Graph Learning Across Data Silos

Xiang Zhang, *Student Member, IEEE,* Qiao Wang, *Senior Member, IEEE*

***Abstract*—We consider the problem of inferring graph topology from smooth graph signals in a novel but practical scenario where data are located in multiple clients and prohibited from leaving local clients due to factors such as privacy concerns. The main difficulty in this task is how to exploit the potentially heterogeneous data of all clients under data silos. To this end, we first propose an auto-weighted multiple graph learning model to jointly learn a personalized graph for each local client and a single consensus graph for all clients. The personalized graphs match local data distributions, thereby mitigating data heterogeneity, while the consensus graph captures the global information. Moreover, the model can automatically assign appropriate contribution weights to local graphs based on their similarity to the consensus graph. We next devise a tailored algorithm to solve the induced problem, where all raw data are processed locally without leaving clients. Theoretically, we establish a provable estimation error bound and convergence analysis for the proposed model and algorithm. Finally, extensive experiments on synthetic and real data are carried out, and the results illustrate that our approach can learn graphs effectively in the target scenario.**

***Index Terms*—Graph learning, smooth graph signals, graph signal processing, data silos, privacy-preserving**

I. Introduction

Graphs are powerful tools for flexibly describing topological relationships between data entities [1], which have been extensively applied in many celebrated models, e.g., spectral clustering [2] and graph neural networks (GNNs) [3]. Among these applications, a graph that accurately represents the information inherent in the structured data is required, which, however, is not available in many cases. An alternative approach is to learn graphs directly from raw data, termed graph learning (GL), for downstream tasks [1], [4].

In the literature, many studies learn graphs from statistical models, such as Gaussian Graphical Models (GGMs) [5]. In general, these models aim to infer precision matrices (inverse covariance matrices), which encode conditional independence between random variables. Furthermore, [6]–[8] introduce (generalized) Laplacian constraints on the learned precision matrices, which enjoy many downstream applications such as graph spectral analysis [2]. Recently, with the rise of graph signal processing (GSP) [9], various methods attempted to learn graphs from a signal processing perspective. One of the most studied GSP-based models—the smoothness-based model—postulates that the observed graph signals are smooth over the underlying graph [10], [11]. Intuitively, a smooth graph signal means that the signal values corresponding to two connected nodes of the underlying graph are similar [11]. Typically, learning graphs from smooth signals is equivalent to minimizing a constrained quadratic Laplacian form problem, where the quadratic Laplacian form is related to signal smoothness, and the constraints (regularizers) are used to assign properties to the graphs, such as sparsity [10] and node connectivity [11]. Many signals appear to be smooth over their underlying graphs, e.g., meteorology data [10] and medical data [12], implying numerous applications of smoothness-based GL.

Here, we consider learning graphs in a previously unexplored scenario, where data are stored across multiple clients (e.g., companies and organizations) and are not allowed to leave their clients due to factors such as privacy concerns. This scenario is known as data-silos [13], and a typical example is medical data. Suppose some hospitals collect brain fMRI data from autistic and non-autistic individuals separately to learn the impact of autism on connectivity networks of brain functional regions [12]. However, the data are prohibited from leaving the hospital where they are stored as people are reluctant to disclose their private data. A naive way is for each client to infer graph topology independently, which avoids any data leakage. Although feasible, this approach ignores potential topological relationships between local graphs, resulting in suboptimal results. Thus, it is more reasonable to learn graphs using data information from all clients. The task is not trivial and faces two main challenges. The first challenge is how to leverage siloed data from all clients collaboratively. Furthermore, graph signals across different silos are inherently non-IID due to factors such as different data collection protocols and heterogeneous underlying graphs [14]. We still use the brain fMRI data as an example. The brain functional connectivity networks of autistic and non-autistic individuals are different due to the impact of autism. It is unreasonable to utilize heterogeneous data to learn a single global graph like traditional paradigms [10], [11]. Thus, the second challenge is handling heterogeneous data.

Regarding the first challenge, federated learning (FL) [13], [15], [16] is an emerging tool for learning models based on datasets distributed across multiple clients under privacy constraints. The primary feature of FL is that clients transmit model updates instead of raw data to a central server to collaboratively learn a global model [15]. Privacy can be preserved in this schema to some extent since all data are processed locally [16]. However, traditional FL algorithms, e.g., FedAvg [17], learn a single global model from all data, which may suffer from performance degradation when data across different clients are heterogeneous. A widely used approach to handle data heterogeneity—the second challenge—is personalized FL (PFL) [18]. The philosophy behind PFL is that we learn for each client a personalized model that matches its data distribution to mitigate the impact of heterogeneity. The techniques for adapting global models for individual

The authors are with the School of Information Science and Engineering, Southeast University, Nanjing 210096, China (e-mail: xiangzhang369@seu.edu.cn, qiaowang@seu.edu.cn).

clients include transfer learning [19], multi-task learning [20], and meta-learning [21].

Borrowing the idea from PFL, we learn a personalized graph for each local client to handle data heterogeneity. However, it poses two additional challenges due to the characteristics of GL tasks. (i) Our goal is to learn all local graphs jointly by exploiting their latent relationships so that each local graph benefits from "borrowing" information from other datasets. Thus, it is crucial to describe the relationships between local graphs, which is the focus of multiple graph learning (MGL). A common approach is to design regularization penalties, e.g., fused Lasso penalty [22], group Lasso penalty [23], Gram matrix-based penalty [24], and those describing temporally topological relationships [25]–[27]. These regularizers characterize topological relationships among multiple graphs, but few consider capturing common structures from all local graphs. In practice, many graphs share common structures. For example, connections between normally functioning brain regions in autistic and non-autistic individuals should remain the same. The common structures reflect the global information across all local datasets, which may be useful for many downstream tasks. (ii) It is infeasible to apply existing PFL algorithms, such as [20], [28]–[31], to our task since their personalization methods do not fit the GL problem. Besides, the problem of concern can be categorized as cross-silo FL, where clients are a few companies or organizations rather than massive devices in cross-device FL [15], [17]. Thus, a tailored algorithm is required to learn graphs in the target scenario.

To address these issues, we propose a framework to learn graphs from smooth but heterogeneous data under data silos. Our contributions can be summarized as follows:

- We propose an auto-weighted MGL model in which local personalized graphs and a consensus graph are jointly learned. The consensus graph can capture common structures representing global information across all datasets, while the local graphs preserve the heterogeneity of local datasets. Furthermore, our model can automatically assign contribution weights to local graphs based on their similarity to the consensus graph. Theoretically, we provide the estimation error bound of the proposed method to reveal some key factors affecting graph estimation performance.
- We develop a tailored algorithm to learn graphs under privacy constraints. Our algorithm follows the communication protocol of FL, where model updates instead of raw data are transmitted to a central server to learn all graphs collaboratively. The convergence analysis of the proposed algorithm is also provided.
- Extensive experiments with synthetic and real-world data are conducted to validate our framework, and the results show that our approach can effectively learn graphs under data silos.

Organization: The rest of this paper is organized as follows. We start with background information and the problem of concern in Section II. The proposed model for learning graphs in the target scenario is presented in III, followed by the corresponding algorithm in Section IV. Experimental setups and results are provided in Section V. Finally, concluding remarks are presented in Section VI.

Notations: Throughout this paper, vectors, matrices, and sets are written in bold lowercase letters, bold uppercase letters, and calligraphic uppercase letters, respectively. Given a vector $\mathbf{y}$ and matrix $\mathbf{Y}$, $\mathbf{y}[i]$ and $\mathbf{Y}[ij]$ are the i-th entry of $\mathbf{y}$ and the (i,j) entry of $\mathbf{Y}$. Besides, $\mathbf{1}$, $\mathbf{0}$, and $\mathbf{I}$ represent all-one vectors, all-zero vectors, and identity matrices, respectively. For a vector or matrix, the ℓ_1, ℓ_2, ℓ_∞, and Frobenius norm are represented by $\|\cdot\|_1$, $\|\cdot\|_2$, $\|\cdot\|_\infty$, $\|\cdot\|_{\mathrm{F}}$, respectively. Moreover, $\|\cdot\|_{\mathrm{F,off}}$ denotes the Frobenius norm of off-diagonal elements of a matrix, and $\mathrm{diag}(\cdot)$ means converting a vector to a diagonal matrix. The notations $\circ, \dagger, \mathrm{Tr}(\cdot)$ stand for Hadamard product, pseudo inverse, and trace operator, respectively. For a set $\mathcal{Y}$, $\mathrm{conv}\,[\mathcal{Y}]$ is the affine and convex hulls of $\mathcal{Y}$. Finally, $\mathbb{R}$ and $\mathbb{S}$ represent the domain of real values and symmetric matrices whose dimensions depend on the context.

II. Background and Problem Statement

A. GSP Background

We consider undirected graphs with non-negative weights and no self-loops. For such a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with d vertices, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of vertices and edges, respectively, its adjacency matrix $\mathbf{A} \in \mathbb{S}^{d\times d}$ is a symmetric matrix with zero diagonal entries and non-negative off-diagonal entries. The Laplacian matrix of $\mathcal{G}$ is $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [32], where the degree matrix $\mathbf{D} \in \mathbb{S}^{d\times d}$ is a diagonal matrix satisfying $\mathbf{D}[ii] = \sum_{j=1}^{d} \mathbf{A}[ij]$. The matrices $\mathbf{A}$ and $\mathbf{L}$ encode the topology of $\mathcal{G}$ since they have a one-to-one relationship. We study the graph signal $\mathbf{x} = [\mathbf{x}[1], \ldots, \mathbf{x}[d]]^\top \in \mathbb{R}^d$ associated with $\mathcal{G}$, where $\mathbf{x}[i]$ is the signal value of node $i \in \mathcal{V}$. The smoothness of $\mathbf{x}$ over $\mathcal{G}$ is defined as follows.

Definition 1. *(Smoothness [10]). Given a graph signal* $\mathbf{x}$ *and a graph* $\mathcal{G}$ *whose Laplacian matrix and adjacency matrix are* $\mathbf{L}$ *and* $\mathbf{A}$*, respectively, the smoothness of* $\mathbf{x}$ *over* $\mathcal{G}$ *is*

$$\mathbf{x}^\top \mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i,j} \mathbf{A}[ij]\left(\mathbf{x}[i] - \mathbf{x}[j]\right)^2. \tag{1}$$

The Laplacian quadratic form (1) is known as the Dirichlet energy, which quantifies how much the signal $\mathbf{x}$ changes w.r.t. $\mathcal{G}$. A small value of (1) indicates limited signal variability, meaning that $\mathbf{x}$ is smooth over the corresponding graph [10].

B. Graph Learning From Smooth Signals

Given N observations $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{d\times N}$, smoothness-based GL aims to infer the underlying graph topology $\mathcal{G}$ under the assumption that the signals $\mathbf{X}$ are smooth over $\mathcal{G}$. Formally, the problem is written as

$$\min_{\mathbf{L}\in\mathcal{L}} \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n^\top \mathbf{L}\mathbf{x}_n - \alpha \mathbf{1}^\top \log\left(\mathrm{diag}(\mathbf{L})\right) + \beta\|\mathbf{L}\|_{\mathrm{F,off}}^2, \tag{2}$$

where the first term is to quantify the smoothness of $\mathbf{X}$ over $\mathcal{G}$, and the last two terms are regularizers that endow $\mathcal{G}$ with desired properties. The first and second terms of the regularizers control node connectivity and edge sparsity of the learned graph [11], where α and β are predefined constants.

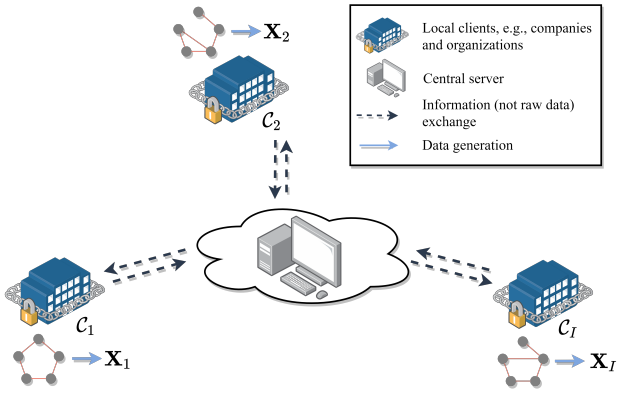


Fig. 1: The illustration of the target scenario. The clients $\mathcal{C}_1, \dots \mathcal{C}_I$ store graphs signals $\mathbf{X}_1, \dots, \mathbf{X}_I$ generated from I distinct but related graphs. The data are not allowed to leave their clients, but all clients can commute with a central server.

We use Laplacian matrix $\mathbf{L}$ to represent graph topology of $\mathcal{G}$, which lies in the following set $\mathcal{L}$

$$\mathcal{L} \triangleq \left\{\mathbf{L} : \mathbf{L} \in \mathbb{S}^{d\times d}, \mathbf{L}\mathbf{1} = \mathbf{0}, \mathbf{L}[ij] \leq 0 \text{ for } i \neq j\right\}. \quad (3)$$

Based on (1), problem (2) can be rephrased as

$$\min_{\mathbf{A}\in\mathcal{A}} \frac{1}{2N}\|\mathbf{A} \circ \mathbf{C}\|_{1,1} - \alpha \mathbf{1}^\top \log(\mathbf{A}\mathbf{1}) + \beta\|\mathbf{A}\|_{\mathrm{F}}^2, \quad (4)$$

where $\|\cdot\|_{1,1}$ is the element-wise ℓ_1 norm of a matrix. Besides, $\mathbf{C} \in \mathbb{R}^{d\times d}$ is a pairwise distance matrix defined as

$$\mathbf{C}[ij] = \|\widetilde{\mathbf{x}}_i - \widetilde{\mathbf{x}}_j\|_2^2, \quad (5)$$

where $\widetilde{\mathbf{x}}_i \in \mathbb{R}^N$ is the i-th row vector of $\mathbf{X}$. Similarly, $\mathcal{A}$ is the set containing all adjacency matrices,

$$\mathcal{A} = \left\{\mathbf{A} : \mathbf{A} \in \mathbb{S}^{d\times d}, \mathbf{A} \geq 0, \mathrm{diag}(\mathbf{A}) = \mathbf{0}\right\}, \quad (6)$$

where $\mathbf{A} \geq 0$ means that all elements of $\mathbf{A}$ are non-negative. By the definition of $\mathcal{A}$, the number of free variables of $\mathbf{A}$ is $p := \frac{d(d-1)}{2}$ [11]. For simplicity, we define a vector $\mathbf{w} \in \mathbb{R}^p$ whose elements are the upper triangle variables of $\mathbf{A}$. Then, problem (4) can be rewritten into a vector form as

$$\min_{\mathbf{w}\geq 0} \frac{1}{N}\mathbf{z}^\top \mathbf{w} - \alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}) + 2\beta\|\mathbf{w}\|_2^2, \quad (7)$$

where $\mathbf{S}$ is a linear operator satisfying $\mathbf{S}\mathbf{w} = \mathbf{A}\mathbf{1}$, and $\mathbf{z}$ is the vector form of the upper triangle elements of $\mathbf{C}$[1].

C. Problem Statement

As shown in Fig.1, suppose there are I clients $\mathcal{C}_1, \dots, \mathcal{C}_I$, and the i-th client stores signals $\mathbf{X}_i \in \mathbb{R}^{d\times N_i}$ generated from the graph $\mathcal{G}_i$, where N_i is the size of the i-th dataset. All graphs are defined over the same node set, and the data are prohibited from leaving the client where they are stored due to factors such as privacy concerns. However, the clients can exchange information with a central server. We assume that (i) graph signals $\mathbf{X}_i$ are smooth over $\mathcal{G}_i$, and (ii) $\mathcal{G}_1, \dots, \mathcal{G}_I$ may be heterogeneous, meaning that the corresponding graph signals could be non-IID. Our goal is to infer the graphs from $\mathbf{X}_1, \dots, \mathbf{X}_I$ in the case of data silos.

[1]The pairwise distance vector $\mathbf{z}$ is calculated from $\mathbf{x}_1, \dots, \mathbf{x}_N$. Thus, $\mathbf{z}$ is also referred to as observation data in the following sections.

III. Model Formulation

In this section, we first propose an auto-weighted MGL model to learn graphs in the target scenario, and then analyze its estimation error bound.

A. Basic Formulation

We assume that I local graphs, which are denoted as $\mathbf{w}_1, \dots, \mathbf{w}_I \in \mathbb{R}^p$, have some common structures named consensus graphs $\mathbf{w}_{\mathrm{con}} \in \mathbb{R}^p$. Intuitively, the consensus graph should be close to local graphs since they share common edges. Furthermore, the consensus graph should also be sparse to remove redundant noisy edges [33]. Thus, we learn I local graphs and the consensus graph jointly via

$$\begin{aligned}\min_{\mathbf{w}_i, \mathbf{w}_{\mathrm{con}}\in\mathcal{W}} &\sum_{i=1}^{I} \underbrace{\frac{1}{N_i}\mathbf{z}_i^\top \mathbf{w}_i - \alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}_i) + 2\beta\|\mathbf{w}_i\|_2^2}_{g_i(\mathbf{w}_i)} \\ &+ \lambda\nu \sum_{i=1}^{I}\|\mathbf{w}_i - \mathbf{w}_{\mathrm{con}}\|_2 + \lambda\|\mathbf{w}_{\mathrm{con}}\|_1 \\ = \min_{\mathbf{w}_i, \mathbf{w}_{\mathrm{con}}\in\mathcal{W}} & G(\mathbf{W}) + \lambda R(\mathbf{W}) \end{aligned} \quad (8)$$

where $\mathbf{z}_i$ is the observed data in client $\mathcal{C}_i$, $\mathcal{W} := \{\mathbf{w} : \mathbf{w} \geq 0\}$, $\mathbf{W} = [\mathbf{w}_1, \dots, \mathbf{w}_I, \mathbf{w}_{\mathrm{con}}] \in \mathbb{R}^{p\times(I+1)}$, $G(\mathbf{W}) := \sum_{i=1}^{I} g_i(\mathbf{w}_i)$, and $R(\mathbf{W}) := \nu\sum_{i=1}^{I}\|\mathbf{w}_i - \mathbf{w}_{\mathrm{con}}\|_2 + \|\mathbf{w}_{\mathrm{con}}\|_1$. Our model consists of two parts. The first part $G(\mathbf{W})$ represents that local clients utilize the traditional smoothness-based model (7) to learn graphs $\mathbf{w}_1, \dots, \mathbf{w}_I$ from locally stored data. The second term $R(\mathbf{W})$ is the regularizer that topologically connects local graphs via the consensus graph. Specifically, the first term $\nu\sum_{i=1}^{I}\|\mathbf{w}_i - \mathbf{w}_{\mathrm{con}}\|_2$ of $R(\mathbf{W})$ measures difference between local graphs and the consensus graph. We add the ℓ_1 norm term because $\mathbf{w}_{\mathrm{con}}$ is expected to be sparse. The constants ν and λ are predefined parameters. The parameters of problem (8) are α, β, ν, and λ. However, as stated in [11], tuning the importance of the log-degree term w.r.t. the other graph terms has a scaling effect. We can fix α and search for the other parameters. Therefore, the free parameters of our model are β, ν, and λ.

The proposed model (8) enjoys the following advantages. Firstly, our model provides a way to learn local personalized graphs $\mathbf{w}_1, \dots, \mathbf{w}_I$ jointly. Compared to learning graphs independently, our model utilizes the information from all datasets, which could boost learning performance. Secondly, we learn a personalized graph for each client, which alleviates the bias of learning a single global graph using all data in the case of data heterogeneity. Lastly, unlike most PFL methods that only learn personalized local models [18], our model learns a consensus graph reflecting global information.

B. Auto-weighted Multiple Graph Learning

At first glance, our model (8) treats I local graphs equally since they share the same weight in $\nu\sum_{i=1}^{I}\|\mathbf{w}_i - \mathbf{w}_{\mathrm{con}}\|_2$. This is unreasonable because the similarity between the consensus graph and different local graphs could vary widely. Fortunately, we will show that our model (8) can implicitly assign appropriate weights to local graphs, which we termed

contribution weights, based on their similarity to $\mathbf{w}_{\text{con}}$ via the inverse distance weighting schema [34].

Specifically, if we solve (8) by alternately updating $\mathbf{w}_1, \ldots, \mathbf{w}_I$ and $\mathbf{w}_{\text{con}}$—which is exactly how we solve it in Section IV—when we fix the consensus graph $\mathbf{w}_{\text{con}}$, the subproblem of updating $\mathbf{w}_i$ is

$$\min_{\mathbf{w}_i \in \mathcal{W}} \; g_i(\mathbf{w}_i) + \rho \|\mathbf{w}_i - \mathbf{w}_{\text{con}}\|_2, \tag{9}$$

where we let $\rho = \lambda\nu$ for simplicity. We take the derivative of the Lagrange function of (9) and set it to zero, which yields

$$\rho\widetilde{\gamma}_i \frac{\partial \|\mathbf{w}_i - \mathbf{w}_{\text{con}}\|_2^2}{\partial \mathbf{w}_i} + \frac{\partial g_i(\mathbf{w}_i)}{\partial \mathbf{w}_i} - \frac{\partial \boldsymbol{\theta}_i^\top \mathbf{w}_i}{\partial \mathbf{w}_i} = \mathbf{0}, \tag{10}$$

where $\boldsymbol{\theta}_i \in \mathbb{R}^p$ is the Lagrange multiplier, and

$$\widetilde{\gamma}_i = \frac{1}{2\|\mathbf{w}_i - \mathbf{w}_{\text{con}}\|_2}. \tag{11}$$

Note that (11) depends on $\mathbf{w}_i$, meaning that $\widetilde{\gamma}_i$ and $\mathbf{w}_i$ are coupled with each other. However, if we set $\widetilde{\gamma}_i$ stationary, (10) is the solution to the following problem

$$\min_{\mathbf{w}_i \in \mathcal{W}} \; g_i(\mathbf{w}_i) + \rho\widetilde{\gamma}_i \|\mathbf{w}_i - \mathbf{w}_{\text{con}}\|_2^2. \tag{12}$$

After solving (12), we can use the obtained $\mathbf{w}_i$ to update the weight $\widetilde{\gamma}_i$ via (11). Therefore, we can alternately update $\mathbf{w}_i$ and $\widetilde{\gamma}_i$ to solve (9). Similarly, it is not difficult to check that we can update $\mathbf{w}_{\text{con}}$ using the same strategy as updating $\mathbf{w}_i$.

Combing the alternative updates of $\mathbf{w}_i, \mathbf{w}_{\text{con}}$, and $\widetilde{\gamma}_i$, the basic formulation (8) is rephrased as

$$\begin{aligned} \min_{\mathbf{w}_i, \mathbf{w}_{\text{con}}} & \sum_{i=1}^{I} \frac{1}{N_i} \mathbf{z}_i^\top \mathbf{w}_i - \alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}_i) + 2\beta \|\mathbf{w}_i\|_2^2 \\ & + \frac{\rho}{2} \sum_{i=1}^{I} \gamma_i \|\mathbf{w}_i - \mathbf{w}_{\text{con}}\|_2^2 + \lambda \|\mathbf{w}_{\text{con}}\|_1 \\ \text{s.t. } & \mathbf{w}_i, \mathbf{w}_{\text{con}} \in \mathcal{W}, \gamma_i = \frac{1}{\|\mathbf{w}_i - \mathbf{w}_{\text{con}}\|_2}. \end{aligned} \tag{13}$$

One merit of solving (8) via the reformulation (13) is that it naturally produces a contribution weight γ_i for the local graph $\mathbf{w}_i$. The weight value is determined by the similarity between $\mathbf{w}_i$ and $\mathbf{w}_{\text{con}}$. For the $\mathbf{w}_i$ close to $\mathbf{w}_{\text{con}}$, a larger γ_i is assigned to the corresponding term, increasing the contribution of the i-th local graph to the consensus graph. On the contrary, the local graph far from the consensus graph obtains a small γ_i. Thus, our model can implicitly and automatically assign appropriate weights to local clients based on their similarity to the consensus graph.

C. Theoretical Analysis

Let $\mathbf{W}^* = [\mathbf{w}_1^*, \ldots, \mathbf{w}_I^*, \mathbf{w}_{\text{con}}^*] \in \mathbb{R}^{p \times (I+1)}$ be the true graphs and $\widehat{\mathbf{W}} = [\widehat{\mathbf{w}}_1, \ldots, \widehat{\mathbf{w}}_I, \widehat{\mathbf{w}}_{\text{con}}] \in \mathbb{R}^{p \times (I+1)}$ be the estimated graphs of our model (8). We aim to derive the estimation error bound of our proposed multiple graph estimator. Before conducting the analysis, we make the following assumptions.

Assumption 1. *The observed signals, as well as the corresponding pairwise distance vectors $\mathbf{z}_i$ are bounded, i.e., there exists a constant C_z such that $\|\mathbf{z}_i\|_2 \leq C_z$ for $i = 1, \ldots, I$.*

Assumption 2. *Let $h(\mathbf{w}) = -\alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}) + 2\beta\|\mathbf{w}\|_2^2$. The gradient of $h(\mathbf{w})$ at the real graph is bounded, i.e., there exists a constant C_h such that $\|\nabla h(\mathbf{w}_i^*)\|_2 \leq C_h$ for $i = 1, \ldots, I$.*

Assumption 1 naturally holds in the real world since unbounded signals do not make sense. Assumption 2 holds when the true graph has limited edge weights and no isolated nodes, which is common in the real world. Moreover, without loss of generality, we assume the data sizes of I graphs are the same, i.e., $N_i = N$ for $i = 1, \ldots, I$. The analysis can be easily generalized to the case where data sizes vary across different clients. We further suppose that the obtained data vector (pairwise distance vector) $\mathbf{z}_i$ can be written as

$$\mathbf{z}_i[j] = \mathbf{z}_i^*[j] + \mathbf{e}_i[j], \; j = 1, \ldots, p, \tag{14}$$

where $\mathbf{z}_i^*$ is the true data vector, and $\mathbf{e}_i$ is the error vector caused by factors such as noisy measurements. In our analysis, we assume that $\mathbf{e}_i \sim \mathcal{N}(\mathbf{0}, \sigma_e^2 \mathbf{I})$. Then, we present the estimation error bound of the proposed graph estimator as stated in the following theorem.

Theorem 1. *Under Assumptions 1 and 2, given δ and $\nu > 0$, let λ satisfy*

$$\lambda \geq \frac{C_z\sqrt{I} + \sigma_e\sqrt{pI + \delta}}{C_r N}, \tag{15}$$

where $C_r := \nu\sqrt{I\omega_{\max}(\mathbf{L}_m)} + \sqrt{p}$,

$$\mathbf{L}_m \in \mathbb{R}^{(I+1)\times(I+1)} = \begin{bmatrix} 1 & 0 & \cdots & 0 & -1 \\ 0 & 1 & \cdots & 0 & -1 \\ \vdots & \ddots & \cdots & 0 & -1 \\ 0 & 0 & \cdots & 1 & -1 \\ -1 & -1 & -1 & -1 & I \end{bmatrix}, \tag{16}$$

and $\omega_{\max}(\mathbf{L}_m)$ is the maximum eigenvalue of $\mathbf{L}_m$. Then, we have probability at least $1 - \exp\left(-\frac{1}{2}\left(\delta - pI\log\left(1 + \frac{\delta}{pI}\right)\right)\right)$ such that

$$\left\|\widehat{\mathbf{W}} - \mathbf{W}^*\right\|_{\text{F}} \leq \frac{C_r\lambda}{\beta} + \frac{C_h\sqrt{I}}{2\beta}. \tag{17}$$

Proof. See Appendix A for details. □

Theorem 1 characterizes the estimation error bound of our proposed model w.r.t. some key factors, such as graph size d (the number of the free variables p), data size N, the number of local graphs I, and measurement noise level σ_e. The theorem states that, if the weight λ before the regularizer $R(\mathbf{W})$ is selected appropriately, the estimation error of our model is bounded by the r.h.s. of (17) with high probability. The upper bound consists of two parts. The first part is related to data sizes. When N goes to infinity, (15) indicates that λ could be small enough that the first part decreases to zero. The second part is determined by the regularizer $h(\mathbf{w}^*)$ of the basic graph learning model (7). This can be regarded as a systematic error since $h(\mathbf{w})$ is chosen using prior knowledge, and the error will not decrease as N increases. We should mention that our model is flexible, and one can choose any $h(\mathbf{w})$ as the regularizer of local models to bring desired properties to the learned graphs. For the well-selected regularizer, the gradient

of $h(\mathbf{w})$ at $\mathbf{w}^*$ will be bounded tightly, and we can obtain a small upper bound on the estimation error.

D. Connections to Existing MGL Models

In the literature, there have been some works that learn multiple graphs based on the assumption of common structures. These works differ mainly in how they describe structural relationships among multiple graphs. To avoid notational confusion, we use different notations to interpret existing works. We let $\mathbf{K} \in \mathbb{R}^{d\times d}$ denote the common graph structures and $\mathbf{U}_1, \ldots, \mathbf{U}_I \in \mathbb{R}^{d\times d}$ represent the unique structures of I local graphs. Under these notations, the graph regularizer $R(\mathbf{W})$ in our model (8) can be roughly summarized as (not exactly equivalent to) $\|\mathbf{K}\|_{1,1} + \nu\sum_{i=1}^{I}\|\mathbf{U}_i\|_{\mathrm{F}}$. Unlike our model, the work [35] defines the regularizer as $\|\mathbf{K}\|_{1,1} + \nu\|\mathbf{U}\|_{1,r}$, where $r \in [1, +\infty)$, $\|\mathbf{U}\|_{1,r} = \sum_{i,j=1}^{d}\|\mathbf{U}[ij]\|_r$, and $\mathbf{U}[ij] = [\mathbf{U}_1[ij], \ldots, \mathbf{U}_I[ij]]^\top \in \mathbb{R}^I$. The term $\|\mathbf{U}\|_{1,r}$ is a group lasso regularizer that can capture common sparsity structure among all local graphs. The second work [36] assumes that the common structure $\mathbf{K}$ is the average of all local graphs, i.e., $\mathbf{K} = \frac{1}{I}\sum_{i=1}^{I}\mathbf{A}_i, \mathbf{U}_i = \mathbf{A}_i - \mathbf{K}$. A regularizer is then designed as $\|\mathbf{K}\|_{1,1} + \nu\sum_{i=1}^{I}\|\mathbf{U}_i\|_{1,1}$. Unlike our work, [37] proposes a regularizer $\|\mathbf{K}\|_{1,1} + \nu\sum_{i=1}^{I}\|\mathbf{U}_i\|_{\mathrm{F}}^2$. Moreover, the local graph learning model of [37] is different from ours, and [37] requires all graphs, including the consensus graph, are represented by Laplacian matrices. Recently, [38] designs two regularizers, i.e., $\sum_{i=1}^{I}\|\mathbf{U}_i\|_{1,1}$ and the other similar to [35].

In summary, our model differs from existing works in the following aspects. (i) The regularizer $R(\mathbf{W})$ is different from existing works, which can implicitly assign a contribution weight for each local graph. This is used for the first time in the multiple graph learning problem. (ii) Our model and [37] learn multiple graphs based on the smoothness assumption (6), while [35], [36] are based on the GGMs. (iii) We further consider the problem of learning graphs in the case of data silos, which has not been explored in the existing GL-related works.

IV. Model Optimization

In this section, we propose an algorithm with provable convergence analysis for solving (8) in the target scenario.

A. The Proposed Algorithm

According to the reformulation (13), our algorithm consists of two steps: (i) updating $\mathbf{w}_i$ in the local client $\mathcal{C}_i$ and (ii) updating $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_I]^\top \in \mathbb{R}^I$ and $\mathbf{w}_{\mathrm{con}}$ in the central server. The complete flow is displayed in Algorithm 1.

Updating $\mathbf{w}_i$ in the local client $\mathcal{C}_i$: This update corresponds to lines 3-11 in Algorithm 1. In the t-th communication round, $\mathcal{C}_i$ first receives $\mathbf{w}_{\mathrm{con}}^{(t)}$ and $\gamma_i^{(t)}$ from the central server. Let $f_i^{(t)}(\mathbf{w}_i, \mathbf{w}_{\mathrm{con}}) := g_i(\mathbf{w}_i) + \frac{\rho\gamma_i^{(t)}}{2}\|\mathbf{w}_i - \mathbf{w}_{\mathrm{con}}\|_2^2$ [2], and the sub-problem of $\mathcal{C}_i$ becomes

$$\begin{aligned}\mathbf{w}_i^{(t+1)} =& \underset{\mathbf{w}_i\in\mathcal{W}}{\operatorname{argmin}} f_i^{(t)}(\mathbf{w}_i, \mathbf{w}_{\mathrm{con}}^{(t)})\\ =& \underset{\mathbf{w}_i\in\mathcal{W}}{\operatorname{argmin}} \frac{1}{N_i}\mathbf{z}_i^\top\mathbf{w}_i - \alpha\mathbf{1}^\top\log(\mathbf{S}\mathbf{w}_i + \zeta\mathbf{1})\\ &+ 2\beta\|\mathbf{w}_i\|_2^2 + \frac{\rho\gamma_i^{(t)}}{2}\|\mathbf{w}_i - \mathbf{w}_{\mathrm{con}}^{(t)}\|_2^2, \end{aligned} \tag{18}$$

where ζ is a small enough constant to avoid zero node degree. We use the accelerated projected gradient descent algorithm to update $\mathbf{w}_i$. At the beginning of local updates, $\mathcal{C}_i$ first initializes $\mathbf{w}_i^{(t,0)} = \mathbf{w}_i^{(t-1,K_i^{(t-1)})}$ and $\mathbf{w}_i^{(t,-1)} = \mathbf{w}_i^{(t-1,K_i^{(t-1)}-1)}$, where $K_i^{(t)}$ is the number of local loops of $\mathcal{C}_i$ in the t-th outer iteration. Besides, $\mathbf{w}_i^{(t,k)}$ represents the updated graph of $\mathcal{C}_i$ in the t-th outer iteration and the k-th local loop. We then update the i-th local graph $K_i^{(t)}$ times by

$$\mathbf{w}_{i,\mathrm{ex}}^{(t,k)} = \mathbf{w}_i^{(t,k)} + \xi\left(\mathbf{w}_i^{(t,k)} - \mathbf{w}_i^{(t,k-1)}\right) \tag{19}$$

$$\begin{aligned}\breve{\mathbf{w}}_i^{(t,k+1)} = \mathbf{w}_{i,\mathrm{ex}}^{(t,k)} - \eta_w\Big(&\nabla g_i(\mathbf{w}_{i,\mathrm{ex}}^{(t,k)})\\ &+ \rho\gamma_i^{(t)}\left(\mathbf{w}_{i,\mathrm{ex}}^{(t,k)} - \mathbf{w}_{\mathrm{con}}^{(t)}\right)\Big)\end{aligned} \tag{20}$$

$$\mathbf{w}_i^{(t,k+1)} = \mathrm{Proj}_{\mathcal{W}}\left(\breve{\mathbf{w}}_i^{(t,k+1)}\right), \tag{21}$$

where $\xi \in [0, 1)$ is an momentum weight. In (20), $\nabla g_i(\mathbf{w})$ is calculated as $\frac{1}{N_i}\mathbf{z}_i - \alpha\mathbf{S}^\top\left(\frac{1}{\mathbf{S}\mathbf{w}+\zeta\mathbf{1}}\right) + 4\beta\mathbf{w}$. Moreover, η_w is the stepsize, the choice of which will be discussed in the next subsection. The operator $\mathrm{Proj}_{\mathcal{W}}(\cdot)$ means projecting variables into $\mathcal{W}$. When local updates finish, $\mathcal{C}_i$ sends $\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t,K_i^{(t)})}$ to the central server. Note that the local update of $\mathbf{w}_i$ is inexact in one communication round since we run (19)-(21) for $K_i^{(t)}$ times without convergence. We use inexact updates because we aim to update all local graphs synchronously. Due to system heterogeneity, the time required for each client to update $\mathbf{w}_i$ until convergence may vary widely. It takes longer if the central server waits for all clients to update their graphs until convergence.

Updating $\mathbf{w}_{\mathrm{con}}$ and $\boldsymbol{\gamma}$ in the central server: The updates correspond to lines 12-14 in Algorithm 1. After the central server receives all the updated local graphs, $\mathbf{w}_1^{(t+1)}, \ldots, \mathbf{w}_I^{(t+1)}$, we update $\mathbf{w}_{\mathrm{con}}^{(t+1)}$ by solving the following problem

$$\begin{aligned}\mathbf{w}_{\mathrm{con}}^{(t+1)} &= \underset{\mathbf{w}_{\mathrm{con}}\in\mathcal{W}}{\operatorname{argmin}} \sum_{i=1}^{I} f_i^{(t)}(\mathbf{w}_i^{(t+1)}, \mathbf{w}_{\mathrm{con}}) + \lambda\|\mathbf{w}_{\mathrm{con}}\|_1\\ &= \underset{\mathbf{w}_{\mathrm{con}}\in\mathcal{W}}{\operatorname{argmin}} \sum_{i=1}^{I}\frac{\rho\gamma_i^{(t)}}{2}\|\mathbf{w}_i^{(t+1)} - \mathbf{w}_{\mathrm{con}}\|_2^2 + \lambda\|\mathbf{w}_{\mathrm{con}}\|_1.\end{aligned} \tag{22}$$

The problem is equivalent to

$$\begin{aligned}\mathbf{w}_{\mathrm{con}}^{(t+1)} =& \underset{\mathbf{w}_{\mathrm{con}}\in\mathcal{W}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{w}_{\mathrm{con}} - \frac{\sum_{i=1}^{I}\gamma_i^{(t)}\mathbf{w}_i^{(t+1)}}{\sum_{i=1}^{I}\gamma_i^{(t)}}\right\|_2^2\\ &+ \frac{\lambda}{\rho\sum_{i=1}^{I}\gamma_i^{(t)}}\|\mathbf{w}_{\mathrm{con}}\|_1.\end{aligned} \tag{23}$$

By defining $C_\gamma^{(t)} := \sum_{i=1}^{I}\gamma_i^{(t)}$ and $\mu^{(t)} := \frac{\lambda}{C_\gamma^{(t)}\rho}$, we obtain

$$\mathbf{w}_{\mathrm{con}}^{(t+1)} = \mathrm{prox}_{\mu^{(t)}\|\cdot\|_1}\left(\frac{\sum_{i=1}^{I}\gamma_i^{(t)}\mathbf{w}_i^{(t+1)}}{C_\gamma^{(t)}}\right), \tag{24}$$

[2] $f_i^{(t)}(\mathbf{w}_i, \mathbf{w}_{\mathrm{con}}^{(t)})$ is a function of $\mathbf{w}_i$ with fixed $\mathbf{w}_{\mathrm{con}}^{(t)}$. Similarly, $f_i^{(t)}(\mathbf{w}_i^{(t+1)}, \mathbf{w}_{\mathrm{con}})$ is a function of $\mathbf{w}_{\mathrm{con}}$ with fixed $\mathbf{w}_i^{(t+1)}$.

Algorithm 1 The algorithm for solving (8)

Input: α, β, ν, ξ, λ, and signals $\mathbf{X}_1, \ldots, \mathbf{X}_I$

Initialize $\gamma_i^{(0)} = 1/I$, $\mathbf{w}_{\text{con}}^{(0)} = \mathbf{w}_i^{(0)} = \mathbf{w}_i^{(-1,0)} = \mathbf{w}_i^{(-1,-1)}$ for $i = 1, \ldots, I$, and let $K_i^{(-1)} = 0$
for $t = 0, \ldots, T-1$ **do**
  // Update $\mathbf{w}_1, \ldots, \mathbf{w}_I$ in parallel in local clients
  for $i = 1, \ldots, I$ in parallel **do**
    Receive $\gamma_i^{(t)}$ and $\mathbf{w}_{\text{con}}^{(t)}$ from the central server
    Initialize $\mathbf{w}_i^{(t,0)} = \mathbf{w}_i^{(t-1,K_i^{(t-1)})}$ and $\mathbf{w}_i^{(t,-1)} = \mathbf{w}_i^{(t-1,K_i^{(t-1)}-1)}$
    for $k = 0, \ldots, K_i^{(t)} - 1$ **do**
      Update $\mathbf{w}_i^{(t,k+1)}$ using (19)-(21)
    end for
    Let $\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t,K_i^{(t)})}$ and send it to central server
  end for
  // Update $\mathbf{w}_{\text{con}}$ and $\boldsymbol{\gamma}$ in central server
  Update $\mathbf{w}_{\text{con}}^{(t+1)}$ using (24)
  Update $\gamma_i^{(t+1)}$ using (25), for $i = 1, \ldots, I$
  Send $\mathbf{w}_{\text{con}}^{(t+1)}$ and $\gamma_i^{(t+1)}$ to the i-th client $\mathcal{C}_i$
end for
return $\mathbf{w}_1^{(T)}, \ldots, \mathbf{w}_I^{(T)}$ and $\mathbf{w}_{\text{con}}^{(T)}$

where $\text{prox}_{\mu^{(t)}\|\cdot\|_1}(\cdot)$ is the proximal operator of ℓ_1 norm. It is observed from (24) that $\mathbf{w}_{\text{con}}^{(t+1)} \in \mathcal{W}$ since it is a combination of $\mathbf{w}_i^{(t+1)}$ with positive weights, and the proximal operator will not move the vector out of $\mathcal{W}$.

After obtaining $\mathbf{w}_{\text{con}}^{(t+1)}$, we update $\gamma_i^{(t+1)}$ as[3]

$$\gamma_i^{(t+1)} = \frac{1}{\left\|\mathbf{w}_i^{(t+1)} - \mathbf{w}_{\text{con}}^{(t+1)}\right\|_2}, \text{ for } i = 1, \ldots, I. \quad (25)$$

Finally, we send $\mathbf{w}_{\text{con}}^{(t+1)}$ and $\gamma_i^{(t+1)}$ back to the client $\mathcal{C}_i$.

B. Convergence Analysis

Before proceeding with the convergence analysis, let us first discuss the properties of the objective function.

Proposition 1. *The objective function $f_i^{(t)}(\mathbf{w}_i, \mathbf{w}_{\text{con}}^{(t)})$ is $L_i^{(t)}$-Lipschitz smooth w.r.t. $\mathbf{w}_i$ on $\widetilde{\mathcal{W}}$, where $\widetilde{\mathcal{W}} = \text{conv}[\cup_{0\le\xi\le1}\{\mathbf{y} + \xi(\mathbf{y} - \mathbf{y}')|| \ \mathbf{y}, \mathbf{y}' \in \mathcal{W}\}]$ is the feasible set extended by the momentum update, and $L_i^{(t)} := 4\beta + 2\alpha(d-1)/\zeta^2 + \rho\gamma_i^{(t)}$. Moreover, $f_i^{(t)}(\mathbf{w}_i, \mathbf{w}_{\text{con}}^{(t)})$ is $S_i^{(t)}$-strongly convex w.r.t. $\mathbf{w}_i$, where $S_i^{(t)} = 4\beta + \rho\gamma_i^{(t)}$.*

Proof. We place the proof in Appendix B. □

Proposition 1 points out that the objective function $f_i^{(t)}(\mathbf{w}_i, \mathbf{w}_{\text{con}}^{(t)})$ enjoys some nice properties, i.e., it is $L_i^{(t)}$-Lipschitz smooth and $S_i^{(t)}$-strongly convex, which are essential to the convergence of accelerated gradient descent algorithms [39]. Next, we further make the following assumptions.

Assumption 3. *The feasible set is convex and compact.*

[3]To avoid dividing by zero, we can update $\gamma_i^{(t+1)}$ as $\frac{1}{\left\|\mathbf{w}_i^{(t+1)} - \mathbf{w}_{\text{con}}^{(t+1)}\right\|_2 + \epsilon_\gamma}$, where ϵ_γ is a small enough constant.

Assumption 4. *The (constant) stepsize satisfies $\eta_w \le 1/L_{max}$, where L_{max} is the maximum value of $L_i^{(t)}$ for all i and t.*

Assumption 3 is common in the constrained optimization algorithms. Assumption 4 requires the (constant) stepsize to be bounded. Based on these assumptions, we conduct convergence analysis of our proposed algorithm as follows.

Theorem 2. *Based on Assumptions 3 and 4, in each communication round of Algorithm 1, the updated $\mathbf{w}_{\text{con}}$ and $\mathbf{w}_i$ monotonically decrease the objective function of the proposed model* (8) *until the solution converges to a local optimum of the problem* (8).

Proof. See Appendix C for details. □

The theorem reveals that as communication round t increases, the proposed algorithm will converge to a local optimum of problem (8). In Section V-B, we will experimentally test the convergence of the proposed algorithm.

C. Privacy Analysis

In our algorithm, each client $\mathcal{C}_i$ uses its private data $\mathbf{X}_i$ to update the local graph $\mathbf{w}_i$ by solving (18). After local updates, clients send model updates instead of raw data to the central server to update the consensus graph and weights. Following the paradigm of FL, all data is kept locally without any leakage during execution, which protects data privacy to a certain extent. Therefore, the proposed algorithm can learn graphs in the target scenario. However, inference attacks [40] and model inversion attacks [41] show that the updates sent by local clients may still reveal private information. To further prevent information leakage, some popular techniques such as differential privacy [42] may be helpful. This is beyond the scope of this study and will be left to future work.

D. Some Discussions

Differences from distributed learning algorithms: Our algorithm may look similar to distributed learning algorithms at first glance, but there are actually fundamental differences. (i) Our algorithm falls into the emerging field of federated learning, which aims to train models on siloed datasets under privacy constraints, while distributed learning aims to employ multiple clients (computing nodes) to accelerate training on a large but "flat" dataset [13]. (ii) In distributed learning, data are centrally stored and can be shuffled and balanced across clients. Thus, data of different clients are IID. In contrast, in federated learning, data are generated locally and may be non-IID. In our problem, the data are generated from heterogeneous graphs and are hence non-IID. We propose to learn personalized graphs to alleviate the impact of non-IID data. (iii) Distributed learning divides the dataset into multiple subsets uniformly and randomly, and data sizes across different computing nodes are close. However, in federated learning, all data are generated from local clients, which may be geographically and functionally diverse organizations, bringing widely varying data sizes of different clients.

Differences from existing FL algorithms: On the other hand, our algorithm differs from existing (personalized) FL algorithms in the following ways due to the specificity of our problem. (i) Unlike the algorithms of cross-device FL [17], [28], our algorithm learns graphs in a cross-silo FL scenario. Specifically, all clients, rather than a subset of clients, participate in local updates in each communication round. In addition, local clients utilize all stored data to update graphs instead of sampling a batch of data like many cross-device FL algorithms [17], [28]. (ii) The optimization problems corresponding to local and global updates are customized. For example, our algorithm updates the consensus graph by solving a ℓ_1 norm regularized quadratic optimization problem, whereas the global update of existing algorithms is a simple aggregation of local models [20], [28], [30]. (iii) Finally, compared to decentralized PFL algorithms [29], [31], our algorithm follows the "central server-clients" schema.

V. Experiments

In this section, we will test our proposed framework using both synthetic data and real-world data. First, some experimental setups are introduced.

A. Experimental Setups

1) Graph generation: We first generate Gaussian radial basis function (RBF) graphs $\mathcal{G}_0$ by following the method in [10]. Specifically, we generate 20 vertices whose coordinates are randomly in a unit square. Edge weights are calculated using $\exp(-\mathrm{dist}(i,j)^2/2\sigma_r^2)$, where $\mathrm{dist}(i,j)$ is the distance between vertices i and j, and $\sigma_r = 0.5$ is a kernel width parameter. We remove the edges whose weights are smaller than 0.7. Then, we keep a certain proportion—say q—of edges in $\mathcal{G}_0$ unchanged to form the consensus graph. Finally, we add $(1-q)|\mathcal{E}_0|$ edges randomly to the consensus graph to form heterogeneous graphs $\mathcal{G}_1, \ldots, \mathcal{G}_I$, where $|\mathcal{E}_0|$ is the number of edges in $\mathcal{G}_0$. We denote $1-q$ as the degree of heterogeneity.

2) Signal generation: For the i-th local graph $\mathcal{G}_i$, we generate N_i smooth signals from the distribution [10]

$$\mathbf{X}_{i,n} \sim \mathcal{N}\left(\mathbf{0}, (\mathbf{L}_i^*)^\dagger + \sigma_w^2 \mathbf{I}\right), \quad n = 1, \ldots, N_i, \tag{26}$$

where σ_w is noise scale and $\mathbf{L}_i^*$ is the real Laplacian matrix of $\mathcal{G}_i$, and $\mathbf{X}_{i,n}$ is the n-th data (column) of $\mathbf{X}_i$.

3) Evaluation metric: Four metrics are employed to evaluate the learned graphs, as listed in (27). The first three metrics are used to evaluate classification tasks since determining whether two vertices are connected can be regarded as a binary classification problem. Specifically, TP is the true positive rate, TN is the true negative rate, FP is the false positive rate and FN denotes false negative rate. The third metric, F1-score (FS) is the harmonic mean of $\mathrm{Precision}$ and Recall. The last metric is the relative error (RE), where $\widehat{\mathbf{w}}$ is the learned graph, and $\mathbf{w}^*$ is the groundtruth. The results of all metrics for local graphs are the average of I clients.

$$\begin{aligned} \mathrm{Precision} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \quad \mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \\ \mathrm{FS} &= \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FN}+\mathrm{FP}}, \quad \mathrm{RE} = \frac{\|\widehat{\mathbf{w}}-\mathbf{w}^*\|_2}{\|\mathbf{w}^*\|_2}. \end{aligned} \tag{27}$$

4) Baselines: We employ three categories of baselines. (i) We learn all local graphs independently (IGL), which enjoys full privacy as local clients do not release any data, including model updates. (ii) We use two traditional FL algorithm, FedAvg [17] and FedProx [43], to solve the following problem

$$\min_{\mathbf{w}\in\mathcal{W}} \frac{1}{N_{all}} \sum_{i=1}^{I} \mathbf{z}_i^\top \mathbf{w} - \alpha \mathbf{1}^\top \log\left(\mathbf{S}\mathbf{w} + \zeta \mathbf{1}\right) + 2\beta \|\mathbf{w}\|_2^2. \tag{28}$$

The model learns a global graph collaboratively using data from all clients regardless of data heterogeneity. (iii) We leverage two personalized FL algorithms, MRMTL [44] and Ditto [45], to jointly learn all local graphs by further considering data heterogeneity. See supplementary materials for more details on how they are applied to multiple graph learning. For our algorithm, we use PPGL-L and PPGL-C to represent the learned local and consensus graphs, respectively.

5) Determination of parameters: For our model, α is fixed as 1, and β is selected as the value achieving the best FS in IGL. Parameters ν and λ are determined by grid search in $[1, 100]$ and $[0.01, 1]$. On algorithmic side, we set $\xi = 0.9, K_i^{(t)} = K = 5, T = 50$. The stepsize η_w is 0.005, which is small enough to satisfy $\eta_w \leq 1/L_{max}$. Moreover, we let $I = 5$ and $\sigma_w = 0.1$ for synthetic data. All parameters of baselines are selected as those achieving the best FS values.

B. Synthetic Data

1) Data size: We first test the effect of data size. We fix q as 0.5 and assume that all clients have the same data size, i.e., $N_i = N, i = 1, \ldots, I$. We vary N from 20 to 100, and the results are shown in Table.I. It is observed that FedAvg and FedProx tend to obtain high Recall but low $\mathrm{Precision}$, meaning that the corresponding graphs contain more edges. As N increases, FedAvg yields the worst FS, which is a comprehensive metric for evaluating the learned graph topology. The reason is that FedAvg learns a global graph using data from all clients. However, local graphs are heterogeneous in our experiment. Therefore, the learned global graph is far from local graphs. In sharp contrast, IGL learns local graphs independently and achieves better performance than FedAvg when N is large. Our model also learns a personalized graph for each local client. Unlike IGL, we learn all local graphs jointly via a consensus graph, leading to better performance than IGL. Our model is also superior to Ditto and MRMTL—two PFL algorithms—since it is tailored for graph learning problems. Additionally, our model learns a consensus graph that captures the common structures of local graphs.

We also test our model with different data sizes for different clients. We set N_i to 20, 40, 60, 80, and 100 for $i = 1, \ldots, 5$, respectively. As illustrated in Fig.2, clients with larger N_i perform better than those with small N_i. However, our model can improve the performance of clients with small N_i because they can "borrow" information from the clients with large N_i.

Figure 3 provides the visualization of the learned graphs. We can observe that, compared with other methods, our model can effectively capture the common and specific structures of the real graph.

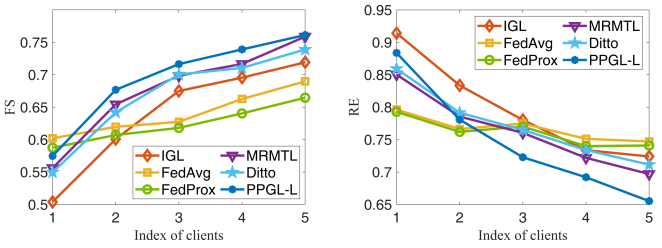
TABLE I: Performance of varying data sizes.

	$N = 20$				$N = 50$				$N = 100$			
	Precision ↑	Recall ↑	FS ↑	RE ↓	Precision ↑	Recall ↑	FS ↑	RE ↓	Precision ↑	Recall ↑	FS ↑	RE ↓
IGL	0.594	0.469	0.521	0.915	0.692	0.620	0.651	0.778	0.776	0.739	0.755	0.698
FedAvg	0.434	0.759	0.552	0.802	0.464	**0.792**	0.584	0.768	0.519	0.789	0.624	0.744
FedProx	0.423	**0.767**	0.544	**0.765**	0.523	0.728	0.606	0.718	0.623	0.703	0.655	0.690
MRMTL	0.522	0.551	0.547	0.881	0.647	0.688	0.663	0.756	0.738	0.789	0.760	0.680
Ditto	0.582	0.547	0.557	0.884	0.662	0.671	0.663	0.775	0.732	0.780	0.750	0.696
PPGL-L	0.547	0.579	0.561	0.911	0.625	0.760	0.678	0.731	0.698	**0.870**	0.774	**0.619**
PPGL-C	**0.636**	0.516	**0.566**	0.874	**0.781**	0.656	**0.710**	**0.701**	**0.881**	0.736	**0.800**	0.627

[1] In this paper, ↑ indicates that larger values are better, and ↓ indicates that smaller values are better.

TABLE II: Performance of varying data heterogeneity.

	$q = 0.3$				$q = 0.6$				$q = 0.9$			
	Precision ↑	Recall ↑	FS ↑	RE ↓	Precision ↑	Recall ↑	FS ↑	RE ↓	Precision ↑	Recall ↑	FS ↑	RE ↓
IGL	**0.701**	0.593	0.640	0.783	**0.736**	0.645	0.685	0.758	0.822	0.706	0.759	0.717
FedAvg	0.385	**0.841**	0.527	0.835	0.460	**0.867**	0.599	0.783	0.811	0.922	0.862	0.672
FedProx	0.437	0.720	0.543	0.770	0.569	0.750	0.645	0.716	0.874	0.863	0.868	0.641
MRMTL	0.662	0.691	0.674	0.761	0.690	0.730	0.707	0.753	0.816	0.794	0.807	0.663
Ditto	0.667	0.675	0.668	0.765	0.698	0.714	0.704	0.739	0.821	0.783	0.804	0.667
PPGL-L	0.637	0.736	**0.679**	**0.740**	0.661	0.776	**0.715**	**0.700**	0.800	0.812	0.805	0.635
PPGL-C	0.462	0.516	0.487	0.874	0.653	0.726	0.687	0.752	**0.825**	**0.927**	**0.873**	**0.634**



(a) (b)

Fig. 2: The results of varying data sizes of different clients. The data sizes $N_1, \ldots, N_5$ are 20, 40, 60, 80, and 100, respectively.

2) Data heterogeneity: We next explore the effect of data heterogeneity. In this experiment, we fix $N_i = 50$ for all clients and vary q from 0.3 to 0.9. As listed in Table II, when data heterogeneity is small (large q), FedAvg and FedProx achieve satisfactory performance since the consensus graph is close to local graphs. In this case, FedAvg and FedProx benefit from using all data to learn a global graph. However, with the increase in data heterogeneity, the personalized algorithms outperform FedAvg and FedProx. Moreover, our model outperforms Ditto and MRMTL due to the more reasonable consensus graph learning formulation. Our model obtains the best performance for all data heterogeneity since we learn personalized graphs and a consensus graph simultaneously.

3) The impact of adaptive weight schema: Then, we test the effectiveness of the method of adjusting weights. We conduct an ablation experiment where the adaptive weight schema is removed from our model and let all local clients share the same weight. The local graphs and consensus graph output by this model are denoted as "PPFL-L-w/o" and "PPFL-C-w/o", respectively. Two cases are considered, i.e., varying data sizes and varying data heterogeneity. For Case 1, we let q be 0.5 and set N_i to 20, 40, 60, 80, and 100 for $\mathcal{C}_1 - \mathcal{C}_5$, respectively. For Case 2, we fix $N_i = 100$ and first generate a graph $\mathcal{G}_1$ for $\mathcal{C}_1$. The graphs of $\mathcal{C}_2$-$\mathcal{C}_5$ are generated based on $\mathcal{G}_1$ with q equal to 0.8, 0.6, 0.4, and 0.2, respectively. The results are listed in Table III. Note that removing adaptive weight schema from our model will degrade performance, indicating the effectiveness of the proposed auto-weighted model.

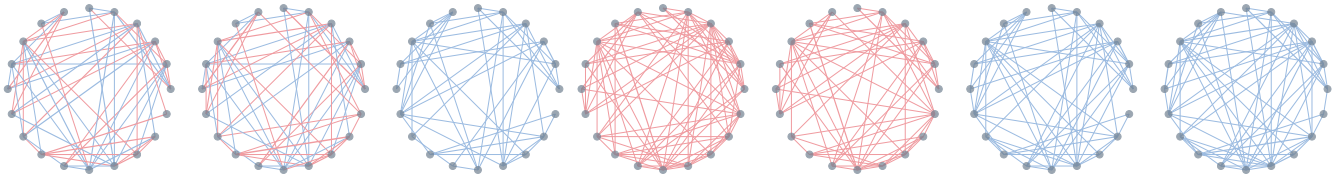
TABLE III: The results of removing adaptive weight schema.

	Methods	Precision ↑	Recall ↑	FS ↑	RE ↓
Case 1	PPFL-L	0.691	**0.773**	0.691	0.705
	PPFL-C	**0.743**	0.702	**0.722**	**0.689**
	PPFL-L-w/o	0.618	0.751	0.678	0.711
	PPFL-C-w/o	0.702	0.692	0.697	0.701
Case 2	PPFL-L	0.718	**0.875**	0.789	**0.605**
	PPFL-C	**0.888**	0.744	**0.810**	0.611
	PPFL-L-w/o	0.704	0.857	0.773	0.625
	PPFL-C-w/o	0.880	0.730	0.798	0.632

Moreover, the learned average weights of different clients are displayed in Fig.4, and two trends are observed. First, local graphs with large data sizes contribute more to the consensus graph. Second, local graphs close to the consensus graph obtain larger weights. Thus, the weight adjustment strategy can effectively allocate a reasonable weight for each client.

4) Parameter sensitivity: In this experiment, we set $N_i = 100$ and $q = 0.5$, and evaluate the learning performance of different λ and ν. As displayed in Fig.5, local graphs are less sensitive to the parameters than the consensus graph. When ν and λ are too large, the performance of the consensus graph degrades rapidly. However, there exist combinations of ν and λ that achieve the highest FS for both local graphs and the consensus graph. We need to grid search for the optimal combination of ν and λ, especially for consensus graphs.

5) Algorithm convergence: Finally, we test the convergence of the proposed algorithms. Fig.6 shows that our algorithms with different K can converge as stated in Theorem 2. When



(a) Ground truth (b) PPGL (c) IGL (d) FedAvg (e) FedProx (f) Ditto (g) MRMTL

Fig. 3: The graphs of $\mathcal{C}_1$ learned by different methods when $N_i = 100, q = 0.5$. In (a)-(b), red edges are those in the consensus (global) graph, while blue edges are only in the local graphs.

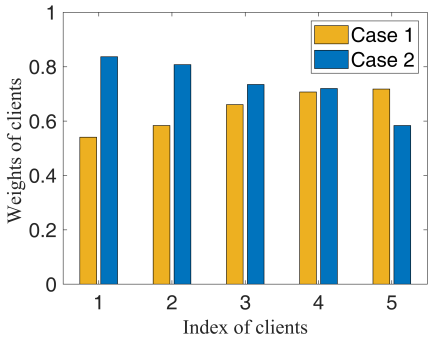
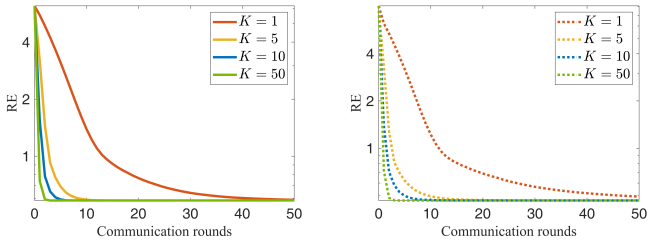
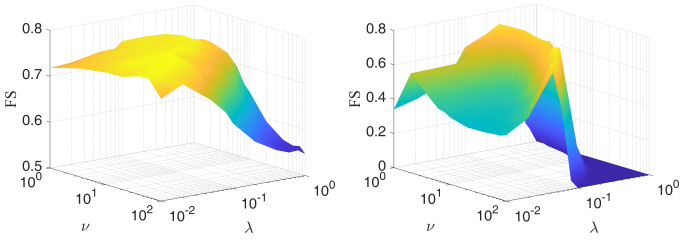


Fig. 4: The learned $\boldsymbol{\gamma}$ of different clients



(a) Local graphs (b) The consensus graph

Fig. 6: The convergence of the proposed algorithm.



(a) Local graphs (b) The consensus graph

Fig. 5: The results of parameter sensitivity.

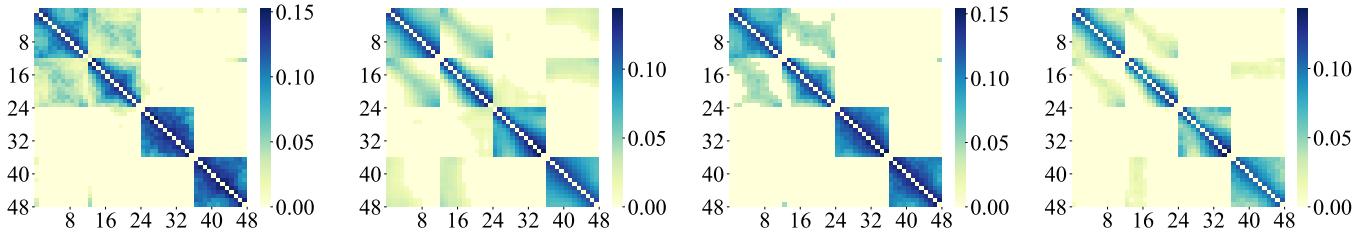
K is small, fewer local updates are performed. In this case, more communication rounds between local clients and the central server are required. On the other hand, when K is large, our algorithm only takes a few rounds to converge.

C. Real-world Data

1) COIL-20 data: We employ the COIL-20 dataset[4], which is a collection of gray-scale images including 20 objects taken from 360 degrees, to learn the relationships between these images. Each object has 72 images (five degrees an image) of size 32×32. We randomly select four objects and divide the images of each object into three views. Each view contains images taken in consecutive 60 degrees, e.g., $[0°, 55°]$. Therefore, each view contains 48 images taken from 4 objects, i.e., 12 images per object. A basic assumption is that the 48 images of different views are defined on the same node set. The image itself is taken as graph signals, i.e., $\mathbf{X}_i \in \mathbb{R}^{48\times 1024}$. Data distributions of different views are heterogeneous because they are from different shooting angles. Furthermore, we assume that photos from three views are taken by three photographers, and they are reluctant to share their photos, which is the concern of this study. Our goal is to learn a graph representing the relationships among these 48 images for each view, i.e., $I = 3$, under data silos. The three graphs should share some common structures because they come from the same objects. Moreover, there should be four communities in the relation graphs since all images belong to four objects. Therefore, to evaluate the learned graphs, we employ the Louvain algorithm [46] to detect communities in the learned graphs. Three metrics are leveraged to evaluate the detection results, i.e., normalized mutual information (NMI), Fowlkes and Mallows index (FMI) [47], and Rand Index (RI). The labels of images are taken as the ground truth, and the results of local graphs are the average of three views. As displayed in Table IV, the Louvain algorithm successfully finds all communities in the consensus graph learned by our models. Instead, some mistakes are made in the graphs learned by IGL and FedAvg. This is expected since the consensus graph captures the common structure of local graphs, which may remove noisy edges of different views. Furthermore, the local graphs learned by our model also outperform those of other models since our proposed model is tailored for graph learning problems. As shown in Fig.7, the communities can be clearly observed in the consensus graph. However, more confusing edges appear in the graphs of IGL and FedAvg.

2) Medical data: We finally employ blood-oxygenation-level-dependent (BOLD) time series extracted from fMRI data to learn brain functional connectivity graphs. The basic assumption is that autism may affect brain functional connectivity. If we can learn about the differences in brain functional connectivity graphs between autistic and non-autistic people,

[4]https://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php



(a) IGL- $\mathcal{C}_2$ (b) FedAvg (c) PPGL-L- $\mathcal{C}_2$ (d) PPGL-C

Fig. 7: The figure depicts the communities in the learned graphs. The suffix $-\mathcal{C}_2$ stands for the local graphs of $\mathcal{C}_2$.

TABLE IV: The community detection results of graphs learned by different methods.

	IGL	FedAvg	FedProx	MRMTL	Ditto	PPGL-L	PPGL-C
NMI ↑	0.779	0.745	0.767	0.795	0.786	0.865	**1**
RI ↑	0.855	0.843	0.851	0.891	0.883	0.919	**1**
FMI ↑	0.755	0.736	0.745	0.773	0.768	0.852	**1**

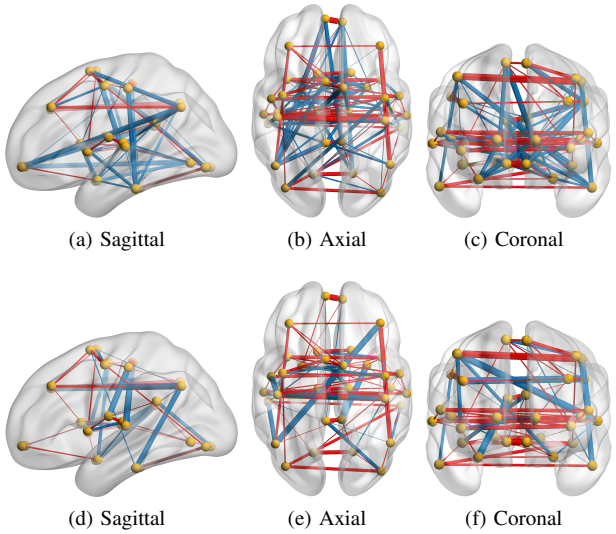
TABLE V: Comparison between the graphs learned from the autism and non-autism groups.

	#Edges	Average Edge weights
Autism	114	0.091
Non-autism	125	0.109

it may help us better understand the mechanism of autism. However, medical data are privacy-sensitive, and their transmission to an unreliable central server is usually prohibited. In this experiment, the used dataset[5] contains 539 autistic individuals and 573 typical controls, from which we select fifty autistic and fifty non-autistic subjects. Each client (subject) contains 176 signals (the length of the BOLD time series is 176). Following [12], we select 34 functional regions of interest from 90 standard regions of the Anatomical Automatic Labeling (AAL) template, indicating that $\mathbf{X}_i \in \mathbb{R}^{34\times 176}, i = 1, \ldots, 100$. Then, we use our method to learn the connectivity graphs of the 34 regions, and the edges with weights less than 0.01 are removed. We should mention that the purpose of this experiment is to show that our method can learn graphs that reflect the impact of autism on brain functional connectivity. Next, we will show that the selected 34 regions are sufficient to achieve this goal.

Table V displays the average number of edges and their average weights of the graphs learned from the autism and non-autism subjects. It is observed the average number of the graphs from autism subjects is smaller than those of non-autism subjects. Furthermore, the graphs learned from autism subjects exhibit smaller edge weights than those of non-autism subjects, indicating weaker connectivity between different functional regions. This is consistent with the study [48], which shows that autism may cause underconnectivity of brain functional regions. Furthermore, Fig.8 depicts the learned graphs using our model. It is observed that graphs

[5]http://preprocessed-connectomes-project.org/abide/



(a) Sagittal (b) Axial (c) Coronal

(d) Sagittal (e) Axial (f) Coronal

Fig. 8: The figure depicts our learned graphs from three views. The top and bottom rows show the graphs of non-autistic and autistic subjects, respectively. The red edges are those in the consensus graph, while the blue edges are those unique edges in local graphs.

of autism and non-autism individuals share many common edges learned in the consensus graph. On the other hand, there are some noticeable topological changes between the graphs of autism and non-autism individuals. These edges reflect changes in the functional connectivity caused by autism, which may aid in diagnosing autism. In the future, a domain expert may help interpret the results from a medical perspective.

VI. Conclusion

In this paper, we proposed a framework to learn graphs under data silos. In our framework, we jointly learned a personalized graph for each client to handle data heterogeneity, and a consensus graph for all clients to capture global information. Then, we devised a tailored algorithm in which all private data are processed in local clients to preserve data privacy. Convergence analyses for the proposed algorithm were provided. Extensive experiments showed that our approach can efficiently learn graphs in the target scenario. Future research may include generalizing our framework to other graph learning models except for the smoothness assumption.

APPENDIX A
PROOF OF THEOREM 1

We first provide the following lemmas necessary to prove the Theorem 1.

Lemma 1. *The function $h(\mathbf{w})$ is 4β-strongly convex.*

Proof. For any $\mathbf{w} \geq 0$, the Hessian matrix of $h(\mathbf{w})$ w.r.t. $\mathbf{w}$ is $\mathbf{H} = 4\beta\mathbf{I} + \alpha\mathbf{S}^\top \mathrm{diag}\left((\mathbf{S}\mathbf{w})^{\cdot(-2)}\right)\mathbf{S}$. It is not difficult to check that $\mathbf{H} \succ 4\beta\mathbf{I}$. Thus, $h(\mathbf{w})$ is 4β-strongly convex. □

Lemma 2. *For any $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_{I+1}] \in \mathbb{R}^{p\times(I+1)}$, we have $R(\mathbf{Y}) \leq \left(\nu\sqrt{I\omega_{\max}(\mathbf{L}_m)} + \sqrt{p}\right)\|\mathbf{Y}\|_{\mathrm{F}}$.*

Proof. Let $\widetilde{\mathbf{y}}_j \in \mathbb{R}^{I+1}$ be the j-th row vector of the matrix $\mathbf{Y}$, it is not difficult to obtain

$$\begin{aligned}
R(\mathbf{Y}) =& \nu\sum_{i=1}^{I}\|\mathbf{y}_i - \mathbf{y}_{I+1}\|_2 + \|\mathbf{y}_{I+1}\|_1 \\
\leq& \nu\sqrt{I}\sqrt{\sum_{i=1}^{I}\|\mathbf{y}_i - \mathbf{y}_{I+1}\|_2^2} + \sqrt{p}\|\mathbf{y}_{I+1}\|_2 \\
=& \nu\sqrt{I}\sqrt{\sum_{j=1}^{p}\widetilde{\mathbf{y}}_j\mathbf{L}_m\widetilde{\mathbf{y}}_j^\top} + \sqrt{p}\|\mathbf{y}_{I+1}\|_2 \\
\leq& \nu\sqrt{I}\sqrt{\sum_{j=1}^{p}\omega_{\max}(\mathbf{L}_m)\|\widetilde{\mathbf{y}}_j\|_2^2} + \sqrt{p}\|\mathbf{Y}\|_{\mathrm{F}} \\
=& \left(\nu\sqrt{I\omega_{\max}(\mathbf{L}_m)} + \sqrt{p}\right)\|\mathbf{Y}\|_{\mathrm{F}},
\end{aligned} \tag{29}$$

where the first inequality holds due to the basic inequality $\frac{\sum_{i=1}^n a_i}{n} \leq \sqrt{\frac{\sum_{i=1}^n a_i^2}{n}}$ for $a_i \geq 0$ and norm inequality $\|\mathbf{a}\|_1 \leq \sqrt{p}\|\mathbf{a}\|_2$ for $\mathbf{a} \in \mathbb{R}^p$. The second inequality holds due to the definition of the ℓ_2 matrix norm and $\|\mathbf{y}_{I+1}\|_2 \leq \|\mathbf{Y}\|_{\mathrm{F}}$. Finally, we complete the proof. □

With the two lemmas, we start the proof of Theorem 1. First, the objective function (8) can be rewritten in a matrix form. Let $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_I, \mathbf{0}] \in \mathbb{R}^{p\times(I+1)}$, and we have

$$\min_{\mathbf{W}\geq 0}\ (1/N)\left\langle \mathbf{Z}, \mathbf{W}\right\rangle + H(\mathbf{W}) + \lambda R(\mathbf{W}), \tag{30}$$

where $H(\mathbf{W}) = \sum_{i=1}^I h(\mathbf{w}_i)$. Since $\widehat{\mathbf{W}}$ is the solution of (30), we have

$$\begin{aligned}
&(1/N)\left\langle \mathbf{Z}, \widehat{\mathbf{W}}\right\rangle + H(\widehat{\mathbf{W}}) + \lambda R(\widehat{\mathbf{W}}) \\
\leq& (1/N)\left\langle \mathbf{Z}, \mathbf{W}^*\right\rangle + H(\mathbf{W}^*) + \lambda R(\mathbf{W}^*).
\end{aligned} \tag{31}$$

Accordingly, we yield that

$$\begin{aligned}
&(1/N)\left\langle \mathbf{Z}^*, \widehat{\mathbf{W}} - \mathbf{W}^*\right\rangle + H(\widehat{\mathbf{W}}) - H(\mathbf{W}^*) \\
\leq& \lambda R(\mathbf{W}^*) - \lambda R(\widehat{\mathbf{W}}) + (1/N)\left\langle \mathbf{E}, \mathbf{W}^* - \widehat{\mathbf{W}}\right\rangle,
\end{aligned} \tag{32}$$

where $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_I, \mathbf{0}] \in \mathbb{R}^{p\times(I+1)}$ is the error matrix, and $\mathbf{Z}^* = [\mathbf{z}_1^*, \ldots, \mathbf{z}_I^*, \mathbf{0}] \in \mathbb{R}^{p\times(I+1)}$ is the real pairwise distance matrix.

Then, we define a variable $v = \sum_{i=1}^I\sum_{j=1}^p \frac{1}{\sigma_e^2}(\mathbf{e}_i[j])^2 = \frac{1}{\sigma_e^2}\|\mathbf{E}\|_{\mathrm{F}}^2$ which follows a chi-squared distribution with the degree of freedom as pI. Using the Wallace inequality [49], for $\delta > 0$, we have

$$\Pr(v \geq pI + \delta) \leq \exp\left(-\frac{1}{2}\left(\delta - pI\log\left(1 + \frac{\delta}{pI}\right)\right)\right), \tag{33}$$

which will lead to

$$\begin{aligned}
&\Pr\left((1/N)\|\mathbf{E}\|_{\mathrm{F}} \leq (\sigma_e/N)\sqrt{pI+\delta}\right) \\
\geq& 1 - \exp\left(-\frac{1}{2}\left(\delta - pI\log\left(1 + \frac{\delta}{pI}\right)\right)\right).
\end{aligned} \tag{34}$$

According to (34), with probability at least $1 - \exp\left(-\frac{1}{2}\left(\delta - pI\log\left(1+\frac{\delta}{pI}\right)\right)\right)$, we obtain

$$\begin{aligned}
&(1/N)\left\langle \mathbf{E}, \mathbf{W}^* - \widehat{\mathbf{W}}\right\rangle \leq (1/N)\|\mathbf{E}\|_{\mathrm{F}}\|\mathbf{W}^* - \widehat{\mathbf{W}}\|_{\mathrm{F}} \\
&\leq (\sigma_e/N)\sqrt{pI+\delta}\|\mathbf{W}^* - \widehat{\mathbf{W}}\|_{\mathrm{F}}.
\end{aligned} \tag{35}$$

Next, we focus on H related terms in (32) and yield

$$\begin{aligned}
&H(\widehat{\mathbf{W}}) - H(\mathbf{W}^*) = \sum_{i=1}^I h(\widehat{\mathbf{w}}_i) - h(\mathbf{w}_i^*) \\
\geq& \sum_{i=1}^I \left\langle \nabla h(\mathbf{w}_i^*), \widehat{\mathbf{w}}_i - \mathbf{w}_i^*\right\rangle + 2\beta\left\|\widehat{\mathbf{w}}_i - \mathbf{w}_i^*\right\|_2^2 \\
\geq& \sum_{i=1}^I -\left\|\nabla h(\mathbf{w}_i^*)\right\|_2\left\|\widehat{\mathbf{w}}_i - \mathbf{w}_i^*\right\|_2 + 2\beta\left\|\widehat{\mathbf{w}}_i - \mathbf{w}_i^*\right\|_2^2 \\
\geq& -C_h\sqrt{I}\left\|\widehat{\mathbf{W}} - \mathbf{W}^*\right\|_{\mathrm{F}} + 2\beta\left\|\widehat{\mathbf{W}} - \mathbf{W}^*\right\|_{\mathrm{F}}^2.
\end{aligned} \tag{36}$$

The first inequality holds due to that h is a 4β- strongly convex function, as stated in Lemma 1. The second inequality holds due to Cauchy-Schwarz inequality. The last inequality holds due to Assumption 2 and the fact that $\sum_{i=1}^I\|\widehat{\mathbf{w}}_i - \mathbf{w}_i^*\|_2 \leq \sqrt{I}\|\widehat{\mathbf{W}} - \mathbf{W}^*\|_{\mathrm{F}}$.

For $\mathbf{Z}^*$ related terms, we have

$$\begin{aligned}
&(1/N)\left\langle \mathbf{Z}^*, \widehat{\mathbf{W}} - \mathbf{W}^*\right\rangle \geq -(1/N)\left\|\mathbf{Z}^*\right\|_{\mathrm{F}}\left\|\widehat{\mathbf{W}} - \mathbf{W}^*\right\|_{\mathrm{F}} \\
&\geq -\left(\sqrt{I}C_z/N\right)\left\|\widehat{\mathbf{W}} - \mathbf{W}^*\right\|_{\mathrm{F}}.
\end{aligned} \tag{37}$$

The second inequality holds due to Assumption 1.

Finally, we focus on the R related terms and have

$$\begin{aligned}
&R(\mathbf{W}^*) - R(\widehat{\mathbf{W}}) \\
=& \nu\sum_{i=1}^I(\|\mathbf{w}_i^* - \mathbf{w}_{\mathrm{con}}^*\|_2 - \|\widehat{\mathbf{w}}_i - \widehat{\mathbf{w}}_{\mathrm{con}}\|_2) \\
&+ \|\mathbf{w}_{\mathrm{con}}^*\|_1 - \|\widehat{\mathbf{w}}_{\mathrm{con}}\|_1 \\
\leq& \nu\sum_{i=1}^I\|(\mathbf{w}_i^* - \widehat{\mathbf{w}}_i) - (\mathbf{w}_{\mathrm{con}}^* - \widehat{\mathbf{w}}_{\mathrm{con}})\|_2 + \|\mathbf{w}_{\mathrm{con}}^* - \widehat{\mathbf{w}}_{\mathrm{con}}\|_1 \\
=& R(\mathbf{W}^* - \widehat{\mathbf{W}}) \leq \left(\nu\sqrt{I\omega_{\max}(\mathbf{L}_m)} + \sqrt{p}\right)\|\mathbf{W}^* - \widehat{\mathbf{W}}\|_{\mathrm{F}},
\end{aligned} \tag{38}$$

where the last inequality holds due to Lemma 2.

Plugging (35), (36), (37), and (38) into (32), it is not difficult to yield

$$2\beta\left\|\widehat{\mathbf{W}} - \mathbf{W}^*\right\|_{\mathrm{F}}^2 \leq (\sigma_e/N)\sqrt{pI+\delta}\left\|\mathbf{W}^* - \widehat{\mathbf{W}}\right\|_{\mathrm{F}}$$

$$+ C_h\sqrt{I}\left\|\widehat{\mathbf{W}}-\mathbf{W}^*\right\|_{\mathrm{F}} + \left(\sqrt{I}C_z/N\right)\left\|\widehat{\mathbf{W}}-\mathbf{W}^*\right\|_{\mathrm{F}} + \lambda\left(\nu\sqrt{I\omega_{\max}(\mathbf{L}_m)}+\sqrt{p}\right)\left\|\widehat{\mathbf{W}}-\mathbf{W}^*\right\|_{\mathrm{F}}. \quad (39)$$

Finally, combining (39) and (15), we can obtain the conclusion of Theorem 1.

Appendix B
Proof of Proposition 1

The proof is mainly from [50] but with some modifications. For two vectors $\mathbf{y}$ and $\mathbf{y}'$ in $\widetilde{\mathcal{W}}$, we have

$$\begin{aligned}
&\left\|\nabla_{\mathbf{y}} f_i^{(t)}(\mathbf{y},\mathbf{w}_{\mathrm{con}}^{(t)}) - \nabla_{\mathbf{y}'} f_i^{(t)}(\mathbf{y}',\mathbf{w}_{\mathrm{con}}^{(t)})\right\|_2 \\
=&\left\|\left(4\beta+\rho\gamma_i^{(t)}\right)(\mathbf{y}-\mathbf{y}') - \alpha\mathbf{S}^\top\left(\frac{1}{\mathbf{S}\mathbf{y}+\zeta\mathbf{1}}-\frac{1}{\mathbf{S}\mathbf{y}'+\zeta\mathbf{1}}\right)\right\|_2 \\
\le&\left(4\beta+\rho\gamma_i^{(t)}\right)\|\mathbf{y}-\mathbf{y}'\|_2 + \alpha\left\|\mathbf{S}^\top\right\|_2\left\|\frac{1}{\mathbf{S}\mathbf{y}+\zeta\mathbf{1}}-\frac{1}{\mathbf{S}\mathbf{y}'+\zeta\mathbf{1}}\right\|_2 \\
\le&\left(4\beta+\rho\gamma_i^{(t)}\right)\|\mathbf{y}-\mathbf{y}'\|_2 + \frac{\alpha\|\mathbf{S}\|_2}{\zeta^2}\|\mathbf{S}\mathbf{y}-\mathbf{S}\mathbf{y}'\|_2 \\
\le&\left(4\beta+\rho\gamma_i^{(t)}\right)\|\mathbf{y}-\mathbf{y}'\|_2 + \frac{\alpha\|\mathbf{S}\|_2^2}{\zeta^2}\|\mathbf{y}-\mathbf{y}'\|_2 \\
=&\left(4\beta+\frac{2\alpha(d-1)}{\zeta^2}+\rho\gamma_i^{(t)}\right)\|\mathbf{y}-\mathbf{y}'\|_2 \\
:=&L_i^{(t)}\|\mathbf{y}-\mathbf{y}'\|_2. \quad (40)
\end{aligned}$$

The first inequality holds due to the triangle inequality, while the second equality holds due to Lemma 1 in [50]. From (40), we can conclude that $f_i^{(t)}(\mathbf{w}_i,\mathbf{w}_{\mathrm{con}}^{(t)})$ is $L_i^{(t)}$-Lipschitz smooth w.r.t. $\mathbf{w}_i$ on $\widetilde{\mathcal{W}}$. On the other hand, the strong convexity of $f_i^{(t)}(\mathbf{w}_i,\mathbf{w}_{\mathrm{con}}^{(t)})$ can be proved in the same way as Lemma 1.

Appendix C
Proof of Theorem 2

First, let us provide a lemma that is essential to the proof.

Lemma 3. *([51]) For any two positive constants $a, b > 0$, we have*

$$\sqrt{a}-\frac{a}{2\sqrt{b}} \le \sqrt{b}-\frac{b}{2\sqrt{b}}. \quad (41)$$

Proof. For $a, b > 0$, it is not difficult to derive that

$$\begin{aligned}
&\left(\sqrt{a}-\sqrt{b}\right)^2 \ge 0 \\
\Longrightarrow\ & a-2\sqrt{ab}+b \ge 0 \\
\Longrightarrow\ & \sqrt{a}-\frac{a}{2\sqrt{b}} \le \frac{\sqrt{b}}{2} \\
\Longrightarrow\ & \sqrt{a}-\frac{a}{2\sqrt{b}} \le \sqrt{b}-\frac{b}{2\sqrt{b}}. \quad (42)
\end{aligned}$$

The proof of Lemma 3 completes. □

Then, we start to prove Theorem 2. Let us consider the t-th communication round. According to Proposition 1, the objective function $f_i^{(t)}(\mathbf{w}_i,\mathbf{w}_{\mathrm{con}}^{(t)})$ is $L_i^{(t)}$-Lipschitz smooth and $S_i^{(t)}$-strongly convex w.r.t. $\mathbf{w}_i$. Based on Assumption 3 and the stepsize η_w defined in Assumption 4, the accelerated projected gradient descent algorithm (19)-(21) is proved to converge to the optimal minimum [39]. Thus, after enough $K_i^{(t)}$ iterations, we can obtain $\mathbf{w}_i^{(t+1)}$ such that

$$f_i^{(t)}(\mathbf{w}_i^{(t+1)},\mathbf{w}_{\mathrm{con}}^{(t)}) \le f_i^{(t)}(\mathbf{w}_i^{(t)},\mathbf{w}_{\mathrm{con}}^{(t)}). \quad (43)$$

On the other hand, when updating $\mathbf{w}_{\mathrm{con}}$ in the t-th communication round, the $\mathbf{w}_{\mathrm{con}}^{(t+1)}$ output by proximal operator (24) is the solution of problem (22), i.e.,

$$\mathbf{w}_{\mathrm{con}}^{(t+1)} = \operatorname*{argmin}_{\mathbf{w}_{\mathrm{con}}} \sum_{i=1}^{I} f_i^{(t)}(\mathbf{w}_i^{(t+1)},\mathbf{w}_{\mathrm{con}}) + \lambda\|\mathbf{w}_{\mathrm{con}}\|_1.$$

Therefore, we have

$$\begin{aligned}
&\sum_{i=1}^{I} f_i^{(t)}(\mathbf{w}_i^{(t+1)},\mathbf{w}_{\mathrm{con}}^{(t+1)}) + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_1 \\
\le&\sum_{i=1}^{I} f_i^{(t)}(\mathbf{w}_i^{(t+1)},\mathbf{w}_{\mathrm{con}}^{(t)}) + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t)}\|_1. \quad (44)
\end{aligned}$$

Combining (43) and (44), we can obtain

$$\begin{aligned}
&\sum_{i=1}^{I} f_i^{(t)}(\mathbf{w}_i^{(t+1)},\mathbf{w}_{\mathrm{con}}^{(t+1)}) + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_1 \\
\le&\sum_{i=1}^{I} f_i^{(t)}(\mathbf{w}_i^{(t)},\mathbf{w}_{\mathrm{con}}^{(t)}) + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t)}\|_1. \quad (45)
\end{aligned}$$

Bring the updated $\gamma_i^{(t)}$ in (25) to (45), we obtain

$$\begin{aligned}
&\sum_{i=1}^{I} g_i(\mathbf{w}_i^{(t+1)}) + \rho\frac{\|\mathbf{w}_i^{(t+1)}-\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_2^2}{2\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2} + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_1 \\
\le&\sum_{i=1}^{I} g_i(\mathbf{w}_i^{(t)}) + \rho\frac{\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2^2}{2\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2} + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t)}\|_1. \quad (46)
\end{aligned}$$

According to Lemma 3, we have

$$\begin{aligned}
&\sum_{i=1}^{I}\|\mathbf{w}_i^{(t+1)}-\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_2 - \sum_{i=1}^{I}\frac{\|\mathbf{w}_i^{(t+1)}-\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_2^2}{2\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2} \\
\le&\sum_{i=1}^{I}\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2 - \sum_{i=1}^{I}\frac{\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2^2}{2\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2}. \quad (47)
\end{aligned}$$

Summing (46) and (47), we have

$$\begin{aligned}
&\sum_{i=1}^{I} g_i(\mathbf{w}_i^{(t+1)}) + \rho\|\mathbf{w}_i^{(t+1)}-\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_2 + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t+1)}\|_1 \\
\le&\sum_{i=1}^{I} g_i(\mathbf{w}_i^{(t)}) + \rho\|\mathbf{w}_i^{(t)}-\mathbf{w}_{\mathrm{con}}^{(t)}\|_2 + \lambda\|\mathbf{w}_{\mathrm{con}}^{(t)}\|_1. \quad (48)
\end{aligned}$$

Thus, the update of the t-th communication round will monotonically decrease the objective function of (8). As t increases, the optimization will converge. When the convergence reaches, the KKT condition of problem (8) will hold, i.e., Algorithm 1 at least converges to a local optimal minimum.

REFERENCES

[1] 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, 2019.

[2] U. Von Luxburg, "A tutorial on spectral clustering," *Stat. Comput.*, vol. 17, pp. 395–416, 2007.

[3] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, 2020.

[4] 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, 2019.

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

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

[7] J. Ying, J. V. de Miranda Cardoso, and D. Palomar, "Nonconvex sparse graph learning under laplacian constrained graphical model," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 7101–7113, 2020.

[8] J.-F. Cai, J. V. de Miranda Cardoso, D. Palomar, and J. Ying, "Fast projected newton-like method for precision matrix estimation under total positivity," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024.

[9] 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, 2018.

[10] 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, 2016.

[11] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Int. Conf. Artif. Intell. Stat., AISTATS*. PMLR, 2016, pp. 920–929.

[12] X. Pu, T. Cao, X. Zhang, X. Dong, and S. Chen, "Learning to learn graph topologies," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 4249–4262, 2021.

[13] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, 2021.

[14] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ Digit. Med*, vol. 3, no. 1, pp. 1–7, 2020.

[15] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, 2020.

[16] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A survey of distributed optimization," *Annu. Rev. Control*, vol. 47, pp. 278–305, 2019.

[17] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Stat., AISTATS*. PMLR, 2017, pp. 1273–1282.

[18] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, 2022.

[19] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *arXiv preprint arXiv:2002.10619*, 2020.

[20] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[21] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2017, pp. 1126–1135.

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

[23] P. J. Bickel and E. Levina, "Regularized estimation of large covariance matrices," *Ann. Statist*, vol. 36, no. 1, p. 199–227, 2008.

[24] Y. Yuan, D. W. Soh, X. Yang, K. Guo, and T. Q. Quek, "Joint network topology inference via structured fusion regularization," *arXiv preprint arXiv:2103.03471*, 2021.

[25] X. Zhang and Q. Wang, "Time-varying graph learning under structured temporal priors," in *Proc. Eur. Signal Process. Conf.* IEEE, 2022, pp. 2141–2145.

[26] K. Yamada, Y. Tanaka, and A. Ortega, "Time-varying graph learning based on sparseness of temporal variation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.* IEEE, 2019, pp. 5411–5415.

[27] X. Zhang and Q. Wang, "A graph-assisted framework for multiple graph learning," *IEEE Trans. Signal. Inf. Process. Netw.*, pp. 1–16, 2024.

[28] C. T Dinh, N. Tran, and J. Nguyen, "Personalized federated learning with moreau envelopes," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 21 394–21 405, 2020.

[29] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, "Personalized and private peer-to-peer machine learning," in *Proc. Int. Conf. Artif. Intell. Stat., AISTATS*. PMLR, 2018, pp. 473–481.

[30] O. Marfoq, G. Neglia, A. Bellet, L. Kameni, and R. Vidal, "Federated multi-task learning under a mixture of distributions," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 15 434–15 447, 2021.

[31] F. Chen, G. Long, Z. Wu, T. Zhou, and J. Jiang, "Personalized federated learning with graph," *arXiv preprint arXiv:2203.00829*, 2022.

[32] L. Stanković, M. Daković, and E. Sejdić, "Introduction to graph signal processing," in *Vertex-Frequency Analysis of Graph Signals*. Springer, 2019, pp. 3–108.

[33] Z. Hu, F. Nie, W. Chang, S. Hao, R. Wang, and X. Li, "Multi-view spectral clustering via sparse graph learning," *Neurocomputing*, vol. 384, pp. 1–10, 2020.

[34] F. Nie, J. Li, X. Li *et al.*, "Self-weighted multiview clustering with multiple graphs." in *Int. Joint Conf. Artif. Intell.*, 2017, pp. 2564–2570.

[35] S. Hara and T. Washio, "Learning a common substructure of multiple graphical gaussian models," *Neur. Netw.*, vol. 38, pp. 23–38, 2013.

[36] 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.

[37] A. Karaaslanli, S. Saha, S. Aviyente, and T. Maiti, "Multiview graph learning for single-cell rna sequencing data," *bioRxiv*, 2021.

[38] A. Karaaslanli and S. Aviyente, "Multiview graph learning with consensus graph," *arXiv preprint arXiv:2401.13769*, 2024.

[39] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.

[40] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*. IEEE, 2017, pp. 3–18.

[41] M. Al-Rubaie and J. M. Chang, "Reconstruction attacks against mobile-based continuous authentication systems in the cloud," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 12, pp. 2648–2663, 2016.

[42] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory of Cryptography Conf.* Springer, 2006, pp. 265–284.

[43] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Mach. Learn. and Sys.*, vol. 2, pp. 429–450, 2020.

[44] K. Liu, S. Hu, S. Z. Wu, and V. Smith, "On privacy and personalization in cross-silo federated learning," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 5925–5940, 2022.

[45] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2021, pp. 6357–6368.

[46] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, no. 3-5, pp. 75–174, 2010.

[47] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[48] R. K. Kana, L. E. Libero, and M. S. Moore, "Disrupted cortical connectivity theory as an explanatory model for autism spectrum disorders," *Phys. Life Rev*, vol. 8, no. 4, pp. 410–437, 2011.

[49] D. L. Wallace, "Bounds on normal approximations to student's and the chi-square distributions," *Ann. Inst. Stat. Math.*, pp. 1121–1130, 1959.

[50] S. S. Saboksayr, G. Mateos, and M. Cetin, "Online discriminative graph learning from multi-class smooth signals," *Signal Process.*, vol. 186, p. 108101, 2021.

[51] F. Nie, H. Huang, X. Cai, and C. Ding, "Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 23, 2010.