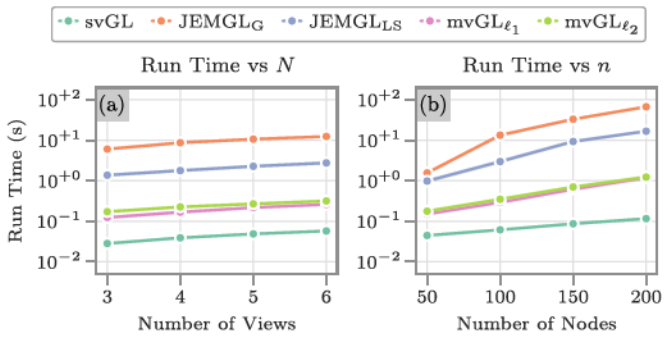

Fig. 2. Comparison of the proposed methods to two JEMGL models in terms of time complexity. Performance of svGL is shown as baseline.

mvGL$^{\text{pw}}_{\ell_1}$ and mvGL$^{\text{pw}}_{\ell_2}$ also require the selection of β controlling the similarity of the learned views. We perform a grid search on these hyperparameters and use the value that provides the best performance. JEMGL$_G$ and JEMGL$_{LS}$ include two hyperparameters ρ_n and ρ_1. Compared to mvGL, these parameters do not have a one-to-one relation with the learned graphs' properties, i.e., density and similarity across views. Therefore, the approach proposed in the original work is followed. Namely, we set $\rho_1 = 1$ and ρ_n is selected from the set $\{10^{-2+2r/15} \mid r = 0, \ldots, 20\}$ as the value maximizing the performance. However, with this setup, the learned graphs are observed to be not sparse leading to very poor performance.[3] Thus, we threshold the learned graphs by removing edges whose weights fall into the bottom τ quantile of all edge weights, where τ is set to the value providing the best performance.

1) Comparison to JEMGL: In this experiment, the proposed methods are compared to JEMGL on datasets where G_0 is generated using ER model with edge probability set to 0.1. $N = 6$ views are generated independently from G_0. Namely, each view's adjacency matrix is $\mathbf{W}^i = \mathbf{W} - \mathbf{E}^i_1 \odot \mathbf{W} + \mathbf{E}^i_2 \odot (1 - \mathbf{W})$ where $\mathbf{E}^i_1$ removes 10% of edges from $\mathbf{W}$ and $\mathbf{E}^i_2$ added the same amount of new edges to $\mathbf{W}$. The remaining parameters for the data generation process are $n = 100, p = 500$ and $\eta = 10$. Graph signals are generated using different graph filters $h(\mathbf{L}^i)$ to observe how JEMGL and mvGL compare across different types of graph signals.

Fig. 1 shows the view and the consensus graph learning performance of the methods. svGL performance is also included as baseline. For both learning the individual views and consensus graphs, JEMGL models perform better than the proposed methods when the data is generated using a Gaussian filter ($h(\mathbf{L}) = \mathbf{L}^\dagger$). However, JEMGL$_G$ and JEMGL$_{LS}$ has worse view performance when heat filter ($h(\mathbf{L}) = \exp(-\alpha\mathbf{L})$, $\alpha = 5$ in this experiment) and Tikhonov filter ($h(\mathbf{L}) = (\mathbf{I} + \alpha\mathbf{L})^{-1}$, $\alpha = 10$ in this experiment) are used. In some cases, its performance is even worse than svGL. In terms of consensus performance, JEMGL$_{LS}$ is worse than the proposed methods for heat and Tikhonov filters, while JEMGL$_G$ has comparable results. Note that, in JEMGL, graph signals are modeled using a Gaussian graphical model whose precision matrix is equal to the graph Laplacian, which is the case when $h(\mathbf{L}) = \mathbf{L}^\dagger$. Therefore, while JEMGL performs well when the underlying assumptions are valid, it does not generalize well to other graph signal models. On the other hand, the proposed methods can perform well across different graph filters, since it uses the more general assumption of smoothness rather than Gaussian graphical models. Finally, performance of svGL is overall lower than the proposed approach, indicating the importance of data sharing across views when learning multiview graphs. Interestingly, svGL has a similar performance to mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ for the consensus graph when signals are generated with Tikhonov filter; however, its view performance is still worse.

The proposed methods are also compared to the two JEMGL methods in terms of their time complexity. Datasets are generated as above with the graph filter set to the Gaussian filter. Fig. 2 shows the time complexity of the methods as a function of number of views (N) and number of nodes (n). In both cases, svGL is the fastest method, which is expected as it infers each view separately without any regularization making it an easier optimization problem. The proposed methods are much faster than JEMGL models. The figure also shows that the different methods' time complexity increases at a similar rate with number of views. On the other hand, rate of increase with respect to the number of nodes is higher for JEMGL compared to the proposed methods. This is due to the fact that JEMGL's time complexity is cubic in the number of nodes (similar to most Gaussian graphical models), while mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ are both quadratic in the number of nodes.

2) Comparison to mvGL$^{\text{pw}}$: In this experiment, the proposed consensus-based methods are compared to their pairwise counterparts in order to observe advantages of consensus-based regularization over widely employed pairwise regularization.

[3]This is because of the fact that sparsity is imposed through ℓ_1-norm regularization in JEMGL. However, it is known that ℓ_1-norm can fail to impose sparsity in graphical lasso framework when a Laplacian matrix is learned [51].

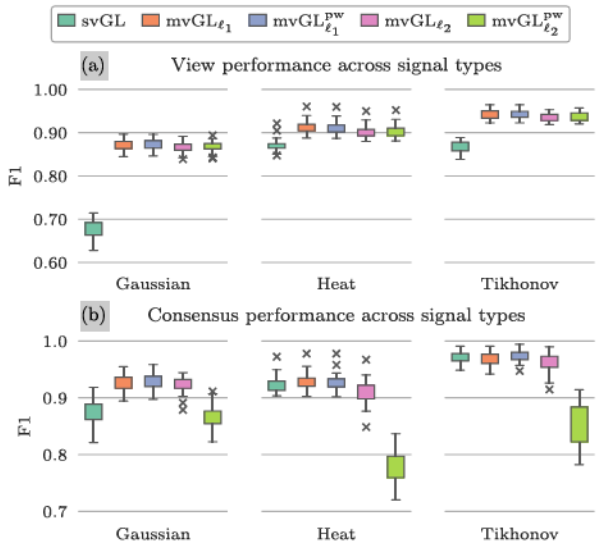

Fig. 3. Comparison of the proposed methods to their pairwise version across different smooth graph signals types. Performance of svGL is shown as baseline.

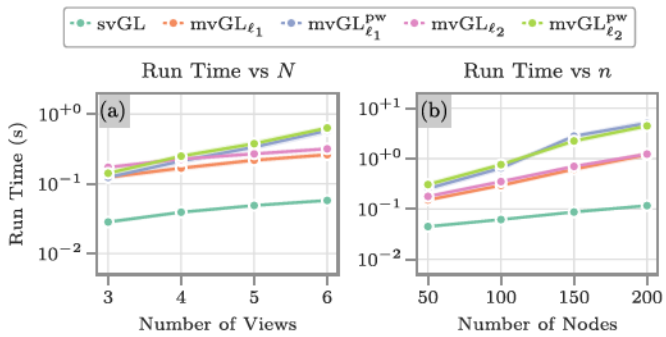

Fig. 4. Comparison of the proposed methods to their pairwise version in terms of time complexity. Performance of svGL is shown as baseline.

The experiment setup is the same as the previous experiment. Fig. 3 reports the performance of learning views and consensus graphs across different graph filters. In terms of learning views, both regularization types perform very similar to each other and they both perform better than svGL as expected. mvGL$_{\ell_1}$ and mvGL$^{\text{pw}}_{\ell_1}$ are also observed to perform slightly better than ℓ_2-norm-based regularization. In terms of consensus performance, mvGL$_{\ell_1}$ and mvGL$^{\text{pw}}_{\ell_1}$ has very similar performance. However, mvGL$^{\text{pw}}_{\ell_2}$ has worse performance than its consensus-based counterpart mvGL$_{\ell_2}$. The reason for this is that the regularization term $r(\cdot)$ in mvGL$_{\ell_2}$ allows one to impose sparsity to the learned consensus graph, while this is not possible for mvGL$^{\text{pw}}_{\ell_2}$ leading to a poor learning performance.

Second, two regularization types are compared based on their time complexity. Fig. 4(a) reports run time of the different methods as a function of number of views. For $N = 3$, both regularization types have similar run time. Pairwise regularization requires more run time than consensus-based regularization as N increases, since it has a time complexity that is quadratic in N, while the proposed method is linear in N. Fig. 4(b) shows the run time as a function of number of nodes. The proposed method runs faster than pairwise regularization for all considered values of n. While both regularization types have time complexity that is quadratic in number of nodes, we observe that the difference between run times of mvGL and mvGL$^{\text{PW}}$ increase with number of nodes. This could be due to the difference between convergence rates of the optimization procedure. Namely, both mvGL and mvGL$^{\text{PW}}$ are solved with the same ADMM-based optimization. While mvGL has N $c(\cdot)$ terms, mvGL$^{\text{pw}}_{\ell_1}$ has $N(N-1)/2$ $c(\cdot)$ terms to impose similarity across views. Higher number of terms in mvGL$^{\text{PW}}$ may increase the number of iterations required for ADMM-based optimization to converge.

In conclusion, although the performance of pairwise and consensus-based regularization is very similar, consensus-based regularization requires less run time, which makes it more preferable especially for large number of views. Moreover, we observe some improvements in learning consensus graph when ℓ_2 regularization is used, since the proposed framework can impose more structure to the learned consensus graph.

3) Comparison Across Performance Metrics: In order to ensure that the previous discussion is valid across different performance metrics, methods are compared using AUPRC, accuracy and NMI. The experiment setup is the same as previous experiments and Fig. 5 shows the results. The view performance results are consistent with the previous discussion. Namely, JEMGL$_{\text{G}}$ and JEMGL$_{\text{LS}}$ have the highest AUPRC, Accuracy and NMI values when data is generated with the Gaussian filter. For heat and Tikhonov, they perform worse than mvGL$_{\ell_1}$, mvGL$_{\ell_2}$ and their pairwise counterparts, indicating the importance of smoothness assumption. Both mvGL and mvGL$^{\text{PW}}$ perform better than svGL in terms of all metrics. Although there are cases mvGL$_{\ell_1}$ is better than mvGL$^{\text{pw}}_{\ell_1}$ and vice versa, their overall performance is almost the same. Similar to before, ℓ_1-norm is also observed to perform better than ℓ_2-norm.

Results for learning the consensus graph are also consistent with the observations in the previous two experiments. For Gaussian filter, svGL is the worst performing method, while JEMGL$_{\text{G}}$ and JEMGL$_{\text{LS}}$ are the best ones. As in Section V-A1, JEMGL$_{\text{G}}$ has comparable results as mvGL$_{\ell_1}$ for heat and Tikhonov filters, while JEMGL$_{\text{LS}}$ does not perform well. In terms of NMI and accuracy, comparison of consensus and pairwise regularization is the same as Section V-A2. For AUPRC, we observe a different result: mvGL$_{\ell_2}$ and mvGL$^{\text{pw}}_{\ell_2}$ perform similarly and they are better than mvGL$_{\ell_1}$ and mvGL$^{\text{pw}}_{\ell_1}$. This is due to the way AUPRC measures the performance. As mentioned, AUPRC compares edge weights to ground truth and does not consider the sparsity of the learned graph. Thus, we observe that mvGL$_{\ell_2}$ and mvGL$^{\text{pw}}_{\ell_2}$ learn better edge weights than mvGL$_{\ell_1}$ and mvGL$^{\text{pw}}_{\ell_1}$. However, they are worse in terms of capturing sparsity.

4) Sensitivity to Data Parameters: We study the effect of the different data parameters, such as the number of views or signals, on the performance of mvGL. The experiment is conducted on a dataset where views are independently generated from G_0 as before where $r\%$ of G_0's edges are removed and the same amount of new edges are added. We evaluate the performance of mvGL with respect to varying number of views N, number of signals p, amount of noise η, perturbation amount r and graph model used to generate G_0. The remaining parameters are $h(\mathbf{L}) = \mathbf{L}^{\dagger}$ and $n = 100$.

	Gaussian				Heat				Tikhonov			
	F1	AUPRC	Accuracy	NMI	F1	AUPRC	Accuracy	NMI	F1	AUPRC	Accuracy	NMI
svGL	0.6757	0.6321	0.9359	0.3798	0.8708	0.9010	0.9725	0.6802	0.8675	0.8510	0.9739	0.6713
JEMGL_{G}	**0.8916**	**0.8846**	**0.9784**	**0.7162**	0.8756	0.8735	0.9750	0.6882	0.9164	0.9129	0.9833	0.7677
JEMGL_{LS}	**0.9103**	**0.8904**	**0.9823**	**0.7565**	0.8064	0.7569	0.9612	0.5625	0.8725	0.8410	0.9744	0.6778
$\text{mvGL}^{\text{pw}}_{\ell_1}$	0.8725	0.8391	0.9751	0.6818	**0.9105**	**0.9295**	**0.9814**	**0.7585**	**0.9427**	**0.9339**	**0.9886**	**0.8330**
$\text{mvGL}^{\text{pw}}_{\ell_2}$	0.8674	0.8359	0.9742	0.6729	0.9029	0.9228	0.9798	0.7423	0.9376	0.9290	0.9876	0.8207
mvGL_{ℓ_1}	0.8716	0.8363	0.9749	0.6804	**0.9122**	**0.9314**	**0.9818**	**0.7620**	**0.9425**	**0.9314**	**0.9886**	**0.8327**
mvGL_{ℓ_2}	0.8660	0.8321	0.9740	0.6706	0.9015	0.9229	0.9795	0.7395	0.9350	0.9245	0.9871	0.8143
svGL	0.8745	0.8093	0.9772	0.7185	0.9233	**0.9582**	0.9838	0.7897	**0.9717**	0.9609	**0.9944**	**0.9062**
JEMGL_{G}	**0.9552**	**0.9671**	**0.9909**	**0.8591**	**0.9344**	0.9329	**0.9869**	**0.8080**	0.9650	**0.9706**	0.9930	0.8856
JEMGL_{LS}	**0.9658**	**0.9605**	**0.9931**	**0.8878**	0.8725	0.8687	0.9744	0.6786	0.9376	0.9370	0.9875	0.8155
$\text{mvGL}^{\text{pw}}_{\ell_1}$	0.9290	0.9046	0.9862	0.8032	0.9269	0.9510	0.9847	0.7962	**0.9733**	0.9687	**0.9946**	**0.9126**
$\text{mvGL}^{\text{pw}}_{\ell_2}$	0.8658	0.9163	0.9713	0.6709	0.7794	**0.9521**	0.9442	0.5658	0.8542	**0.9785**	0.9656	0.6805
mvGL_{ℓ_1}	0.9268	0.8956	0.9859	0.8022	**0.9278**	0.9451	**0.9851**	**0.7964**	0.9689	0.9560	0.9939	0.9027
mvGL_{ℓ_2}	0.9235	0.9134	0.9849	0.7859	0.9102	0.9504	0.9807	0.7664	0.9622	0.9556	0.9924	0.8882

Fig. 5. Comparison of the methods using different performance measures. Top table shows view performance. Bottom table shows consensus performance. The top-2 highest scores are shown in boldface.

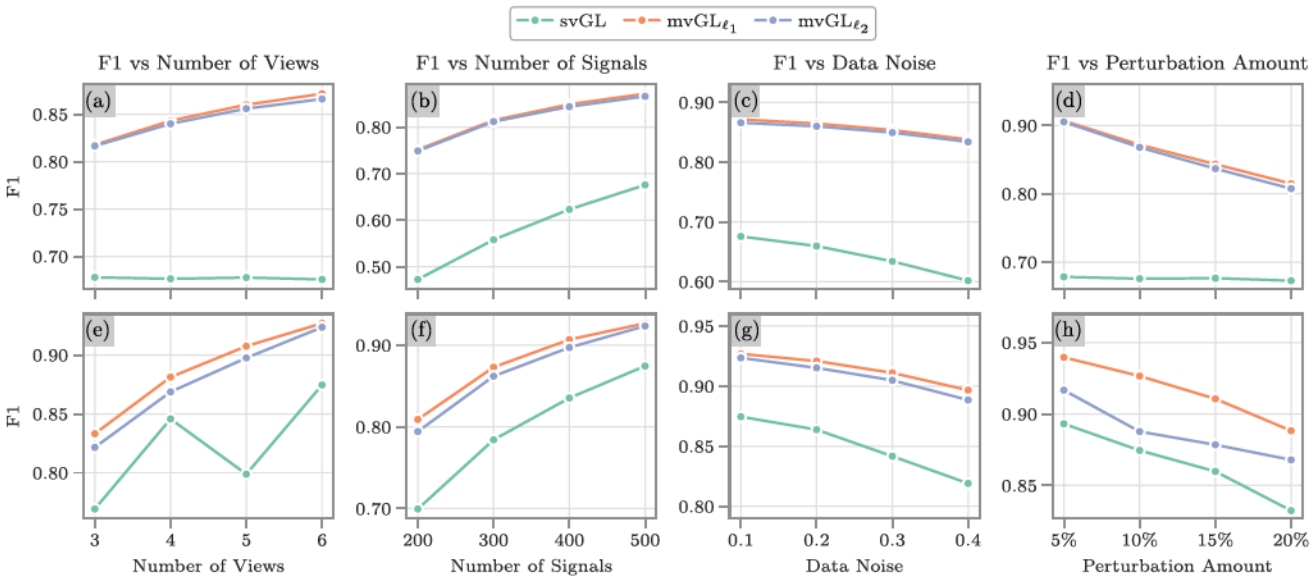

Fig. 6. Effect of data parameters on the performance of the proposed methods. (a)–(d) show the performance when learning view graphs. (e)–(h) show the consensus graph learning performance. Performance of svGL is shown as baseline.

Fig. 6 shows the experiment results for mvGL_{ℓ_1} and mvGL_{ℓ_2}. svGL performance is also reported as baseline. Comparison of JEMGL and mvGL^{pw} to the proposed method is the same as the previous experiments. Thus, they are not included in this discussion. We start with highlighting what is common to all subfigures. First, both models have higher performance than svGL in terms learning view and consensus graphs, indicating that the proposed optimization problem in (3) can effectively share information across views irrespective of which $c(\cdot)$ is used. In most cases, mvGL_{ℓ_1} is better than mvGL_{ℓ_2} in identifying consensus graphs. Although not very apparent in the figures, mvGL_{ℓ_1} performs slightly better than mvGL_{ℓ_2} in learning view graphs as well. The reason for this observation is that the perturbation model used to generate simulated data is more inline with mvGL_{ℓ_1} assumptions. Namely, each view is *independently* generated from the consensus graph by a small amount of edge removal and addition. Thus, if an edge differs across views, only a few number of views are different than the consensus graph. This relation between views and consensus graph is more inline with mvGL_{ℓ_1} assumptions as discussed in Section IV-D.

Next, we discuss how data parameters affect the proposed methods. Fig. 6(a) and (e) show the performance when N is increased from 3 to 6 ($p = 500$, $\eta = 0.1$, $r = 10$) for views and consensus graph, respectively. Performance of mvGL_{ℓ_1} and

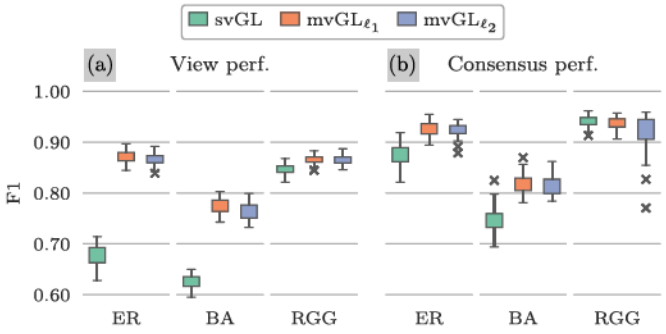

Fig. 7. Performance of the proposed methods across different graph models for learning views (a) and learning consensus graph (b). Performance of svGL is shown as baseline.

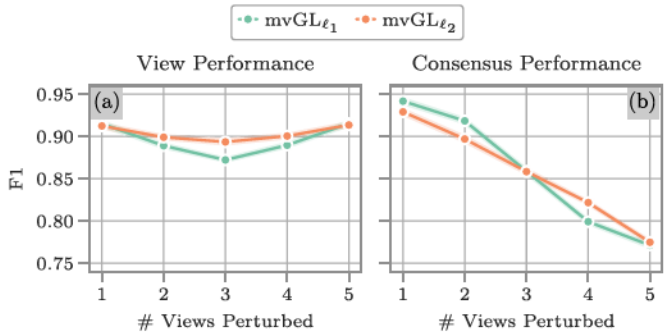

Fig. 8. Comparison of mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ on an the perturbation model q_i described in Section V-A5. (a) and (b) show the performance of learning views and consensus, respectively.

mvGL$_{\ell_2}$ in learning the views and the consensus improves with more views as expected, since the proposed methods have more data to learn from. The view performance of svGL stays the same as it does not share information across views. Since the consensus graph of svGL is the median of views, Fig. 6(e) shows an uptrend with more views. However, we also observe that svGL's consensus performance drops with odd number of views, which is the effect of the median operator and no information sharing across views. Comparing this to mvGL$_{\ell_1}$, which in theory also learns the consensus as the median of views, we observe that it does not suffer from such instabilities. Thus, while one can learn a consensus graph from svGL, learning the consensus through a joint learning scheme improves the results. Fig. 6(b) and (e) show the performance of methods with respect to p ($N = 6$, $\eta = 0.1$, $r = 10$). As expected, the performance of all methods improves with increasing number of signals. Fig. 6(c) and (f) show the effect of noise on methods ($p = 500$, $N = 6$, $r = 10$). All methods suffer from increasing data noise. However, the performances of mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ drop less than svGL, since they share information across views which makes them more robust to noise. Fig. 6(d) and (h) show the effect of perturbation amount r on learning the views and consensus graph, respectively. As we increase the perturbation, views become less similar to the consensus graph and to each other. Therefore, we observe a drop in performance of mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ as expected. For svGL, view performance stays the same since it learns each view independently. As views become less similar to the consensus, svGL's consensus performance also drops similar to the proposed methods.

Finally, we inspect the graph learning performance across three graph models: Erdős–Rényi, Barabási–Albert and random geometric graph (RGG). In ER graphs, node pairs are independently connected with probability 0.1. BA model is a growth model where a graph is generated by iteratively growing an initial star graph with 6 nodes. At each iteration, a new node is added to the graph with 5 edges that are preferentially attached to existing nodes with the highest degree. For RGG, we used the setup from [15], where 100 2D points are randomly drawn from $[0, 1]^2$ and they are connected to each other with weight $\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/\sigma^2)$ where $\mathbf{x}_i$ is the coordinates of ith point and $\sigma = 0.25$. Weights smaller than 0.6 are set to 0, while the remaining ones are set to 1. Fig. 7 shows the F1 scores of mvGL$_{\ell_1}$, mvGL$_{\ell_2}$ and svGL. Proposed methods consistently outperform svGL across all three graph models when learning view graphs. For the consensus graph, svGL has similar performance to mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ when RGG is used and it underperforms otherwise. Consistent with the previous work [15], performance of methods are slightly better for RGG than ER and the lowest F1 scores are obtained for BA.

5) Comparison of mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$: In the previous experiments, mvGL$_{\ell_1}$ is observed to perform better than mvGL$_{\ell_2}$. In this section, we compare the two methods to investigate cases where one is more suitable than the other. Comparison is performed on a dataset where G_0 is generated using ER model with $n = 100$ nodes and edge probability 0.1. We generated $N = 6$ views from G_0 as follows. First, each view graph is initialized as $G^i = G_0$. Next, 100 edges of G_0 are selected randomly and each selected edge is removed from randomly chosen k view graphs. Similarly, 100 unconnected node pairs are selected randomly from G_0 and each pair is connected by an edge in randomly selected k view graphs. This model is also edge removal and addition model, i.e., $\mathbf{W}^i = \mathbf{W} - \mathbf{E}_1^i \odot \mathbf{W} + \mathbf{E}_2^i \odot (\mathbf{1} - \mathbf{W})$, used in previous experiments. However, in this case $\mathbf{E}_1^i$ and $\mathbf{E}_2^i$ are dependent across views resulting in the same edges being perturbed across views for large values of k. The remaining parameters of the simulation are $\eta = 10$, $h(\mathbf{L}) = \mathbf{L}^\dagger$ and $p = 500$. Fig. 8 shows the performance of mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$. For $k < 3$, both methods have similar view performance and mvGL$_{\ell_1}$ performs better than mvGL$_{\ell_2}$ in learning the consensus graph. The reason for this is that when k is small, deviations of views from the consensus graph is independent. Thus, this setup is more inline with the assumption of mvGL$_{\ell_1}$. For $k > 3$, mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ still have similar view performance. However, this time mvGL$_{\ell_2}$ seems to have better consensus performance. When k is large, more views deviate from the consensus graph; thus we have a dataset closer to the assumption of mvGL$_{\ell_2}$. View performance of both methods drops for $k = 3$, since views deviate the most from each other at this value of k. We also observe that mvGL$_{\ell_2}$ has higher performance than mvGL$_{\ell_1}$ for $k = 3$. The reason for this could be that mvGL$_{\ell_2}$ is less restrictive since it allows more views to deviate from consensus graph at the same time as discussed in Section IV-D. Finally, increasing k drops the performance of both methods in

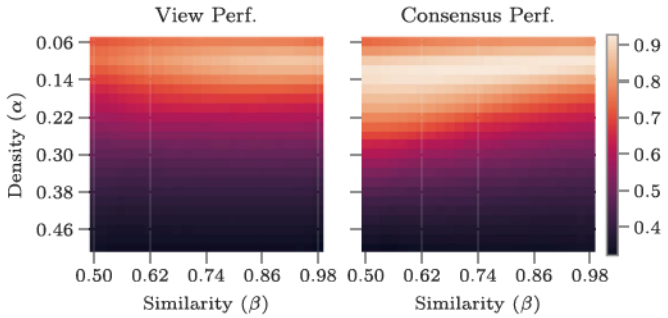

Fig. 9. Sensitivity of mvGL$_{\ell_1}$'s performance to its hyperparameters. Performance is measured with F1 score.

learning the consensus graph, since the views deviate more from the consensus graph for larger values of k.

6) Hyperparameter Sensitivity: Finally, we discuss sensitivity of the proposed method to its hyperparameters. We consider a dataset where G_0 is generated as an ER graph with edge probability 0.1. From G_0, $N = 6$ views are independently generated as in Section V-A1. The remaining parameters are $n = 100$, $p = 500$, $\eta = 10$ and $h(\mathbf{L}) = \mathbf{L}^\top$. Fig. 9 shows the sensitivity of mvGL$_{\ell_1}$ to its hyperparameters.[4] The x-axis is the similarity between the views (corresponding to β parameter) and the y-axis is the density of the learned graphs (corresponding to α parameter). Fig. 9 shows that the density of learned graphs is important for both learning the views and the consensus. In particular, the performance is maximized when the density of learned graphs is close to that of ground truth, which is 0.1 for this dataset. Similarity of views has less influence on the performance than density. For learning the views, there is a large region of β values (those that yield a similarity larger than 0.8) where mvGL$_{\ell_1}$ performs well. In terms of consensus performance, mvGL$_{\ell_1}$ is less sensitive to β such that a good performance is obtained even when similarity across views is as low as 0.5.

B. EEG Dataset

In this section, the proposed method is applied to EEG recordings collected from multiple subjects, where the goal is to learn the functional connectivity networks (FCNs) for each subject. Traditional neuroimaging studies collapse data from multiple subjects to make inferences about functional connectivity patterns within a group of subjects. While studying group level brain connectivity is crucial for detecting disorders, designing effective treatment and mapping brain networks, recent research in functional connectome fingerprinting points out inter-subject variability during cognitive tasks [52], [53]. In this section, we apply the proposed methods to learn both subject level individual FCNs, i.e., G^is, as well as the group level FCN, i.e., G_0. The graphs learned by the proposed framework are validated by (i) comparing them to those found by svGL and (ii) comparing their topology to existing literature.

[4]Sensitivity analysis of mvGL$_{\ell_2}$ is not reported, as it leads to the same conclusions as mvGL$_{\ell_1}$.

a) Data description: The EEG data is recorded from 20 subjects performing a cognitive control-related error processing task [54]. The data collection was conducted by following the experimental protocol approved by the Institutional Review Board (IRB) of the Michigan State University (IRB: LEGACY13-144). The EEG signals are recorded with a BioSemi ActiveTwo system using a cap with 64 Ag–AgCl electrodes placed at standard locations of the International 10–20 system. The sampling rate was 512 Hz. The collected data is processed using standard artifact rejection algorithms [55], followed by the volume conduction minimization from the Current Source Density (CSD) Toolbox [56].

During recordings, subjects perform a letter version of the speeded reaction Flanker task, where a string of five letters, either congruent (e.g., SSSSS) or incongruent (e.g., SSTSS), is shown at each trial. Subjects use a standard mouse to react to the center letter and the goal is to capture the Error-Related Negativity (ERN) after an error response. Each trial begins with a flanking stimulus (e.g., SS SS) of 35 ms followed by the target stimuli (e.g., SSSSS/SSTSS) displayed for about 100 ms. There is a 1200 to 1700 ms break between trials. Each subject performs 480 trials, where number of trials with error response ranges from 34 to 139 across subjects. In this paper, data from error trials are employed for graph learning and only 34 error trials corresponding to each subject are taken into account to ensure that the subjects have the same amount of data.

b) Preprocessing: Before performing graph learning, we preprocess the EEG data as follows. Let $\overline{\mathbf{X}}^{sk} \in \mathbb{R}^{64\times 512}$ be the EEG recordings of the sth subject's kth trial, where 64 is the number of electrodes and 512 is the number of time points. Thus, each row of $\overline{\mathbf{X}}^{sk}$ corresponds to EEG signals from an electrode. Based on our previous work [57], which indicates increased error related activity for the θ frequency band (4–7 Hz) and time window (approximately 0–100 ms), we first bandpass filter each row of $\overline{\mathbf{X}}^{sk}$ within the θ band and then consider samples from 0–100 ms. We perform graph learning using the data matrix, $\mathbf{X}^{sk} \in \mathbb{R}^{64\times 50}$, obtained after filtering and windowing operations.

c) "Ground-truth"-based comparison: Since it is not possible to know the true FCNs for EEG data, we consider a data-driven way of generating "ground-truth" graphs similar to [25]. In particular, we generate two different types of "ground-truth" graphs using svGL as follows.

In the first case, we learn a "ground-truth" graph structure for each subject separately using svGL. For each subject s, a data matrix $\mathbf{X}^s \in \mathbb{R}^{64\times m}$ is constructed from all 34 $\mathbf{X}^{sk}$ matrices, where $m = 34 \times 50$ and 34 is the number of trials. A "ground-truth" graph G^s with edge density of 0.15 is then learned from $\mathbf{X}^s$ using svGL. Next, for each subject, we construct a partial data matrix $\widehat{\mathbf{X}}^s \in \mathbb{R}^{64\times m_\mu}$ from the first $\lfloor 34\mu \rfloor$ $\mathbf{X}^{sk}$ matrices, where $0 < \mu < 1$ and $m_\mu = \lfloor 34\mu \rfloor \times 50$. Graphs are then learned using mvGL$_{\ell_1}$ and svGL from $\widehat{\mathbf{X}}^s$ following the same setup used for simulated data and then compared to "ground-truth" G^s's using F1 score.

The accuracy of recovery results are reported in Fig. 10(a) for different values of μ. As can be seen, mvGL$_{\ell_1}$ has higher

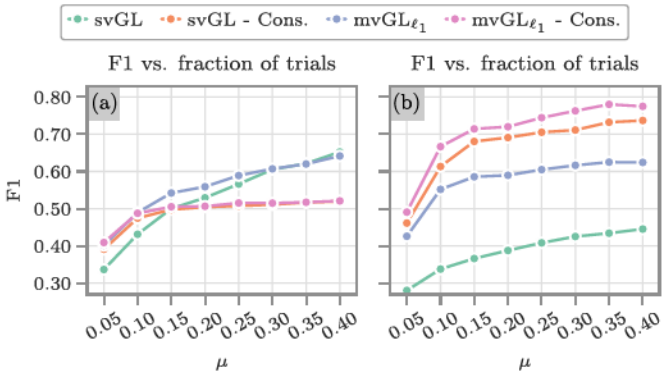

Fig. 10. Performance of different methods when "ground-truth" graphs are constructed with svGL using all EEG trials. (a) A "ground-truth" graph is constructed for each subject using all available trials of the subject. (b) A single "ground-truth" graph is constructed across all trials and subjects.

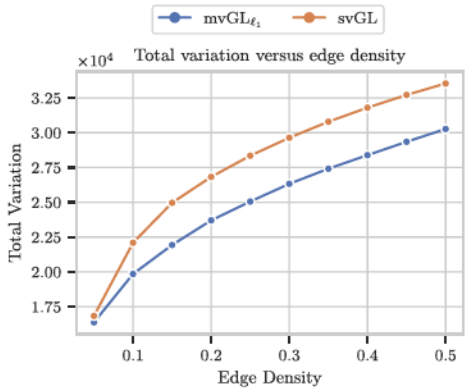

Fig. 11. Total variation of test data on subject graphs learned.

performance than svGL for values of μ up to 0.3. When the number of graph signals to learn from is low, mvGL$_{\ell_1}$ performs better, indicating that the proposed method can share information across different graphs improving the accuracy. For $\mu > 0.3$, the performance of mvGL$_{\ell_1}$ and svGL become similar to each other. The figure also reports the performance of the consensus graph of mvGL$_{\ell_1}$ and svGL, where for the latter the median of the views is used to obtain the consensus graph similar to the simulations. Both consensus graphs are compared to each subject's "ground-truth" G^s and average F1 score across subjects is reported. Since the consensus graphs cannot account for individual variability, their performance is lower than that for the view graphs learned by mvGL$_{\ell_1}$ and svGL.

In the second case, a single "ground-truth" graph across all subjects is found using svGL. In particular, we construct a data matrix $\mathbf{X} \in \mathbb{R}^{64 \times m}$ by concatenating 34 trial data matrices $\mathbf{X}^{sk}$ from all subjects, where $m = 34 \times 50 \times 20$ and 20 is the number of subjects. svGL is then applied to $\mathbf{X}$ to learn a graph G_0 with edge density of 0.15. G_0 is considered to be the ground truth graph for all subjects. mvGL$_{\ell_1}$ and svGL are then applied to previously constructed partial data matrices, i.e., $\widehat{\mathbf{X}}^s$'s, to learn the subject graphs. F1 scores are reported in Fig. 10(b), where consensus graphs learned by both methods have higher F1 score than learned view graphs. This result is expected, since the "ground-truth" G_0 mostly includes shared edges across subjects as it is learned using signals from all subjects. It is also observed that the consensus graph learned by mvGL$_{\ell_1}$ has the highest F1 value, indicating that shared structure is better revealed by the proposed method than the consensus of svGL. Finally, subject graphs learned by mvGL$_{\ell_1}$ is better than those learned by svGL for all μ values. These results indicate that the proposed method i) provides better performance when the number of data samples is low by sharing information across views, ii) efficiently captures the shared structure across views, leading to better performance.

d) Smoothness-based comparison: Both mvGL and svGL rely on the assumption that the data is smooth with respect to the unknown graph structure. Therefore, we next compare their performances based on the smoothness of the learned graphs. To this end, we perform a five-fold cross validation by splitting 34 trials of each subject into five splits. One split is used as training set to learn the subject graphs, while the remaining ones are used as the test set, whose total variation (see (1)) with respect to the learned subject graphs is calculated. svGL and mvGL$_{\ell_1}$ are used to learn the subject graphs with varying edge densities and β parameter of mvGL$_{\ell_1}$ is set to a value where average pairwise correlation between subject graphs is around 0.75. Average total variation of the test dataset is reported in Fig. 11. It can be seen that subject graphs found by mvGL$_{\ell_1}$ have lower total variation, indicating that mvGL$_{\ell_1}$ manages to learn graphs that fit the smoothness assumption better.

e) Community detection: An important goal of FCN analysis is to identify the functionally related subnetworks. Community detection can be used to partition the FCNs to achieve this goal [58]. To further validate the subject graphs found by mvGL, we detect the community structure of the learned graphs and compare them to well-known networks in literature. mvGL$_{\ell_1}$ is employed to learn subject graphs $\{G^s\}_{s=1}^{20}$ and a consensus graph G^c using data from all available trials. We set α and β such that the edge density and average pairwise correlation are 0.15 and 0.75, respectively. svGL is also used to learn subject graphs, which is represented by $\{G^s_{sv}\}_{s=1}^{20}$. α parameter for svGL is set similarly to have 0.15 edge density.

One common approach for analyzing the community structure of FCNs across multiple subjects is to find the group community structure, that represents the shared partitioning across a group of subjects [59]. Existing works usually perform this task by first finding each subject's community structure and then employing consensus clustering [60]. Applying community detection to the consensus graph learned by the proposed method can eliminate this two-step process. To this end, we found community structures of graphs learned by mvGL$_{\ell_1}$ and svGL by maximizing modularity [61] using Leiden algorithm [62]. Let $\{\mathcal{P}^s\}_{s=1}^{20}$ be the subjects' community structure detected from $\{G^s\}_{s=1}^{20}$. Let $\mathcal{P}^c$ be the community structure of G^c. Similarly, let $\{\mathcal{P}^s_{sv}\}_{s=1}^{20}$ be the subjects' community structure found from graphs learned by svGL, i.e., $\{G^s_{sv}\}_{s=1}^{20}$. Finally, let $\mathcal{P}^c_{sv}$ be the group community structure of $\{\mathcal{P}^s_{sv}\}_{s=1}^{20}$ found by consensus clustering [60]. We calculate how consistent consensus community structures, $\mathcal{P}^c$ and $\mathcal{P}^c_{sv}$, are with the subjects' community structures, $\{\mathcal{P}^s\}_{s=1}^{20}$ and $\{\mathcal{P}^s_{sv}\}_{s=1}^{20}$, using normalized mutual

TABLE I
CONSISTENCY OF GROUP COMMUNITY STRUCTURES WITH SUBJECTS' COMMUNITY STRUCTURES AS MEASURED BY NMI

	$\{\mathcal{P}^s\}_{s=1}^{20}$	$\{\mathcal{P}_{sv}^s\}_{s=1}^{20}$
$\mathcal{P}^c$	0.619	0.487
$\mathcal{P}_{sv}^c$	0.418	0.427

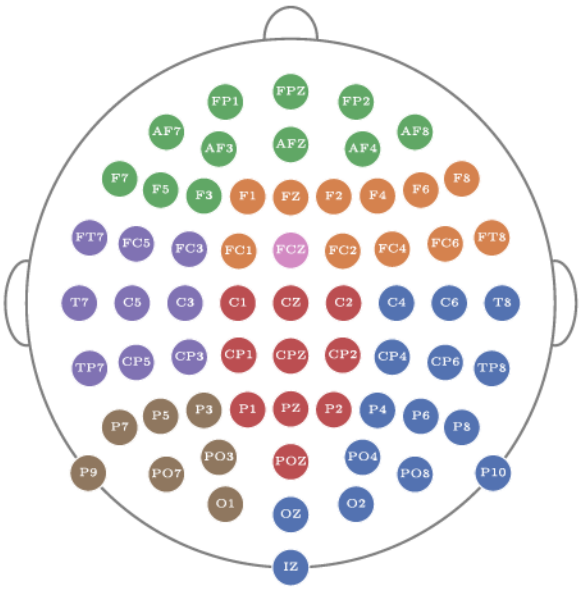

Fig. 12. Community structure of the G^c learned by mvGL$_{\ell_1}$. Each node corresponds to an electrode and the coloring indicates their community memberships. The name of each electrode is coded based on the 10-20 system: Fp for pre-frontal, F for frontal, T for temporal, P for parietal, O for occipital, and C for central. Even numbers indicate right hemisphere locations, and odd numbers indicate left hemisphere locations.

information (NMI) [63]. The results are reported in Table I and it is seen that $\mathcal{P}^c$ is more consistent with both $\{\mathcal{P}^s\}_{s=1}^{20}$ and $\{\mathcal{P}_{sv}^s\}_{s=1}^{20}$. Thus, mvGL learns a consensus graph whose community structure is more consistent with the individual subjects' community structures.

We also study the community structure of G^c, shown in Fig. 12, by comparing it to previous literature. The result indicates that there's a consensus community centered around frontal-central regions consistent with prior work indicating the increased activation of medial prefrontal cortex (mPFC) during cognitive control [64]. In prior work, these regions have also been shown to have increased clustering and decreased path length indicating a stronger community structure [65]. These findings from EEG studies are also supported by spatial maps of the fMRI-derived independent components suggesting distinct contributions from a distributed fronto-striatal system to the processing of conflicts, response-inhibition and error-monitoring, respectively [66]. In addition, there are communities centered around the left and right lateral prefrontal cortices and the visual and motor regions consistent with the task.

VI. CONCLUSION

This paper introduced multiview graph learning based on the smoothness assumption for applications where multiple views of the same phenomenon are observed. The proposed framework learns both the individual view graphs as well as a consensus graph that captures the shared structured across views. The similarity across the different views is ensured through a consensus term between the individual view graphs and the consensus graph. The resulting optimization problem is formulated in a general way such that different functions could be employed for the consensus and regularization terms. The results illustrate the advantage of multiview graph learning over single graph learning when there's shared information across the views and when the data is noisy.

Future work will explore the use of different functions for consensus and regularization such as information-theoretic functionals or generalized norms. While mvGL$_{\ell_1}$ and mvGL$_{\ell_2}$ focus on edge-based similarity across the views, they can also be extended to learn multiple graphs using the commonality of node-based structures, e.g., hub nodes [31]. The proposed approach's time complexity is quadratic in the number of nodes, which is not applicable to large-scale graphs. Future work will also consider scalability.

APPENDIX A
VECTORIZATION OF (3)

In order to show how to vectorize the optimization problem in (3), consider each term separately:

- For smoothness terms, let $\mathbf{K}^i = \mathbf{X}^i\mathbf{X}^{i\top}$, then we obtain $\text{tr}(\mathbf{X}^{i\top}\mathbf{L}^i\mathbf{X}^i) = \text{tr}(\mathbf{X}^i\mathbf{X}^{i\top}\mathbf{L}^i) = \text{tr}(\mathbf{K}^i\mathbf{L}^i)$, where the last term can be written as:

$$\begin{aligned}\text{tr}(\mathbf{K}^i\mathbf{L}^i) &= \sum_{a=1}^{n}\sum_{b=1}^{n} K_{ab}^i L_{ab}^i \\ &= 2\sum_{a=1}^{n}\sum_{b=a+1}^{n} K_{ab}^i L_{ab}^i + \sum_{a=1}^{n} K_{aa}^i L_{aa}^i \\ &= 2\mathbf{k}^{i\top}\boldsymbol{\ell}^i - \mathbf{d}^{i\top}\mathbf{S}\boldsymbol{\ell}^i = (2\mathbf{k}^i - \mathbf{S}^\top\mathbf{d}^i)^\top\boldsymbol{\ell}^i,\end{aligned}$$

  where in the second line we used the fact that both $\mathbf{K}^i$ and $\mathbf{L}^i$ are symmetric matrices and we used $\text{diag}(\mathbf{L}^i) = -\mathbf{S}\boldsymbol{\ell}^i$ in the third line.
- The Frobenius norm terms can be vectorized similarly where we use the symmetric structure of $\mathbf{L}^i$ and $\text{diag}(\mathbf{L}^i) = -\mathbf{S}\boldsymbol{\ell}^i$:

$$\begin{aligned}\left\|\mathbf{L}^i\right\|_F^2 &= \sum_{a=1}^{n}\sum_{b=1}^{n} L_{ab}^i L_{ab}^i \\ &= 2\sum_{a=1}^{n}\sum_{b=a+1}^{n} L_{ab}^i L_{ab}^i + \sum_{a=1}^{n} L_{aa}^i L_{aa}^i \\ &= 2\boldsymbol{\ell}^{i\top}\boldsymbol{\ell}^i + \boldsymbol{\ell}^{i\top}\mathbf{S}^\top\mathbf{S}\boldsymbol{\ell}^i = \boldsymbol{\ell}^{i\top}(\mathbf{S}^\top\mathbf{S} + 2\mathbf{I})\boldsymbol{\ell}^i.\end{aligned}$$

- Since all of the information for $\mathbf{L}^i$ and $\mathbf{L}$ is in the upper triangular parts of the matrices, $c(\cdot)$ and $r(\cdot)$ can simply be converted to $c_v(\cdot)$ and $r_v(\cdot)$, which return the same values given the upper triangular parts.
- For the constraint $\mathbf{L}^i \in \mathbb{L}$ (and $\mathbf{L} \in \mathbb{L}$), we use the definition of $\mathbb{L}$ which includes $L_{ab}^i = L_{ba}^i \leq 0$ and $\mathbf{L}\mathbf{1} = 0$.

The former implies $\text{upper}(\mathbf{L}^i) = \boldsymbol{\ell}^i \leq 0$. $\mathbf{L}\mathbf{1} = 0$ is due to $\text{diag}(\mathbf{L}^i) = -\mathbf{S}\boldsymbol{\ell}^i$ and it can be ignored as we are only learning $\boldsymbol{\ell}^i$.

- Finally, $\text{tr}(\mathbf{L}^i) = 2n$ constrains the sum of node degrees to be $2n$ in the learned $\mathbf{L}^i$. Since the sum of node degrees can also be calculated by $-2\mathbf{1}^\top\boldsymbol{\ell}^i$, we have the constraint $\mathbf{1}^\top\boldsymbol{\ell}^i = -n$ in (4).

Appendix B
Solution for ADMM Steps

In this section, the optimization of two ADMM steps ((7) and (8)) are given. As mentioned in the main text, (7) can be solved separately for $\boldsymbol{\ell}^i$'s, $\mathbf{v}^i$'s and $\boldsymbol{\ell}$:

- Equation (7) can be optimized for each $\boldsymbol{\ell}^i$, separately. For this, rewrite the problem by ignoring all terms that do not depend on $\boldsymbol{\ell}^i$:

$$\begin{aligned}\hat{\boldsymbol{\ell}}^i &= \underset{\boldsymbol{\ell}^i}{\text{argmin}}\, f(\boldsymbol{\ell}^i)+\imath_1(\boldsymbol{\ell}^i)+\hat{\mathbf{y}}^{i\top}(\hat{\mathbf{z}}^i-\boldsymbol{\ell}^i)+\frac{\rho}{2}\left\|\hat{\mathbf{z}}^i-\boldsymbol{\ell}^i\right\|_2^2 \\ &= \underset{\boldsymbol{\ell}^i}{\text{argmin}}\, (2\mathbf{k}^i-\mathbf{S}^\top\mathbf{d}^i)^\top\boldsymbol{\ell}^i+\alpha\boldsymbol{\ell}^{i\top}(\mathbf{S}^\top\mathbf{S}+2\mathbf{I})\boldsymbol{\ell}^i \\ &\qquad +\hat{\mathbf{y}}^{i\top}(\hat{\mathbf{z}}^i-\boldsymbol{\ell}^i)+\frac{\rho}{2}\left\|\hat{\mathbf{z}}^i-\boldsymbol{\ell}^i\right\|_2^2 \\ &\quad \text{s.t.}\quad \mathbf{1}^\top\boldsymbol{\ell}^i=-n, \end{aligned} \tag{16}$$

where we substitute $f(\boldsymbol{\ell}^i)$ and convert the indicator function to a constraint in the second equality. (16) is a quadratic problem with an equality constraint. Using its KKT conditions, its minimizer can be found as:

$$\hat{\boldsymbol{\ell}}^i=\Pi_1[(2\alpha\mathbf{S}^\top\mathbf{S}+(4\alpha+\rho)\mathbf{I})^{-1}(\mathbf{S}^\top\mathbf{d}^i-2\mathbf{k}^i+\hat{\mathbf{y}}^i+\rho\hat{\mathbf{z}}^i)]$$

where Π_1 is the projection operator onto the hyperplane $\{\boldsymbol{\ell}\in\mathbb{R}^m|\mathbf{1}^\top\boldsymbol{\ell}=-n\}$.

- To solve (7) with respect to $\{\mathbf{v}^i\}_{i=1}^N$, first define $\mathbf{V}\in\mathbb{R}^{m\times N}$ where columns of $\mathbf{V}$ are $\mathbf{v}^i$'s. Then, write (7) in terms of $\mathbf{V}$ while ignoring all terms that do not depend on $\mathbf{V}$:

$$\begin{aligned}\hat{\mathbf{V}} &= \underset{\mathbf{V}}{\text{argmin}}\ \beta c_v(\mathbf{V})+\frac{\rho}{2}\sum_{i=1}^N\left\|\mathbf{V}_{\cdot i}-\hat{\mathbf{z}}^i+\hat{\mathbf{z}}+\frac{1}{\rho}\hat{\mathbf{w}}\right\|_2^2 \\ &= \underset{\mathbf{V}}{\text{argmin}}\ \beta c_v(\mathbf{V})+\frac{\rho}{2}\left\|\mathbf{V}-\mathbf{A}\right\|_F^2, \end{aligned} \tag{17}$$

where in the first step, we insert the scaled form of the augmented Lagrangian into (7). In the second line, the summation term is written in a matrix form where $\mathbf{A}\in\mathbb{R}^{m\times N}$ with $\mathbf{A}_{\cdot i}=\hat{\mathbf{z}}^i-\hat{\mathbf{z}}-1/\rho\hat{\mathbf{w}}^i$. The optimization problem in (17) is the proximal operator of $c_v(\cdot)$. For the proposed models in Section IV-D, the proximal operator of $c_v(\cdot)$ has closed form solutions [45], [67].

- Finally, in order to solve (7) with respect to $\boldsymbol{\ell}$; we rewrite it while ignoring the terms that do not depend on $\boldsymbol{\ell}$:

$$\hat{\boldsymbol{\ell}}=\underset{\boldsymbol{\ell}}{\text{argmin}}\,\gamma r_v(\boldsymbol{\ell})+\frac{\rho}{2}\left\|\hat{\mathbf{z}}-\boldsymbol{\ell}+\frac{1}{\rho}\hat{\mathbf{y}}\right\|_2^2, \tag{18}$$

where we again use the scaled form of the augmented Lagrangian. (18) is the proximal operator of $r_v(\cdot)$. For mvGL_{ℓ_2}, $r_v(\cdot)$ is ℓ_1-norm, whose proximal operator is the soft-thresholding operator [67].

The second step of ADMM given in (8) cannot be separated across its variables. However, we solve it with BCD where it is optimized with respect to $\mathbf{z}^i$'s and $\mathbf{z}$ alternatingly with the following subproblems:

$$\begin{aligned}\underset{\mathbf{z}^i}{\text{minimize}}\ &\imath_2(\mathbf{z}^i)+\frac{\rho}{2}\left\|\mathbf{z}^i-\hat{\boldsymbol{\ell}}^i+\frac{1}{\rho}\hat{\mathbf{y}}^i\right\|_2^2 \\ &+\frac{\rho}{2}\left\|\hat{\mathbf{v}}^i-\mathbf{z}^i+\mathbf{z}+\frac{1}{\rho}\hat{\mathbf{w}}^i\right\|_2^2,\ \forall i \end{aligned} \tag{19}$$

$$\begin{aligned}\underset{\mathbf{z}}{\text{minimize}}\ &\imath_2(\mathbf{z})+\frac{\rho}{2}\sum_{i=1}^N\left\{\left\|\hat{\mathbf{v}}^i-\mathbf{z}^i+\mathbf{z}+\frac{1}{\rho}\hat{\mathbf{w}}^i\right\|_2^2\right\} \\ &+\frac{\rho}{2}\left\|\mathbf{z}-\hat{\boldsymbol{\ell}}+\frac{1}{\rho}\hat{\mathbf{y}}\right\|_2^2, \end{aligned} \tag{20}$$

which are derived from (8) by inserting the scaled version of augmented Lagrangian and ignoring the constant terms. At the kth iteration of BCD, $\mathbf{z}$ in (19) is set to the value obtained by solving (20) at the $(k-1)$th iteration. Similarly, $\mathbf{z}^i$'s in (19) are set to the values obtained by solving (19) at the kth iteration. Both (19) and (20) are proximal operators of $\imath_2(\cdot)$, which is equal to the projection onto $\mathbb{R}^m_{\leq 0}$.

References

[1] M. Newman, *Networks*. London, U.K.: Oxford Univ. Press, 2018.
[2] O. Sporns, *Discovering the Human Connectome*. Cambridge, MA, USA: MIT Press, 2012.
[3] H. Li and J. Gui, "Gradient directed regularization for sparse Gaussian concentration graphs, with applications to inference of genetic networks," *Biostatistics*, vol. 7, no. 2, pp. 302–317, 2006.
[4] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, May 2018.
[5] M. Drton and M. H. Maathuis, "Structure learning in graphical modeling," *Annu. Rev. Statist. Appl.*, vol. 4, pp. 365–393, 2017.
[6] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 44–63, May 2019.
[7] 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.
[8] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, "Neural relational inference for interacting systems," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2688–2697.
[9] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical Lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.
[10] D. Thanou, X. Dong, D. Kressner, and P. Frossard, "Learning heat diffusion graphs," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 484–499, Sep. 2017.
[11] 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.
[12] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of graph diffusion processes from observations of stationary signals," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 3, pp. 481–496, Sep. 2018.
[13] R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos, "Identifying the topology of undirected networks from diffused non-stationary graph signals," *IEEE Open J. Signal Process.*, vol. 2, pp. 171–189, 2021.

[14] 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.
[15] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Artif. Intell. Statist.*, 2016, pp. 920–929.
[16] P. Berger, G. Hannak, and G. Matz, "Efficient graph learning from noisy and incomplete data," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 105–119, 2020.
[17] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Learning discrete structures for graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1972–1982.
[18] Y. Chen, L. Wu, and M. Zaki, "Iterative deep graph learning for graph neural networks: Better and robust node embeddings," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 19314–19326.
[19] S. Zhang et al., "Inference of cell type-specific gene regulatory networks on cell lineages from single cell omic datasets," *Nature Commun.*, vol. 14, no. 1, 2023, Art. no. 3064.
[20] A. Karaaslanli, S. Saha, T. Maiti, and S. Aviyente, "Kernelized multiview signed graph learning for single-cell RNA sequencing data," *BMC Bioinf.*, vol. 24, no. 1, pp. 1–17, 2023.
[21] R. F. Betzel, M. A. Bertolero, E. M. Gordon, C. Gratton, N. U. Dosenbach, and D. S. Bassett, "The community structure of functional brain networks exhibits scale-specific patterns of inter-and intra-subject variability," *Neuroimage*, vol. 202, 2019, Art. no. 115990.
[22] M. De Domenico, "Multilayer modeling and analysis of human brain networks," *Giga Sci.*, vol. 6, no. 5, 2017, Art. no. gix004.
[23] K. Tsai, O. Koyejo, and M. Kolar, "Joint Gaussian graphical model estimation: A survey," *Wiley Interdiscipl. Rev. Comput. Statist.*, vol. 14, no. 6, 2022, Art. no. e1582.
[24] P. Danaher, P. Wang, and D. M. Witten, "The joint graphical Lasso for inverse covariance estimation across multiple classes," *J. Roy. Statist. Soc. B.*, vol. 76, no. 2, pp. 373–397, 2014.
[25] 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. 1, pp. 3302–3336, 2022.
[26] Y. Yuan, D. W. Soh, K. Guo, Z. Xiong, and T. Q. S. Quek, "Joint network topology inference via structural fusion regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 10351–10364, Oct. 2023.
[27] S. Hara and T. Washio, "Learning a common substructure of multiple graphical Gaussian models," *Neural Netw.*, vol. 38, pp. 23–38, 2013.
[28] W. Lee and Y. Liu, "Joint estimation of multiple precision matrices with common structures," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1035–1062, 2015.
[29] X. Zhang and Q. Wang, "A graph-assisted framework for multiple graph learning," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 10, pp. 162–178, 2024.
[30] J. Guo, E. Levina, G. Michailidis, and J. Zhu, "Joint estimation of multiple graphical models," *Biometrika*, vol. 98, no. 1, pp. 1–15, 2011.
[31] K. Mohan, P. London, M. Fazel, D. Witten, and S.-I. Lee, "Node-based learning of multiple Gaussian graphical models," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 445–488, 2014.
[32] J. Ma and G. Michailidis, "Joint structural estimation of multiple graphical models," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 5777–5824, 2016.
[33] F. Huang and S. Chen, "Joint learning of multiple sparse matrix Gaussian graphical models," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2606–2620, Nov. 2015.
[34] M. Navarro and S. Segarra, "Joint network topology inference via a shared graphon model," *IEEE Trans. Signal Process.*, vol. 70, pp. 5549–5563, 2022.
[35] H. P. Maretic and P. Frossard, "Graph Laplacian mixture model," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 261–270, 2020.
[36] H. Araghi, M. Sabbaqi, and M. Babaie-Zadeh, "k-Graphs: An algorithm for graph signal clustering and multiple graph learning," *IEEE Signal Process. Lett.*, vol. 26, no. 10, pp. 1486–1490, Oct. 2019.
[37] A. Karaaslanli and S. Aviyente, "Simultaneous graph signal clustering and graph learning," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 10762–10772.
[38] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard, "Learning time varying graphs," in *Proc. 2017 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2017, pp. 2826–2830.
[39] K. Yamada, Y. Tanaka, and A. Ortega, "Time-varying graph learning based on sparseness of temporal variation," in *Proc. 2019 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 5411–5415.
[40] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo, "Enabling prediction via multi-layer graph inference and sampling," in *Proc. 13th Int. Conf. Sampling Theory Appl.*, 2019, pp. 1–4.
[41] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.
[42] J. Miettinen, S. A. Vorobyov, and E. Ollila, "Modelling graph errors: Towards robust graph signal processing," 2019, *arXiv:1903.08398*.
[43] Q. Han, K. Xu, and E. Airoldi, "Consistent estimation of dynamic and multi-layer block models," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1511–1520.
[44] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. B.*, vol. 58, no. 1, pp. 267–288, 1996.
[45] F. Bach et al., "Optimization with sparsity-inducing penalties," *Found. Trends Mach. Learn.*, vol. 4, no. 1, pp. 1–106, 2012.
[46] J. Huang and T. Zhang, "The benefit of group sparsity," *Ann. Stat.*, vol. 38, no. 4, pp. 1978–2004, 2010, doi: 10.1214/09-AOS778.
[47] S. Boyd et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[48] H.-J. M. Shi, S. Tu, Y. Xu, and W. Yin, "A primer on coordinate descent algorithms," 2016, *arXiv:1610.00040*.
[49] J. Eckstein and W. Yao, "Understanding the convergence of the alternating direction method of multipliers: Theoretical and computational perspectives," *Pac. J. Optim.*, vol. 11, no. 4, pp. 619–644, 2015.
[50] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *J. Sci. Comput.*, vol. 78, pp. 29–63, 2019.
[51] J. Ying, J. V. De Miranda Cardoso, and D. Palomar, "Nonconvex sparse graph learning under Laplacian constrained graphical model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 7101–7113.
[52] E. S. Finn et al., "Functional connectome fingerprinting: Identifying individuals using patterns of brain connectivity," *Nature Neurosci.*, vol. 18, no. 11, pp. 1664–1671, 2015.
[53] J. Liu, X. Liao, M. Xia, and Y. He, "Chronnectome fingerprinting: Identifying individuals and predicting higher cognitive functions using dynamic brain connectivity patterns," *Hum. Brain Mapping*, vol. 39, no. 2, pp. 902–915, 2018.
[54] T. P. Moran, D. Taylor, and J. S. Moser, "Sex moderates the relationship between worry and performance monitoring brain activity in undergraduates," *Int. J. Psychophysiol.*, vol. 85, no. 2, pp. 188–194, 2012.
[55] A. Delorme and S. Makeig, "EEGLAB: An open source toolbox for analysis of single-trial EEG dynamics including independent component analysis," *J. Neurosci. Methods*, vol. 134, no. 1, pp. 9–21, 2004.
[56] C. E. Tenke and J. Kayser, "Generator localization by current source density (CSD): Implications of volume conduction and field closure at intracranial and scalp resolutions," *Clin. Neurophysiol.*, vol. 123, no. 12, pp. 2328–2345, 2012.
[57] S. Aviyente, E. M. Bernat, W. S. Evans, and S. R. Sponheim, "A phase synchrony measure for quantifying dynamic functional integration in the brain," *Hum. Brain Mapping*, vol. 32, no. 1, pp. 80–93, 2011.
[58] O. Sporns and R. F. Betzel, "Modular brain networks," *Annu. Rev. Psychol.*, vol. 67, pp. 613–640, 2016.
[59] R. F. Betzel and D. S. Bassett, "Multi-scale brain networks," *Neuroimage*, vol. 160, pp. 73–83, 2017.
[60] A. Lancichinetti and S. Fortunato, "Consensus clustering in complex networks," *Sci. Rep.*, vol. 2, no. 1, pp. 1–7, 2012.
[61] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, 2004, Art. no. 026113.
[62] V. A. Traag, L. Waltman, and N. J. Van Eck, "From Louvain to Leiden: Guaranteeing well-connected communities," *Sci. Rep.*, vol. 9, no. 1, 2019, Art. no. 5233.
[63] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *J. Statist. Mmechanics: Theory Exp.*, vol. 2005, no. 09, 2005, Art. no. P09008.
[64] A. Ozdemir, M. Bolanos, E. Bernat, and S. Aviyente, "Hierarchical spectral consensus clustering for group analysis of functional brain networks," *IEEE Trans. Biomed. Eng.*, vol. 62, no. 9, pp. 2158–2169, Sep. 2015.
[65] M. Bolanos, E. M. Bernat, B. He, and S. Aviyente, "A weighted small world network measure for assessing functional connectivity," *J. Neurosci. Methods*, vol. 212, no. 1, pp. 133–142, 2013.
[66] R. J. Huster et al., "Multimodal imaging of functional networks and event-related potentials in performance monitoring," *Neuroimage*, vol. 56, no. 3, pp. 1588–1597, 2011.
[67] N. Parikh et al., "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, 2014.

Abdullah Karaaslanli received the B.S. degree in electrical and electronics engineering from Bogazici University, Istanbul, Turkey in 2017, and the Ph.D. degree in electrical engineering from Michigan State University, East Lansing, MI, USA, in 2023. He is currently a Postdoctoral Researcher with the Department of Electrical and Computer Engineering, Michigan State University. His research interests include various graph related signal processing and machine learning problems, such as graph topology inference, community detection, and graph-based methods for crowdsourcing.

Selin Aviyente (Senior Member, IEEE) received the B.S. degree (with Hons.) in electrical and electronics engineering from Boğaziçi University, İstanbul, Turkey, in 1997, and the M.S. and Ph.D. degrees in electrical engineering: systems from the University of Michigan, Ann Arbor, MI, USA, in 1999 and 2002, respectively. In 2002, she joined the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI, USA, where she is currently a Professor. Her research interests include statistical signal processing, higher order data representations, and complex network analysis. She was the recipient of the 2005 Withrow Teaching Excellence Award and the 2008 National Science Foundation (NSF) CAREER Award. She is an Associate Editor for IEEE TRANSACTIONS ON INFORMATION THEORY and a Senior Area Editor *IEEE Signal Processing Magazine*. She is also on several technical committees for IEEE Signal Processing Society.