Clustered Federated Learning based on Nonconvex Pairwise Fusion

Xue Yu, Ziyi Liu, Wu Wang and Yifan Sun

Center for Applied Statistics, School of Statistics, Renmin University of China, Beijing, China
{xueyu_2019, ziyiliu, wu.wang, sunyifan}@ruc.edu.cn.

December 27, 2023

Abstract

This study investigates clustered federated learning (FL), one of the formulations of FL with non-i.i.d. data, where the devices are partitioned into clusters and each cluster optimally fits its data with a localized model. We propose a clustered FL framework that incorporates a nonconvex penalty to pairwise differences of parameters. Without a priori knowledge of the set of devices in each cluster and the number of clusters, this framework can autonomously estimate cluster structures. To implement the proposed framework, we introduce a novel clustered FL method called Fusion Penalized Federated Clustering (FPFC). Building upon the standard alternating direction method of multipliers (ADMM), FPFC can perform partial updates at each communication round and allows parallel computation with variable workload. These strategies significantly reduce the communication cost while ensuring privacy, making it practical for FL. We also propose a new warmup strategy for hyperparameter tuning in FL settings and explore the asynchronous variant of FPFC (asyncFPFC). Theoretical analysis provides convergence guarantees for FPFC with general losses and establishes the statistical convergence rate under a linear model with squared loss. Extensive experiments have demonstrated the superiority of FPFC compared to current methods, including robustness and generalization capability.

1 Introduction

Increasing amounts of data are generated by end users’ own devices (also called clients), such as mobile phones, wearable devices, and autonomous vehicles. The fast-growing storage and computational capacities of these terminal devices, coupled with concerns over privacy, have led to a growing interest in federated learning (FL) [25,33]. Standard FL approaches collaboratively train a shared global model for all devices while maintaining data on each device [26,27,34,43,45]. As such, critical concerns such as privacy, security, and data access rights are well-addressed in the FL framework. However, as discussed in [50], the local data stored on each device usually have heterogeneous conditional distributions due to diverse user characteristics. The data and statistical heterogeneity observed across different devices obfuscate the efforts to build a global model in many FL applications.

In addressing the limitations of a global model, a recent proposal, clustered FL [17,41,50,61], assumes that devices can be grouped into clusters, where devices within the same cluster share identical models. Cluster structures are very common in many applications, such as recommendation systems [32] and precision medicine [29]. Thus, clustered FL has drawn increasing attention and is considered a formulation of FL that can address the data heterogeneity problem. However, the key issue regarding clustered FL is that the set of devices in each cluster is unknown a priori. To
tackle this challenge, several methods have been proposed. Sattler et al. \cite{50} proposed a clustered FL (CFL) algorithm under the clustered FL framework in the spirit of the hierarchical divisive clustering algorithm. However, the clustering procedure is implemented at the central server, resulting in high computation costs, especially when the number of devices is large. Inspired by the classical K-means algorithm, Mansour et al. \cite{41} introduced the HYPCLUSTER algorithm and Ghosh et al. \cite{17} presented the Iterative Federated Clustering Algorithm (IFCA). More recently, Marfoq et al. \cite{42} developed the Federated Expectation-Maximization (FedEM) algorithm, which offers a versatile approach for clustered FL and personalized FL. Although the three algorithms eliminate the need for centralized clustering procedures, they require specifying the number of clusters, and the clustering results are usually sensitive to the initial values.

Motivated by the limitations mentioned above in current studies, we propose a fusion penalization-based framework for clustered FL and develop a novel clustered FL method called fusion penalized federated clustering (FPFC). In this work, we introduce a nonconvex function, such as the smoothly clipped absolute deviation (SCAD) penalty \cite{14}, to penalize the pairwise differences of local model parameters. The penalty function can force pairs of local parameters to be exactly equal if they are close to each other, effectively creating clusters within the local parameters. This method places the clustered FL on a solid theoretical footing based on a well-defined objective function. In addition to conventional convergence analysis of algorithms, the statistical properties, e.g., the consistency of identifying the cluster structure of devices, of this method can be rigorously established. The main contributions of our work can be summarized as follows:

- We formulate clustered FL as a penalized optimization problem, where the nonconvex fusion penalization term shrinks the pairwise differences of model parameters of devices and encourages minimal differences, thus clustering the local devices.

- Under the proposed framework, we develop a novel method called fusion penalized federated clustering (FPFC) to address general clustered FL problems. This method can automatically determine the cluster membership of devices and derive optimal models specific to each cluster in a distributed setting, without requiring prior knowledge regarding the number of clusters and the composition of devices within each cluster.

- The proposed method is a general framework that can handle learning tasks with nonconvex losses, such as neural networks. Regarding the algorithm, we extend the standard Alternating Direction Method of Multipliers (ADMM) algorithm to the FL setup. Specifically, we decouple the joint optimization problem into a series of subproblems that can be solved by local devices in parallel, in the meantime, the communication cost is greatly reduced compared to similar methods. Moreover, it allows for inexact minimization of each subproblem, further enhancing computational efficiency.

- We introduce a novel warmup strategy for hyperparameter tuning in the FL setting. This approach significantly reduces the communication cost compared to the conventional cross-validation method, while simultaneously achieving better performance.

- Theoretically, we establish the convergence rate of FPFC for general nonconvex losses under standard assumptions. More importantly, unlike IFCA \cite{17}, which assumes full device participation to achieve convergence, our analysis allows partial participation by selecting a subset of devices to perform local updates at each communication round. Furthermore, the statistical convergence rate of the proposed method has been established under a linear model with squared loss.
Finally, we conduct extensive experiments to compare our method with state-of-the-art approaches. We explore the performance of FPFC in aspects of robustness, generalization capability, and communication efficiency. We also introduce several variants of FPFC, including asynchronous updates. The results consistently demonstrate the versatility and effectiveness of FPFC across a diverse range of settings.

This paper is structured as follows. Section 2 outlines related works. Section 3 provides the relevant background and formulates the main problem. Section 4 details the proposed FPFC algorithm and the regularization parameter tuning strategy. Section 5 presents the theoretical results of FPFC and its statistical properties. The experimental results are presented in Section 6. Section 7 concludes the paper.

2 Related work

As this work addresses several issues of existing clustered FL algorithms via a pairwise fusion penalty, we review the related work in two subtopics: heterogeneous FL and clustering via pairwise fusion penalty.

2.1 Heterogeneous federated learning

Federated learning has sparked significant interest since it was first proposed by [43]. The most popular method is the FederatedAveraging (FedAvg) algorithm, and it has been shown to work well empirically, particularly for non-convex problems. FedAvg converges when the data is i.i.d [54], but it can diverge in practical settings when the data on the users' devices are non-i.i.d, i.e., heterogeneous [35]. Several formulations have been proposed to tackle these heterogeneity issues, such as adding some adaptation terms to the global model, personalized FL, and clustered FL.

Karimireddy et al. [26] pointed out that FedAvg may suffer from device drift, so they used control variables to reduce the drift between different devices and achieve favorable performance. Li et al. [34] applied a proximal term to limit the impact of statistical heterogeneity under an assumption of bounded local dissimilarity. When data distributions across devices exhibit significant non-i.i.d. characteristics, it becomes challenging to train a global model that performs well across all devices. Another choice is to build personalized models for each device instead of learning a single shared model. As one of the formulations, personalized FL aims to jointly train personalized models that can work better than the global model and the local models without collaboration. The most common method for personalization involves two steps [28]. First, a global model is trained under the FL framework, and then each device can train a customized model with its private data based on the global model [16,59]. [13] proposed Personalized FedAvg (Per-FedAvg) to train a shared model that performs well after each device updates it by running a few steps of gradient descent with respect to its own data. Another efficient technique is to regularize personalized models using the relationships between related devices [20,24,55,62]. Smith et al. [52] suggested a multi-task learning (MTL) framework to handle the statistical challenges of FL and proposed MOCHA to address the systems challenges of federated MTL based on the relationships among tasks. Lin et al. [37] proposed to project local models into a shared-and-fixed low-dimensional random subspace and use infimal convolution to achieve personalization.

The formulation of clustered FL has been proposed in recent years. Instead of building a unified model for all devices, clustered FL aims to construct a set of heterogeneous models, each corresponding to one cluster of devices. As one of the earliest clustered FL methods, clustered FL (CFL) [50] follows the idea of the divisive hierarchical clustering algorithm. Specifically, it
divides the devices recursively in a top-down manner, relying on the cosine similarity between the gradient updates of the devices. This method has several favorable properties. In particular, it is applicable to general nonconvex losses, does not require the number of clusters to be known a priori, and comes with theoretical guarantees on the clustering quality. However, it also has several limitations. Practically, optimally bisecting devices requires significant computational resources for the central server, especially when dealing with a large number of devices. Theoretically, its convergence behavior has not been rigorously characterized. Another line of research aims to minimize an objective function (without penalization) involving possible clusterings by using some iterative algorithms that are in the similar spirit of K-means clustering, e.g., Iterative Federated Clustering Algorithm (IFCA) [17] and HYPCLUS-TER [41]. Although their convergence behaviors have been rigorously characterized, these methods need to know the number of clusters in advance and rely on a relatively good initialization to ensure optimal results. Moreover, convergence analysis of IFCA has only been shown for strongly convex losses, although it has good performances in some nonconvex settings. Federated Expectation-Maximization (FedEM) [42] is a state-of-the-art federated multi-task learning method with clustered FL as a special case. Similar to IFCA and HYPCLUSTER, it requires specification of the number of clusters and underlying data distribution, which is infeasible in practice.

Recently, Cho et al. [9] proposed a framework named COMET which combines clustered knowledge transfer and personalized federated learning to reduce communication costs and accommodate model heterogeneity. Similar to IFCA, COMET also utilizes the K-means clustering algorithm on the server for clustering. Devices are required to have access to a public dataset in this framework. Additionally, Learning Across Domains and Devices (LADD) [51] applied K-means clustering to partition devices into clusters based on the image styles of each device in autonomous driving applications. To address the challenges of the straggler effect in synchronous FL and model staleness in asynchronous FL, Zhang et al. [64] presented a semi-asynchronous clustered federated learning framework. They utilized spectral clustering on an affinity matrix that incorporates information about devices’ similarity, gradient direction, and the latency of model updates. Since clustering is conducted at a pre-processing stage, the proposed method can be categorized as a one-shot clustering approach. Another one-shot clustering method is Principal Angles analysis for Clustered Federated Learning (PACFL) [57]. By constructing a proximity matrix based on the principal angles between the local data subspaces, PACFL performed agglomerative hierarchical clustering. The benefit of one-shot clustering is that it reduces communication costs as it only requires running clustering before training. However, both algorithms require specifying the number of groups or adjusting the clustering threshold to determine the structure of clusters.

In contrast to the aforementioned methods, our proposed method, FPFC, can automatically cluster devices without the knowledge of the number of clusters or underlying data distribution. In addition, it does not require a complex centralized clustering procedure. It updates only a subset of devices at each communication round and allows each participating device to solve the corresponding minimization problem inexactly up to a given relative accuracy.

2.2 Clustering via pairwise fusion penalty

The pairwise fusion penalty is a recently proposed technique for the clustering problem. Pelckmans et al. [47] and Hocking et al. [22] applied the pairwise fusion penalty to unsupervised clustering and proposed the convex clustering method. Recently, Ma et al. [39,40] generalized this technique to supervised clustering. Clustered FL can be deemed as a supervised clustering problem in a distributed setting, where subjects are the devices. Another related method is the Network Lasso [18]. As a generalization of the group lasso to a network setting, Network Lasso allows simultaneous
clustering and optimization on a network by penalizing the $\ell_2$-norm of differences of parameters at adjacent nodes. However, it is specifically designed for the fully decentralized scenario, where the network structure of devices is known in advance. The communication cost and requirements of exact minimization and full device participation make it less applicable to FL. (See Section 4.1 for further discussion). In the existing literature, one of the most relevant works is Federated Attentive Message Passing (FedAMP) \cite{23}, which focuses on personalized FL by encouraging collaboration between similar devices. While FedAMP also applies a pairwise fusion penalty, it differs from FPFC in that it aims to seek the optimal personalized model for each device. FedAMP cannot cluster devices because it penalizes the squared norm instead of the norm of the pairwise differences of parameters. Moreover, it requires full device participation, and each device solves the corresponding minimization problem exactly, making it impractical for FL.

In this work, we apply the pairwise fusion penalty to supervised clustering in the FL setting and develop a distributed optimization algorithm. In contrast to Network Lasso, our focus is on a standard FL setting with a hub-and-spoke topology, where the hub represents a coordinating server and the spokes connect to devices. The server only sends one parameter to each participating device at each communication round, significantly reducing communication costs. Moreover, to attain a reliable cluster structure, we resort to nonconvex penalties, which are asymptotically unbiased and are more aggressive than convex penalties, e.g. $\ell_1$ norm penalty, in enforcing a sparser solution. The convergence analysis of FPFC for general nonconvex losses has been rigorously established under standard assumptions (Assumptions \[13\]).

**Notations.** $[m]$ is the set of integers $\{1, \ldots, m\}$, and $\| \cdot \|$ represents the $\ell_2$ norm of vectors.

### 3 Clustered federated learning via nonconvex pairwise fusion

In a system consisting of $m$ devices and one server, communication between the server and devices occurs according to predefined protocols. Each device $i$ is associated with its parameter $\omega_i \in \mathbb{R}^d$. The devices are grouped into $L$ disjoint clusters, where devices within the same cluster share identical parameters. Specifically, let $G = \{G_1, \ldots, G_L\}$ represents a mutually exclusive partition of $[m]$ and $\alpha_l$ denotes the common value for $\omega_i$ within the $l$-th cluster. Thus, for all $i \in G_l$, $\omega_i = \alpha_l$. In practice, the number of $L$ is unknown, and we also have no prior knowledge of the cluster structures. To address these issues, we minimize the following objective function:

$$\min_{\omega} \{F(\omega) = \sum_{i=1}^{m} f_i(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} g(\|\omega_i - \omega_j\|, \lambda)\}, \tag{1}$$

where $\omega = (\omega_1^T, \ldots, \omega_m^T)^T$, and $g(\cdot, \lambda)$ is a penalty function with hyperparameter $\lambda > 0$. Here, $f_i(\omega_i) = \frac{1}{n_i} \sum_{s=1}^{n_i} \ell(\omega_i; z_s^i)$, where $\{z_s^i \equiv (X_s^i, y_s^i), s \in [n_i]\}$ is the dataset stored at the $i$-th device and $\ell(\omega_i; z_s^i)$ is the loss function associated with model parameter $\omega_i$. The first term $\sum_{i=1}^{m} f_i(\omega_i)$ in (1) is the sum of the empirical losses of all devices. The second term serves as a pairwise fusion penalty, encouraging the partitioning of devices into clusters. This framework is advantageous over IFCA and many others, as the number and structure of clusters are automatically and simultaneously determined in the minimization process. A popular choice for the penalty function $g(\cdot, \lambda)$ is the $\ell_1$ norm or Lasso penalty with $g(t, \lambda) = \lambda |t|$. Nevertheless, this convex function will result in biased parameter estimates. Therefore, our attention is directed towards nonconvex functions. A well-known example is the smoothly clipped absolute deviation (SCAD) penalty \[14\], denoted as $P_\alpha(t, \lambda)$, which
has the form

$$P_a(t, \lambda) = \begin{cases} 
\lambda|t|, & |t| \leq \lambda \\
\lambda^2(a+1) \left( a \left| t \right| - 0.5 \left( t^2 + \lambda^2 \right) \right), & |t| > a \lambda 
\end{cases}$$

(2)

It is worth noting that, in contrast to FedAMP [23], we use $\|\omega_i - \omega_j\|$ instead of $\|\omega_i - \omega_j\|^2$ in the penalty function. To see why the nonconvex SCAD penalty consistently recovers the underlying cluster memberships, the first three subfigures of Fig. [40] present the solution paths for the estimated values of $\omega$ against $\lambda$ using the squared $\ell_2$ norm, LASSO ($\ell_1$ norm), and SCAD penalties in a toy example. In this example, we consider a simple linear regression problem with 50 devices divided into 2 clusters. It can be observed that for the SCAD penalty, the estimated values of each device converge to two different values around $-1$ and $1$ which are the true values of the two groups when $\lambda$ reaches a certain value (around 0.4). However, for the $\ell_1$ norm penalty, the estimated values $\omega$ converge to a common value quickly as $\lambda$ increases to 0.08. For the squared $\ell_2$ norm penalty ($\| \cdot \|^2$), the estimated values of different devices gradually approach each other but do not ultimately converge to a common value. Neither squared $\ell_2$ norm nor $\ell_1$ norm penalties can identify the clusters for any $\lambda$.

**Smoothing technique for the SCAD penalty.** The non-differentiability of the SCAD penalty poses challenges in both implementation and theoretical analysis when employing gradient descent algorithms. To overcome the computational and theoretical challenges, we approximate the non-differentiable SCAD penalty $P_a(t, \lambda)$ by the following continuously differentiable function, called the smoothed SCAD penalty:

$$\tilde{P}_a(t, \lambda) = \left( \frac{\lambda}{2 \xi} t^2 + \frac{\xi \lambda}{2} \right) I_{\{|t| \leq \xi\}}(t) + P_a(t, \lambda) I_{\{|t| > \xi\}}(t),$$

(3)

where $0 < \xi < \lambda$ and $I_C(t)$ is the indicator function of the set $C$. The last subgraph of Fig. 1 shows the solution paths for the estimated values against $\lambda$ by using the smoothed SCAD penalty. Because we only approximate the SCAD function around the nondifferential point (i.e., 0), using the smoothed SCAD function does not affect the clustering results. When $\lambda$ reaches a certain value, the smoothed SCAD penalty term still identifies the clusters correctly. The properties of $\tilde{P}_a(t, \lambda)$ are summarized in the following proposition.

**Proposition 1.** The function $\tilde{P}_a(t, \lambda)$ is continuously differentiable at any $t \in \mathbb{R}$. Its derivative $\frac{\partial}{\partial t} \tilde{P}_a(t, \lambda)$ is Lipschitz continuous with Lipschitz constant $L_{\tilde{g}} = \max\{\frac{1}{\xi}, \frac{1}{a-1}\}$. Moreover, $P_a(t, \lambda) \leq \tilde{P}_a(t, \lambda) \leq P_a(t, \lambda) + \frac{\lambda^2}{2}$, for any $t \in \mathbb{R}$, $\lambda > 0$, and $0 < \xi < \lambda$.
Proposition 1 shows that \( \tilde{P}_a(t, \lambda) \to P_a(t, \lambda) \) as \( \xi \to 0^+ \). This justifies the use of \( \tilde{P}_a(t, \lambda) \) as a surrogate of the original SCAD penalty. Our experimental results analyze the effect of different choices of \( \xi \) (see E.2.6 in Appendix E.) and show that the parameter \( \xi \) will not affect the performance as long as it is small. Let \( \tilde{g}(t, \lambda) = \tilde{P}_a(t, \lambda) \) and employing the smoothed SCAD penalty \( \tilde{g}(t, \lambda) \), we obtain an approximate objective function \( \tilde{F}(\omega) \) of the original objective function \( F(\omega) \) in (1). In this manner, our problem is transformed into solving the following optimization problem:

\[
\min_{\omega} \{ \tilde{F}(\omega) = \sum_{i=1}^{m} f_i(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \tilde{g}(\|\omega_i - \omega_j\|, \lambda) \}.
\]

To simplify notations, we omit \( \lambda \) in \( g(\cdot, \lambda) \) and \( \tilde{g}(\cdot, \lambda) \) in the following and will specify it if necessary.

4 Fusion penalized federated clustering

Because the penalty function \( g(\|\omega_i - \omega_j\|) \) (and \( \tilde{g}(\|\omega_i - \omega_j\|) \)) is non-separable in \( \omega_i \), a natural idea is to use ADMM [2,4,19] to recast the optimization problem into separable pieces. Network Lasso [18] provides an ADMM algorithm in a distributed manner. In this section, we first present a brief overview of the Network Lasso algorithm and discuss the potential drawbacks of this algorithm in FL. Then, in Section 4.2, we propose a novel distributed optimization algorithm based on ADMM, FPFC, to address the clustered FL problem.

4.1 Network Lasso

Network Lasso (NetLasso) generalizes the group lasso to a network setting that allows simultaneous clustering and optimization on networks. In fact, our objective function in (1) resembles that of NetLasso under a fully connected network, where any two devices are connected by an edge. An ADMM algorithm has been proposed for the NetLasso problem. With ADMM, each individual node solves its own private optimization problem, broadcasts the solution to its neighbors, and repeats the process until the algorithm converges.

Despite satisfactory numerical performances, the ADMM algorithm of NetLasso has several key drawbacks such that it cannot be directly applied to the FL setup. First, at the local update step, each node (device) requires the model parameters of all its neighbors. In the setup of FL with a hub-and-spoke topology, the server must broadcast all the model parameters of \( m \) devices to each device, which will incur heavy communication costs, and more importantly, may lead to private information leaks due to the exposure of model parameters of other devices. Second, the minimization problem must be solved exactly at each node. This is impractical in FL because some devices may have low computational power which makes it difficult to ensure that each minimization step is conducted exactly, especially when \( f(\cdot) \) is a complex model, such as a neural network. Third, NetLasso requires all devices to update at each communication round, making it less practical and applicable to FL due to the constraint on communication in federated networks. In addition, NetLasso only considers the convex case. Although it can be extended to nonconvex cases, convergence analysis for the nonconvex losses has not been established.

4.2 Fusion penalized federated clustering

To address the aforementioned concerns, we propose a novel distributed optimization method called FPFC to cluster devices in the setup of FL. First, we reformulate (4) into a constrained
optimization problem by introducing new parameters $\theta_{ij} \equiv \omega_i - \omega_j$. Next, we apply a Douglas-Rachford (DR) splitting strategy employed by ADMM to minimize the augmented Lagrangian $\tilde{L}_\rho$. Finally, we revise the $\omega$-update step by utilizing an inexact randomized-block strategy to reduce the communication cost and preserve privacy. The algorithm is presented in Algorithm 1 and its derivation is given in Appendix A.

**Algorithm 1** Fusion Penalized Federated Clustering (FPFC)

| Input: | number of devices $m$; number of communication rounds $K$; number of local gradient epochs $T_i$, $i \in [m]$; stepsize $\alpha$; parameters of $\tilde{g}(\cdot)$: $\xi$, $\lambda$, $a$; and $\rho$. |
| Output: | $\{\omega_k^i\}_{i \in [m]}$. |

Initialize each device $i \in [m]$ with $\omega_i^0 = \ldots = \omega_m^0$. Initialize the server with $\zeta_{ij}^0 = \omega_{ij}^0$ for $i \in [m]$, $\theta_{ij}^0 = 0$ and $v_{ij}^0 = 0$ for $i, j \in [m]$.

for $k = 0$ to $K - 1$ do

1: [Active devices] Server randomly chooses a subset of devices $A_k$.
2: [Communication] Server downloads $\zeta_{ij}^k$ to each device $i \in A_k$.
3: [Local update] For $i \in A_k$:
   $\omega_{ij}^k = \omega_{ij}^k$.
   for $t = 0$ to $T_i - 1$ do
     \[ \omega_{ij}^{k,t+1} = \omega_{ij}^{k,t} - \alpha [\nabla f_i(\omega_{ij}^{k,t}) + \rho(\omega_{ij}^{k,t} - \zeta_{ij}^k)] \] (5)
   end for
   $\omega_{ij}^{k+1} = \omega_{ij}^{k,T_i}$.
4: [Communication] Each device $i \in A_k$ uploads $\omega_{ij}^{k+1}$ to the server.
5: [Server update] For $i, j \in A_k$ ($i < j$):
   Server computes $\delta_{ij}^{k+1} = \omega_{ij}^{k+1} - \omega_{ji}^{k+1} + \frac{v_{ij}^k}{\rho}$ and then updates
   \[
   \theta_{ij}^{k+1} = \begin{cases} 
   \frac{\xi}{\lambda + \rho} \delta_{ij}^{k+1}, & \|\delta_{ij}^{k+1}\| \leq \xi + \frac{\lambda}{\rho} \\
   (1 - \frac{\lambda}{\rho\|\delta_{ij}^{k+1}\|}) \delta_{ij}^{k+1}, & \xi + \frac{\lambda}{\rho} \leq \|\delta_{ij}^{k+1}\| \leq \lambda + \frac{\lambda}{\rho} \\
   \max\{0,1 - \frac{\lambda}{\rho\|\delta_{ij}^{k+1}\|}\} \delta_{ij}^{k+1}, & \lambda + \frac{\lambda}{\rho} < \|\delta_{ij}^{k+1}\| \leq a\lambda \\
   \frac{1 - \lambda}{(a - 1)\rho} \delta_{ij}^{k+1}, & \|\delta_{ij}^{k+1}\| > a\lambda.
   \end{cases}
   \] (6)
   $v_{ij}^{k+1} = v_{ij}^{k} + \rho(\omega_{ij}^{k+1} - \omega_{ji}^{k+1} - \theta_{ij}^{k+1})$.
   $\theta_{ji}^{k+1} = -\theta_{ij}^{k+1}$, $v_{ji}^{k+1} = -v_{ij}^{k+1}$.
   For $i \notin A_k$ and $j \notin A_k$: $\theta_{ij}^{k+1} = \theta_{ij}^k$, $v_{ij}^{k+1} = v_{ij}^k$.
   For $i \in [m]$: $\delta_{ij}^{k+1} = \frac{1}{m} \sum_{j=1}^{m} (\omega_{ij}^{k+1} + \theta_{ij}^{k+1} - \frac{v_{ij}^{k+1}}{\rho})$.

end for

\[
\min_{\omega, \theta} \frac{1}{m} \sum_{i=1}^{m} f_i(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \tilde{g}(\|\theta_{ij}\|),
\] (7)
\[
\text{s.t. } \omega_i - \omega_j = \theta_{ij}, \ i, j \in [m],
\]
where $\theta = \{\theta_{ij}^T, i, j \in [m]\}^T$.

\[
\tilde{L}_\rho(\omega, \theta, v) = \sum_{i=1}^{m} f_i(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \tilde{g}(\|\theta_{ij}\|) + \langle v_{ij}, \omega_i - \omega_j - \theta_{ij} \rangle + \frac{\rho}{2} \|\omega_i - \omega_j - \theta_{ij}\|^2,\]

8
we rely on a DR splitting scheme and can cluster the devices. Here, three iterates work. More importantly, different from the exact symmetry of communication round. For all \( i \not\in A_k \) and \( j \not\in A_k \), \( \theta_{ij}^{k+1} = \theta_{ij}^k, v_{ij}^{k+1} = v_{ij}^k \). Together with the symmetry of \( \theta \) and \( v \), the server only needs to conduct \( O(\tau(2 - \tau)m^2) \) assignment operations at each
We refer to this process as the warmup strategy for parameter tuning.

Remark 2. (Clustering in practice) As there is no available prior knowledge of the cluster structure, clustering is performed only after the algorithm converges. For the SCAD penalty, \( \theta_{ij}^K \) can be zero if devices \( i \) and \( j \) belong to the same cluster. Because we adopt the smoothed SCAD penalty, \( \theta_{ij}^K \) can only be approximately zero but not exactly zero. In practice, devices \( i \) and \( j \) are assigned to the same cluster if \( \| \theta_{ij}^K \| \leq \nu \), where \( \nu \) represents a threshold. In our experiments, we observed that the results are not very sensitive to \( \nu \), and we can obtain satisfactory results by setting \( \nu \in [\xi, 0.5] \).

As a result, we obtain \( \hat{L} \) clusters \( \hat{G}_1, \ldots, \hat{G}_L \), and \( \hat{\alpha}_l = \frac{\sum_{i \in \hat{G}_l} n_i \omega_i^l}{\sum_{i \in \hat{G}_l} n_i} \) (\( l \) in \( \hat{L} \)).

Remark 3. (Choices of initial values) In Algorithm 1, the initial values \( \omega_i^0 (i \in [m]) \) are generated randomly. Our experimental results (see E.2.2 in Appendix E.) show that the final results are very robust to the initial values although the objective function is nonconvex.

4.3 Regularization parameter tuning scheme

The regularization parameter \( \lambda \) is a key parameter of FPFC, it controls clustering and in turn prediction performance. The popular cross-validation method for parameter tuning divides the data on each device into training, testing, and validation sets, and then chooses the parameter that performs the best on the validation set. Usually, we are interested in training models not just for a single value of \( \lambda \), but for a range of values \( \{\lambda_1, \ldots, \lambda_S\} \). However, it is infeasible to train all regularization parameters in parallel, making it difficult to tune parameters in FL. Although hyperparameter selection is a vital topic, most existing works only tune parameters from a range of values sequentially, which is not plausible due to the large communication cost \([10,23,34]\). We informally have not found relevant discussions about parameter tuning in FL. In this section, we provide a practical guide to tuning the regularization parameter \( \lambda \) in FL.

Breheny and Huang \([5]\) argued that the estimated coefficients vary continuously with \( \lambda \) when the objective function is strictly convex. They constructed a decreasing sequence of values for \( \lambda \) and used the estimated coefficients based on the previous value of \( \lambda \) as the initial values for the current \( \lambda \). Inspired by this heuristic method, we present a regularization parameter tuning strategy. Specifically, given a sequence of \( \lambda \): \( \{\lambda_1, \ldots, \lambda_S\} (\lambda_1 < \lambda_2 < \ldots < \lambda_S) \), we first run FPFC algorithm with the smallest parameter \( \lambda_1 \), and then consecutively increase \( \lambda \) to the next value when certain criteria are met, such as constraints on rounds of communication or the change of accuracy. The model parameters returned by training with \( \lambda_{i-1} \) are used as the initial values for the model with \( \lambda_i \). We refer to this process as the warmup strategy for parameter tuning.

Intuitively, a small increase of \( \lambda \) will only cause a small change in the local minimizers of \( F(\omega) \), and the optimized model parameters with the tuning parameter \( \lambda_{i-1} \) will serve as a reasonably good initial value for the model with \( \lambda_i \). Therefore, the warmup strategy makes the algorithm converge very fast. During the process, we always keep the model which has the best performance on validation sets. As shown in our experiments (see Fig. 3), the validation accuracy will usually increase as \( \lambda \) increases, particularly when \( \lambda \) is relatively small. However, beyond a certain level, further increments in \( \lambda \) result in a decrease in the validation accuracy. This occurs because a large \( \lambda \) will aggregate all devices into a single cluster as illustrated in Fig 4, and destroy the underlying cluster structure. Finally, we return to train the best model obtained with \( \lambda_s \) until the algorithm converges.

For the traditional cross-validation method, the initial values corresponding to each \( \lambda_i \) are randomly generated or prespecified, thus it may be time-consuming to approach the minima point for
5 Theoretical guarantees

In this section, we first study the convergence properties of FPFC in the general setting of nonconvex loss functions. Then, we characterize the statistical convergence rate of the proposed penalization framework under a linear model with squared loss. All technical details and proofs are deferred to the appendix due to space limitations. We first make the following assumptions.

**Assumption 1.** ($L_f$-smoothness) There exists a Lipschitz constant $L_f > 0$ such that
\[ \|\nabla f_i(x) - \nabla f_i(y)\| \leq L_f \|x - y\|, \forall x, y \in \mathbb{R}^d. \]

**Assumption 2.** (Boundedness) The optimal value $F^* = \inf_{\omega \in \mathbb{R}^d} F(\omega) > -\infty$.

Assumption 1 implies that all functions $f_i(\cdot) (i \in [m])$ are continuously differentiable. Assumption 1 and 2 are standard in nonconvex optimization, and have been widely used in FL [23,34,56].

Our convergence analysis allows only a subset of devices to update parameters at each communication round. Similar to the work in [56], we define a proper sampling scheme $A$ of $[m]$, which is a random set-valued mapping with values in $2^{[m]}$, and $A_k$ is an i.i.d. realization of $A$. We make the following assumption about the sampling scheme $A$.

**Assumption 3.** For each device $i \in [m]$, there exist $p_i > 0$ such that $P(i \in A) = p_i > 0$.

According to Assumption 3, we have $p_i = \sum_{A \in A} P(A)$. Define $\hat{p} = \min_{i \in [m]} \{p_i\}$. This assumption suggests that every device has a non-zero probability of participating in the training process.

One of the improvements of FPFC over standard ADMM is that all subproblems in (9) can be solved inexactly, which is more realistic in FL. Specifically, we define the $\epsilon$-inexact solution of $\omega_i$-minimization for the $i$-th device at the $k$-th iteration below.

**Definition 1.** ($\epsilon$-inexact solution) For the function $h_i(\omega_i) = f_i(\omega_i) + \frac{\rho}{2m} \sum_{j=1}^{m} \|\omega_i - \omega_j^k - \theta_{ij}^k\|_2^2$, we say that $\omega_i^{k+1}$ is the $\epsilon$-inexact solution of $\min_{\omega_i} h_i(\omega_i)$ if there is a constant $\epsilon \in [0, 1]$ such that
\[ \|\omega_i^{k+1} - \hat{\omega}_i\| \leq \epsilon \|\omega_i^{k+1} - \omega_i^k\|, \]
where $\hat{\omega}_i$ is the exact solution of $\min_{\omega_i} h_i(\omega_i)$.

Definition 1 measures the inexactness of the solution by a relative error $\epsilon$, and such idea concept has been widely used in the literature [38,56]. Because the objective function in (1) is nonconvex, we only expect to find a stationary point $\omega^*$ of (1). Because (1) is equivalent to the constrained problem
\[
\begin{align*}
\min_{\omega, \theta} & \sum_{i=1}^{m} f_i(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} g(\|\theta_{ij}\|), \\
\text{s.t.} & \omega_i - \omega_j = \theta_{ij}, \ i, j \in [m],
\end{align*}
\]

we only expect to find a stationary point $\omega^*$ of (1). Because (1) is equivalent to the constrained problem...
each stationary point $\omega^*_{\epsilon}$ of (1) satisfies the KKT condition of (10), i.e.,

$$
\begin{align}
\nabla_\omega L_0(\omega^*, \theta^*, v^*) &= 0, \\
0 &\in \partial_\theta L_0(\omega^*, \theta^*, v^*), \\
\omega^*_i - \omega^*_j - \theta^*_{ij} &= 0, i, j \in [m],
\end{align}
$$

and vice versa, where $L_0$ is the Lagrange function of (10):

$$
L_0(\omega, \theta, v) = \sum_{i=1}^{m} f(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|\theta_{ij}\|) + \langle v_{ij}, \omega_i - \omega_j - \theta_{ij} \rangle].
$$

Let us define the mapping $\tilde{G}(\omega, \theta, v) = \theta - \text{prox}_{\lambda L_0}(\theta)$. The second condition $0 \in \partial_\theta L_0(\omega^*, \theta^*, v^*)$ in (11) is equal to $\tilde{G}(\omega^*, \theta^*, v^*) = 0$. In practice, we often wish to find an $\epsilon$-stationary point of (1) defined as follows.

**Definition 2.** $\tilde{\omega}$ is called an $\epsilon$-stationary point of (1) if there exists $\tilde{\theta}$ and $\tilde{v}$ such that

$$
\begin{align}
\mathbb{E}\|\nabla_\omega L_0(\tilde{\omega}, \tilde{\theta}, \tilde{v})\|^2 &\leq \epsilon^2, \\
\mathbb{E}\|\tilde{G}(\tilde{\omega}, \tilde{\theta}, \tilde{v})\|^2 &\leq \epsilon^2, \\
\mathbb{E}\|\{(\omega_i - \omega_j - \theta_{ij})^\top, i, j \in [m]\}^\top\|^2 &\leq \epsilon^2.
\end{align}
$$

where the expectation is taken overall the randomness generated by the underlying algorithm.

We first show that at the $k$-th iteration, $\omega_i^{k+1}$ can be an $\epsilon_i$-inexact solution of $\min_{\omega_i} h_i(\omega_i)$ if we run gradient descent in the local update of Algorithm 1 for sufficient epochs. Proof is provided in Appendix B.

**Theorem 1.** Suppose Assumption 1 holds. Assume there exists $L_\epsilon > 0$ such that $\nabla^2 f_i \succeq -L_\epsilon I$ with $\mu = \rho - L_\epsilon > 0$. For any $i \in A_k$, $\omega_i^{0,0} = \omega_i^k$. The stepsize $\alpha$ satisfies $0 < \alpha \leq \frac{1}{L_f + \rho + \mu}$. Then, for $\epsilon_i > 0$, after $T_i = \frac{2\log\left(\frac{\epsilon_i}{\epsilon_i + \mu}\right)}{\log c}$ epochs of gradient descent with $c = 1 - \alpha\frac{2\mu(L_f + \rho)}{L_f + \rho + \mu}$, we have

$$
\|\omega_i^{k+1} - \omega_i^{k+1}\| \leq \epsilon_i\|\omega_i^k - \omega_i^k\|,
$$

where $\omega_i^{k+1} = \arg \min_{\omega_i} h_i(\omega_i)$.

To prove the convergence bounds of (1), we first prove two descent lemmas (Lemmas 1 and 2) regarding $\hat{\mathcal{L}}_\rho$ during the update and then show the boundedness of $\hat{\mathcal{L}}_\rho$ from below (Lemma 3). All proofs are provided in Appendix C.

**Lemma 1.** (Descent of $\hat{\mathcal{L}}_\rho$ during $\omega$ update) Suppose the assertions in Theorem 1 holds. For each device $i \in A_k$, there exists a $\epsilon_i \in [0, 1]$ such that $T_i = \frac{2\log\left(\frac{\epsilon_i}{\epsilon_i + \mu}\right)}{\log c}$ (c is defined in Theorem 1). Let $(\omega^k, \theta^k, v^k)$ be the sequence generated by Algorithm 1. Then, for any $k \in \mathbb{N}$, we have

$$
\hat{\mathcal{L}}_\rho(\omega^k, \theta^k, v^k) - \hat{\mathcal{L}}_\rho(\omega^{k+1}, \theta^k, v^k) \geq \sum_{i \in A_k} \frac{\rho - L_f - 2(e^{-\frac{T_i}{2}} - 1)L_h}{2} \|\omega_i^k - \omega_i^{k+1}\|^2,
$$

where $L_h = L_f + \rho$. 

12
Lemma 2. (Monotonically non-increasing of $\tilde{L}_\rho$) Suppose that the assertions in Lemma 1 hold. Let $(\omega^k, \theta^k, v^k)$ be the sequence generated by the $k$-th iteration of Algorithm 1. Then, we have

$$\tilde{L}_\rho(\omega^k, \theta^k, v^k) - \tilde{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \geq \sum_{i \in A_k} \frac{\rho - L_f - 2(c^{-\frac{1}{2}} - 1)\frac{1}{L_i}}{2} \omega_i k - \omega_i^{k+1}}^2$$

$$+ \frac{1}{2m} \left( \frac{\rho - L_i}{\rho} \right) \sum_{i=1}^{m} \sum_{j=1}^{m} \| \theta_i^k - \theta_i^{k+1} \|^2.$$ 

Lemma 3. (Boundedness of $\tilde{L}_\rho$) Let $(\omega^k, \theta^k, v^k)$ be the sequence generated by the $k$-th iteration of Algorithm 1. If $\rho > \max\{ \frac{1}{\xi}, \frac{1}{\omega} \}$ and Assumption 2 holds, for any $k \in \mathbb{N}^+$, we have $\tilde{L}_\rho(\omega^k, \theta^k, v^k) \geq F^*$.

Finally, we utilize the Proposition 1 to obtain the following theorem.

Theorem 2. Given the Assumptions 1, 2, 3, and the existence of $L_\infty > 0$ such that $\nabla^2 f_i \succeq -L_i \mathbf{I}$ with $\mu = \rho - L_i > 0$. Let $(\omega^k, \theta^k, v^k)$ be the sequence generated by Algorithm 1 if $\xi, \lambda, \alpha, \rho, T_i$, and $\alpha$ are chosen such that

$$\rho > \max\{ \frac{L_f}{L^2 - 2(\xi - 1) \xi - 1}, T_i > - \frac{2 \log 2}{\log c}, 0 < \alpha \leq \frac{1}{L_f + 2 \rho - L_i},$$

where $c = 1 - \frac{2 \mu (\rho + \rho + \rho)}{L_\infty + \rho + \rho}$ and $T = \min_{i \in [m]} T_i$, then we can derive the following results:

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\| \hat{\nabla}_u \mathcal{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2] \leq \frac{C_1 [F(\omega^0) - F^* + m \xi \lambda / 4]}{K},$$

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\| \hat{\nabla}_v \mathcal{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2] \leq \frac{C_2 [F(\omega^0) - F^* + m \xi \lambda / 4]}{K},$$

where $C_1 = \frac{1}{\rho} \left( \frac{6(\xi - 1)}{\rho - L_f - 2(\xi - 1) \xi - 1} \right) L_i \rho + \frac{12 \rho^3}{\rho^2 - L_i L_i} \frac{\rho^2}{\rho^2 - L_i L_i}$, $C_2 = \frac{L_i^2}{m \rho^2 - L_i L_i - 2 L_i \rho}$.

Let $(\tilde{\omega}^K, \tilde{\theta}^K, \tilde{v}^K)$ be selected uniformly at random from $\{ (\omega^0, \theta^1, v^1), \ldots, (\omega^K, \theta^K, v^K) \}$ as the output of Algorithm 1. Suppose $\xi < \frac{\lambda \xi}{m \xi - 1}$. Then, after at most $K = \mathcal{O}(\varepsilon^{-2})$ communication rounds, $(\tilde{\omega}^K, \tilde{\theta}^K, \tilde{v}^K)$ becomes an $\varepsilon$-stationary point of (1) in the sense of Definition 1.

Remark 4. (Choice of hyperparameters) For local update steps $T_i$, the corresponding relative inexact accuracy $\epsilon_i = 1/(\xi - 1)$. The requirements on hyperparameters $T_i$, $\alpha$, $\xi$, $\lambda$, $\alpha$, and $\rho$ are not restrictive and can be easily met. For example, for any $\xi > 0, \lambda > 0, a > 0$, one can choose $\rho = \max\{3 L_f, \frac{2 \rho}{\xi}, \frac{2}{\alpha - 1}, L_i \} + 0.01, \alpha = \frac{1}{3 L_f + 2 \rho - L_i},$ and $T_i = - \frac{2 \log 3}{\log \left( 1 - \frac{2(\rho - L_i)(\rho + \rho)}{(L_f + 2 \rho - L_i)^2} \right), \epsilon_i = 0.5.$

Remark 5. (Comparison of communication complexity) Because (1) is nonconvex, our $\mathcal{O}(\varepsilon^{-2})$ communication complexity is state-of-the-art, matching the lower bound complexity (up to a constant factor) of FL on non-i.i.d. data. Although IFCA has a faster convergence rate $\mathcal{O}(\log(1/\varepsilon))$, it only considers strongly convex loss functions, while we consider more general nonconvex losses.
To characterize the statistical convergence rate, we focus on a linear model. Specifically, we assume that the data on the devices in the $l$-th cluster are generated as following: For $i \in G_l$, $y_i^s = (X_i^s, \alpha_i^{true}) + \tau_i^s$, $s \in [n_i]$, where $y_i^s \in \mathbb{R}$, $X_i^s \in \mathbb{R}^d$, $\alpha_i^{true}$ represents the true model parameters for the $l$-th cluster, and $\tau_i^s$ are the i.i.d. random errors. Further, we use the squared loss function. Denote the true values of parameters $\omega$ as $\omega^{true} = ((\omega_1^{true})^\top, \ldots, (\omega_{m}^{true})^\top)^\top$. Then, $\omega_i^{true} = \alpha_i^{true}$ for $i \in G_l$. Define the minimal differences of the common model parameters between two clusters as

$$b = \min_{i \in G_l, j \in G_{l'}, l \neq l'} \| \omega_i^{true} - \omega_j^{true} \| = \min_{l \neq l'} \| \alpha_l^{true} - \alpha_{l'}^{true} \|.$$ 

Let

$$n = \sum_{i=1}^{m} n_i, \quad n_{min} = \min_{l \in [L]} \sum_{i \in G_l} n_i, \quad n_{max} = \max_{l \in [L]} \sum_{i \in G_l} n_i,$$

and $|G_{min}| = \min_{l \in [L]} |G_l|$, where $\| \cdot \|$ is the cardinality of the set. Under the mild assumptions described in Appendix D, we can establish the following statistical convergence rate.

**Theorem 3.** Suppose that Conditions (1)-(3) in Appendix D hold. Assume that $b > a \lambda$, where the SCAD penalty function $P_n(t, \lambda)$ is constant when $|t| > a \lambda$. If

$$\lambda \gg m|G_{min}|^{-1} \sqrt{Ld^{3}n_{min}^{-1} \log n} + m \sqrt{n_{min}^{-1}d \log n},$$

then there exists a local minimizer $\omega^* = ((\omega_1^*)^\top, \ldots, (\omega_{m}^*)^\top)^\top$ of objective function $F(\omega)$ such that

$$P(\sup_{i \in [m]} \| \omega_i^* - \omega_i^{true} \| \leq \Lambda_n) \geq 1 - 2Ldn^{-n_{min}n_{max}^{-1}} - 2dn^{-1},$$

where $\Lambda_n = C^{-1}|G_{min}|^{-1}c_1^{-1/2} \sqrt{2Ld^{3}n_{min}^{-1} \log n}$, and $C$ and $c_1$ are positive constants defined in Conditions (1) and (3), respectively.

Theorem 3 demonstrates that the objective function (1) has a local minimizer $\omega^*$ that each element $\omega_i^*$ of $\omega^*$ stays within a ball of radius $\Lambda_n$ centered at the true model parameters $\omega_i^{true}$. $\Lambda_n$ is the uniform convergence rate of the least squares estimator of model parameters of all devices when the cluster structure is known. Similar to the convergence rate of least squares estimators in the conventional setting, $\Lambda_n$ decreases at a rate proportional to the sample size of the smallest cluster. The hyperparameter $\lambda$ is critical in FPFC. On the one hand, $\lambda$ must be less than the upper bound $b/a$ such that different devices in different clusters can be distinguished, while on the other hand, it must be greater than the lower bound listed in Theorem 3 such that the devices in the same cluster can be grouped together.

### 6 Performance evaluation

In this section, we conduct extensive experiments to evaluate the performance of FPFC. We first describe the experimental settings and benchmark datasets. Then, we show the advantage of the proposed regularization parameter tuning strategy. Besides, we compare the performance of different methods in aspects of robustness, generality, and communication efficiency. We also present several variations of FPFC in practice. More experimental details are provided in Appendix E. Our code is released at [https://github.com/xueyu-ubc/FPFC](https://github.com/xueyu-ubc/FPFC).
6.1 Experimental settings

Datasets and models. We conduct experiments on different synthetic datasets and three real-world datasets:

- **Synthetic.** We begin with a multi-classification problem with synthetic data. There are \( m = 100 \) devices, and the number of samples on each device follows a power law, in the range \([250, 25810]\). There are \( L = 4 \) clusters, each consisting of 25 devices. For each cluster \( l \in [L] \), we first generate a weight matrix \( W_l \in \mathbb{R}^{10 \times 60} \) and a bias \( b_l \in \mathbb{R}^{10} \). Each element in \( W_l \) and \( b_l \) is independently drawn from \( N(\mu_l, 1) \), where \( \mu_l \sim N(0, 1) \). For \( i \in G_l \), the features \( X_{si} \) for \( s \in [n_i] \) are independently generated from \( N(0, I_{60}) \), and response \( y_{si} \) is then generated from \( y_{si}^* = \arg \max(\text{softmax}(W_l X_i + b_l + \tau_{si}^*)) \), where each noise vector \( \tau_{si}^* \) is generated from \( N(0, 0.5^2 I_{10}) \). We use the cross-entropy loss functions.

- **Housing and Body fat (H&Bf).** We consider a linear regression task based on the Housing and Body fat (H&Bf) benchmark datasets [12]. We set \( m = 8 \) devices. To simulate the cluster structure of devices, we evenly allocate the Housing dataset to the first six devices and the Body fat dataset to the last two devices, resulting in \( L = 2 \) clusters. We use a linear model with squared loss.

- **MNIST & FMNIST.** We use two public benchmark datasets, MNIST [31] and FMNIST [60]. Take MNIST as an example, we first partition 60000 training images and 10000 testing images into \( m = 20 \) devices by a Dirichlet distribution, each of which belongs to one of \( L = 4 \) clusters. We then modify data on each device by randomly swapping two labels [50]. Specifically, the first cluster consists of devices 1-5, where each device’s data labeled as "0" are relabeled as "8" and vice versa. The second cluster consists of devices 6-10, where each device’s data labeled as "1" and "7" are swapped similarly, and so on. The testing data are processed in the same way. We use a multilayer convolutional neural network with the cross-entropy loss. To account for some common properties of the data, we adopt the weight sharing technique from multi-task learning [6], as used in IFCA. Specifically, the weights for the first three layers are common across all devices and clustering is only applied for the last layer, producing a predicted label for each image.

Evaluation metrics. We focus on four metrics for comparisons: (1) testing accuracy (Acc) for classification problems or testing loss for linear regression problems measured by RMSE between the predicted responses and true responses, both of which are averaged over all devices, (2) number of identified clusters (Num), (3) adjusted Rand index (ARI), which measures the agreement between the identified cluster structure and true cluster structure, and (4) communication cost (Cost), which is measured by the number of parameters that need to be transmitted during the communication process. For each dataset, we run all algorithms with three random seeds and report the average and standard deviation of the four metrics.

Setup. We evaluate the performance of FPFC and compare it with the state-of-the-art clustered FL algorithms (IFCA, CFL, and PACFL). To make our experiments more comprehensive, we also report the performance of FedAvg and two personalized FL algorithms (Per-FedAvg and LG-FedAvg (LG) [36]). LOCAL is a baseline that independently trains the personalized model of each device without collaboration. Moreover, we implement FPFC-\( \ell_1 \), a variant of FPFC, which uses an \( \ell_1 \) penalty, i.e., \( g(\|\omega_i - \omega_j\|, \lambda) = \lambda\|\omega_i - \omega_j\|_2 \). Note that FedAMP requires exact computations of each intermediate minimization subproblem, making it unrealistic for FL, we do not consider this method.

Hyperparameter. We randomly split the data on each device into an 80% training set and a 20% testing set, where the training set is used to train models, and the testing set is to make
predictions. We further divide the training set for each device into 80% for training and 20% for validation, where the validation set is used to select hyperparameters. For CFL, we tune $\varepsilon_1$ in the range $[0, 1]$ and $\varepsilon_2$ in the range $[\varepsilon_1, 10\varepsilon_1]$, as suggested in [50]. For FPFC and FPFC-$\ell_1$, we tune $\lambda$ in the range $[0, 5]$ to achieve the lowest RMSE on the validation set for H&BF and $[0, 1]$ to achieve the highest Acc for all synthetic data, MNIST and FMNIST. For a fair comparison, we do not adopt the warmup tuning scheme, but we discuss the advantages of different tuning strategies later. In addition, we set $\rho = 1$, $a = 3.7$, and $\xi = 10^{-4}$. For IFCA, we implement gradient averaging in local updates and give the correct number of clusters in advance. For PACFL, there is a threshold controlling the number of clusters and we tune it in a range that depends upon different datasets. For all algorithms, each device performs 10 epochs of local GD updates for all synthetic data, 20 epochs for H&BF, and SGD updates for 10 epochs and a batch size of 100 for MNIST and FMNIST. The learning rates are 0.1 for all synthetic data and 0.01 for H&BF. For MNIST and FMNIST, we initialize the learning rate to 0.01 and multiply it by 0.9 for every 5 communication rounds. For each dataset, we randomly sample 30% $\sim$ 50% devices to perform the local update at each communication round.

### 6.2 Performance comparison

![Figure 3: Visualization of $||\omega_i - \omega_j||$ computed by FPFC(upper) and the cluster structure by using t-SNE (below) at 0-th, 5-th, 50-th, and 600-th communication round under $\lambda = 0.6$.](image)

Fig.3 visualizes $||\omega_i - \omega_j||$ for $(i, j \in [m])$ on a synthetic dataset over the course of 600 communication rounds. We use t-distributed stochastic neighbor embedding (t-SNE) [58] to visualize the cluster structure of $m = 100$ devices. As can be seen, $\omega_i$ converge to four clusters when $\lambda = 0.6$. These

| Algorithm | Synthetic | H&BF |
|-----------|-----------|------|
| LOCAL     | $84.97\% \pm 0.06$ | $4.89 \pm 0.19$ |
| FedAvg    | $30.35\% \pm 0.05$ | $6.00 \pm 0.36$ |
| LG        | $69.72\% \pm 0.06$ | $4.52 \pm 0.39$ |
| Per-FedAvg| $56.26\% \pm 0.06$ | $5.29 \pm 0.27$ |
| IFCA      | $60.42\% \pm 0.17$ | $5.08 \pm 0.21$ |
| CFL       | $86.60\% \pm 0.05$ | $1.33 \pm 0.47$ |
| PACFL     | $88.30\% \pm 0.06$ | $4.68 \pm 0.41$ |
| FPFC      | $89.46\% \pm 0.04$ | $4.68 \pm 0.41$ |
| H&BF      | $89.46\% \pm 0.04$ | $4.68 \pm 0.41$ |

Table 1: Experimental results on synthetic and H&BF datasets.
results illustrate that FPFC can effectively identify the cluster structure. Table III presents a statistical summary of four metrics on synthetic and H&BF datasets. Because LOCAL, FedAvg, LG, and Per-FedAvg lack the capability to cluster devices, we have omitted reporting their Num and ARI results. FedAvg exhibits the worst performance as expected, which can be attributed to its attempt to learn a single model for all devices, despite their distinct data distributions. These results highlight the limitations of FedAvg in terms of individualized predictions and cluster-based analysis. Both Per-FedAvg and LG have shown superior performance compared to FedAvg. This can be attributed to their ability to personalize the global model for each device. Due to overfitting to local data, LOCAL has worse prediction performance than FPFC. The two clustered FL algorithms, IFCA and CFL, do not perform as well as FPFC. Although they can also cluster devices, their clustering performances are inferior to that of FPFC. Although IFCA is implemented with the correct number of clusters, some clusters have no device in the end, resulting in fewer clusters. PACFL demonstrates competitive results, but it relies on a critical threshold parameter that controls the clustering results. FPFC-ℓ₁ performs better than IFCA and CFL in clustering. However, due to the use of ℓ₁ penalty in FPFC-ℓ₁, when the coefficients of devices that belong to different clusters have significant differences, FPFC-ℓ₁ imposes a stronger penalty to compress these coefficients, leading to incorrect grouping. In contrast, FPFC does not penalize large coefficient differences, and therefore the result of FPFC-ℓ₁ is significantly inferior to FPFC in both clustering and prediction. LOCAL incurs zero communication costs as it does not involve any communication with the server. FedAvg, LG, Per-FedAvg, FPFC, and FPFC-ℓ₁ have the same communication costs due to the consistent size of the transmitted information at each communication round. The communication cost of PACFL is determined by the combined cost of FedAvg and the cost associated with hierarchical clustering in one-shot clustering. IFCA has the highest communication cost due to its sensitivity to initialization, which needs more communication rounds to achieve convergence. Additionally, during each communication round, IFCA requires downloading all cluster models to all devices, resulting in a substantial increase in communication costs.

Fig. 4 shows the training loss, training accuracy and testing accuracy results of all methods on MNIST and FMNIST under some fine-tuned parameters. FPFC also achieves a comparable high test accuracy with only a small communication cost. Despite IFCA demonstrating favorable results
on the FMNIST dataset, as shown in Fig. 5, it comes at the expense of the highest communication costs. We also consider different cluster structures, the effect of regularization parameter $\lambda$, and the tolerance to heterogenous training, see more results in Appendix E.2.

6.3 Regularization parameter tuning

To compare different strategies for regularization parameter tuning, we conduct diverse experiments. The four subgraphs in the first row of Fig. 6 show the process of parameter tuning via warmup. Taking the synthetic dataset as an example, we tune $\lambda$ from $\{0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5\}$. Initially, we set $\lambda = 0$ and run Algorithm 1 until the change in accuracy on the validation set becomes smaller than a predefined tolerance threshold, e.g., $10^{-4}$. Afterward, we use the model obtained with $\lambda = 0$ as the initial model for $\lambda = 0.5$. We then run FPFC with $\lambda = 0.5$ until it converges (i.e., the change in the validation accuracy is below a specified threshold). As observed from the results, the final validation accuracy achieved with $\lambda = 0.5$ is higher than that obtained with the previous $\lambda = 0$. Thus, we retain the model obtained with $\lambda = 0.5$ as the initial model for subsequent $\lambda = 1$. Then, we start to train FPFC with $\lambda = 1$ based on the initial model and repeat this process. As shown in

Figure 5: Total communication cost on MNIST/ FMNIST.

Figure 6: Two strategies for regularization parameter tuning on different datasets.
Table 2: Selected λ by two strategies and running time on synthetic, H&BF, MNIST, and FMNIST datasets.

|          | Synthetic | H&BF | MNIST | FMNIST |
|----------|-----------|------|-------|--------|
|          | λ         | Time | λ     | Time   | λ     | Time   |
| Separate | 0.5       | 4.02h| 4     | 39.41s | 0.1   | 3.94h  |
| Warmup   | 3         | 0.46h| 4     | 14.74s | 0.4   | 0.58h  |

the first subgraph in the first row of Fig. 6, once λ increases to 3.5, the validation accuracy begins to decline, indicating that λ = 3.5 is too large for training. Therefore, we stop increasing λ and revert to training FPFC using the model obtained with λ = 3 as the initial model until the algorithm converges. The bar charts display the running time(in seconds) of the two strategies for each λ during the parameter tuning process on various datasets. Notably, both strategies tune λ within the same range. The bar charts demonstrate that the warmup strategy which initializes the model with the latest model parameters as λ increases indeed speeds up convergence and saves computing time.

The four subgraphs in the second row of Fig. 6 show the generalization performance of the two strategies. For FPFC with separate tuning, the initial values corresponding to each λ_i are randomly generated. Subsequently, we run FPFC using λ_i on the training set and assess its performance (accuracy or RMSE) on the validation set. Finally, we compute the testing accuracy/RMSE of FPFC using the λ that shows the best performance on the validation set. Take the synthetic dataset as an example, the testing accuracies during 0-68th communication rounds are obtained based on the models with λ = 0. The 69th-443th, 444th-458th, 459th-463th, 464th-605th, 606th-623th, 624th-634th, 635th-780th testing results are based on models with λ = 0.5, 1, 1.5, 2, 2.5, 3, 3.5, respectively. Due to the lower validation accuracy observed with λ = 3.5 compared to λ = 3, we finally run the latest model trained with λ = 3 from the 635th communication round and continue running it until the maximum number of iterations is reached. During the warmup stage, the initial values associated with the subsequent λ values gradually converge towards local minimizers. As a result, using better initial values can potentially yield improved results. The final selected regularization parameter and total running time are reported in Table 2. Utilizing the warmup scheme significantly reduces the running time, for example, reducing it to 0.58 hours with warmup compared to 3.94 hours when tuning parameters separately on the MNIST dataset.

6.4 Practical considerations

In this part, we explore the performance of FPFC with all the mentioned state-of-the-art baselines, focusing on robustness and generality. Additionally, we introduce several practical variants of FPFC, including asynchronous updates (asyncFPFC) and a communication-efficient approach. Through these evaluations, we aim to gain valuable insights into the strengths and limitations of FPFC in practical application scenarios.

6.4.1 Robustness

One of the most common issues in FL is that it can suffer from security threats to model training, including malicious devices, communication failures, and adversarial attacks. A significant challenge to the robustness of distributed systems is known as Byzantine failure or Byzantine fault [30]. Byzantine attacks can occur when participating devices provide completely arbitrary or misleading behaviors to the server, thereby disrupting the training process. To address Byzantine failures
in FL, several approaches have been proposed to enhance the robustness of FL by designing a Byzantine-robust FL scheme \cite{sadeghi2020towards,maity2020byzantine,maity2020byzantine,maity2020byzantine}.

To evaluate the robustness of different algorithms, similar to the work of \cite{maity2020byzantine}, we consider three different types of Byzantine model update attacks in FL training. Define the model update sent by a malicious device $k$ is $\tilde{\omega}_k$, then the three types of attacks can be defined as:

(a) Same-value attack: the malicious device $k$ sends the constant value $\tilde{\omega}_k = c\mathbf{I}$ to the server as its model update, where $\mathbf{I} \in \mathbb{R}^d$ is the vector of ones and $c \sim \mathcal{N}(0, \sigma^2)$.

(b) Sign-flipping attack: the malicious device $k$ alters the sign of its model update before sending it to the server. $\tilde{\omega}_k = -|c|\omega_k$, where $c \sim \mathcal{N}(0, \sigma^2)$.

(c) Gaussian attack: the malicious device $k$ sends gaussian noise $\tilde{\omega}_k \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ to the server.

To evaluate the robustness of the personalization layers in the model, we conduct experiments involving attacks with varying levels of noise. For the three Byzantine attacks, the noise level $\sigma$ is set as 100, 10, 100 respectively, as specified in \cite{maity2020byzantine}. We consider different ratios of malicious devices, ranging from 0.0 to 0.8 with increments of 0.2. Fig. 7 shows the average test accuracy of benign devices on the FMNIST dataset for different methods under the three Byzantine attacks. As we can see, FPFC consistently outperforms other baselines and it is the most stable algorithm across different attack scenarios. For full results on synthetic and MNIST datasets, please refer to Appendix E.2.

The robustness of FPFC can be attributed to the utilization of the nonconvex pairwise fusion penalization. According to the formulation of the SCAD penalty $P_a(t, \lambda)$ \cite{fan2001variable}, $P_a(t, \lambda)$ will become a constant when $|t| > a\lambda$, resulting in the ineffective shrinkage of parameter values. The difference between malicious updates and normal updates in FPFC is typically greater than the predefined threshold $a\lambda$ or can be controlled by appropriately selecting the parameters $\lambda$ and $a$. This characteristic limits the impact of attacks on the training process of FPFC. As a result, FPFC maintains its robustness and reliability under different attacks.

6.4.2 Generalization to newcomers

When newcomers join after the federation procedure, FPFC can leverage the models of other devices to help newcomers learn their personalized models. The newcomer $i$ conducts local training first
## Table 3: Average local test accuracy and standard deviation across newcomers on different datasets.

The last column represents the average ranking of all algorithms across three datasets.

| Method  | Synthetic | MNIST | FMNIST | Rank |
|---------|-----------|-------|--------|------|
| LOCAL   | 79.00% ± 0.08 | 90.81% ± 0.06 | 75.29% ± 0.06 | 4.67 |
| FedAvg  | 38.73% ± 0.29 | 73.67% ± 0.04 | 66.26% ± 0.06 | 8.67 |
| LG      | 66.22% ± 0.11 | 95.71% ± 0.03 | 80.25% ± 0.07 | 3.33 |
| Per-FedAvg | 43.26% ± 0.26 | 96.62% ± 0.01 | 77.94% ± 0.06 | 3.67 |
| IFCA    | 33.57% ± 0.28 | **96.65% ± 0.01** | 73.93% ± 0.10 | 5.33 |
| CFL     | 69.43% ± 0.33 | 95.46% ± 0.02 | 70.66% ± 0.07 | 5.00 |
| PACFL   | 71.88% ± 0.30 | 74.45% ± 0.05 | 66.42% ± 0.06 | 6.33 |
| FPFC-ℓ1 | 67.72% ± 0.10 | 92.73% ± 0.05 | 75.54% ± 0.08 | 5.00 |
| FPFC    | **79.31% ± 0.08** | 93.51% ± 0.06 | 75.73% ± 0.07 | **3.00** |

and then sends its model to the server and the server computes the values of \( \theta_{ij} \) and \( v_{ij} \) between newcomer \( i \) and other previously participating devices \( j \). The server then sends the corresponding \( \zeta_i \) to the newcomer \( i \) and the newcomer updates its local model based on the received \( \zeta_i \). This process is repeated until convergence is achieved. As not all baselines have provided official approaches for handling newcomers after training, we developed the following strategies for each method. For the LOCAL method, each newcomer updates its model using its local data. For FedAvg, newcomers utilize the final global model obtained from the server. For LG and Per-FedAvg, newcomers receive the final global model from the server and fine-tune it. For IFCA, each newcomer determines its cluster identity by selecting the cluster with the lowest loss. For CFL and PACFL, we adopt the official approaches described in their papers to obtain personalized models for newcomers.

To evaluate the performance of newcomers’ models, we consider three scenarios where 20% of the participants join the federated network after the federated training process. Table 3 reports the average local test accuracy and standard deviation of the newcomers on three different datasets. As observed, FedAvg has the worst performance as the global model is not well-suited for all devices. The two personalized baselines, LG and Per-FedAvg, outperform FedAvg as they enable newcomers to fine-tune the global model based on their local data. Compared to other baselines, FPFC has comparable performance and demonstrates good generalization capabilities on various datasets.

### 6.4.3 AsyncFPFC

Note that in FPFC, the server needs to select a subset of devices and requires synchronous aggregation at each communication round. However, such algorithms may suffer from synchronization delays due to slower devices, especially in the FL setting. Asynchronous methods have been studied widely due to their advantages over synchronized methods [1, 7, 15, 56]. They do not need a global synchronous aggregation at each round and the server can perform updates once it receives the information from one device. Although the implementation and debugging of asynchronous algorithms are difficult, such methods can make more efficient use of computing resources, leading to faster convergence. In this part, we propose asyncFPFC, an asynchronous variant of FPFC. The detailed implementation of asyncFPFC is provided in Appendix A.

To illustrate that our algorithm can employ the asynchronous update strategy and the advantage of asyncFPFC over FPFC, we conduct diverse experiments using synthetic datasets and three real datasets. The number of computing nodes is 8 for H&BF data and 20 for synthetic, MNIST, and FMNIST datasets. For FPFC, we randomly sample 3 nodes to perform the synchronous aggregation...
for H&BF dataset, and 5 nodes for synthetic, MNIST, and FMNIST. Since the nodes on a computing cloud have similar configurations, we follow the procedure used by [56], where they simulate the devices’ heterogeneity of computation and communication by adding variable delays to each node’s update process, such that some nodes have a slower speed to update and upload its models to the server. Empirically, we set the computing node to have a random delay in the range [0, 20] seconds for H&BF and synthetic, and [0, 800] seconds for MNIST and FMNIST. The results are depicted in Fig. 8. As it can be seen, asyncFPFC outperforms FPFC in terms of training time when the computing power of the nodes is heterogeneous.

6.4.4 Communication-efficient FPFC

Communication is the key challenge in FL due to the limited bandwidth and device instability. Several different approaches are proposed to address this issue by increasing the computation over communication rounds, such as Local SGD [11, 53]. However, taking many steps to update each device’s local model will lead to immediate local over-fitting. A heuristic communication strategy is to gradually increase the number of local updates. Intuitively, at the beginning of training, devices do not have any prior information about the data distribution, so frequent communication among devices is required to collaboratively train models. As each device updates its local model towards its optima, less communication is required.

Following the procedure in [53], we compare the performance of FPFC using different communication strategies. For a fair comparison, we have fixed the total number of local iterations $K \times T$ to a constant value, where $K$ is the communication rounds and $T$ is the number of local updates. Specifically, for synthetic and H&BF datasets, we run FPFC with local GD updates, and the total iterations $K \times T$ is $10^4$ for the synthetic dataset, 800 for H&BF. We compare different strategies, namely, local GD with a constant step $T = 5$, local GD with $T = 10$, and local GD with a growing step, $T = \lceil k/20 \rceil$ for synthetic, $T = k + 1$ for H&BF. For MNIST and FMNIST, we run FPFC with local mini-batch SGD updates, and the total iterations $K \times T$ is $10^3$. We compare local SGD with
a constant step $T = 5$, $T = 10$, and local SGD with a growing step, $T = \lceil k/2 \rceil$. When the total number of iterations $K \times T$ is fixed, conducting a different number of local updates $T$ will result in different communication rounds. For example, for the MNIST dataset with $K \times T = 10^3$, applying different strategies such as $T = 5$, $T = 10$, and $T = \lceil k/2 \rceil$ will lead to communication rounds of 200, 100, and 63, respectively. Fig. 9 shows that the communication strategy with increasing local steps outperforms all the other methods. For MNIST, FPFC with a growing local step can achieve 86.62% accuracy after 62 communication rounds, whereas FPFC with $T = 10$ reaches 85.31%, and FPFC with $T = 5$ only attains 77.05% after the same rounds. For FMNIST, after 62 communication rounds, the testing accuracy obtained by FPFC with a growing local step is 68.28%, for FPFC with $T = 10$ and $T = 5$, the accuracies are 65.62% and 59.34%, respectively. We conclude that frequent communication at the beginning is beneficial to reduce the communication rounds.

7 Conclusion

In this paper, we introduced a novel clustered FL method called FPFC to address the challenges of FL where devices are distributed and partitioned into clusters. By employing a nonconvex pairwise fusion penalty, FPFC can identify the cluster structure and estimate the optimal models in each cluster. We have provided convergence guarantees for FPFC with nonconvex objectives and established the statistical convergence rate under a linear model. Additionally, we proposed a warmup strategy for regularization parameter tuning in FL settings. Experimental results have demonstrated the advantages of FPFC, including its superior performance in terms of prediction accuracy, clustering capability, robustness to attacks, generalization ability, and communication efficiency. However, it is worth noting that FPFC has a key limitation in terms of its slow convergence rate due to the utilization of the ADMM scheme in FL. Future directions involve integrating state-of-the-art privacy-preserving methods and analyzing the convergence results of FPFC with the warmup tuning strategy and asyncFPFC algorithm.
References

[1] By Mahmoud Assran, Arda Aytekin, Hamid Reza Feyzmahdavian, Mikael Johansson, and Michael G. Rabbat. Advances in asynchronous parallel and distributed optimization. *Proceedings of the IEEE*, 108(11):2013–2031, 2020.

[2] Jianchao Bai, Deren Han, Hao Sun, and Hongchao Zhang. Convergence on a symmetric accelerated stochastic admm with larger stepsizes. *CSIAM Transactions on Applied Mathematics*, 3(3):448–479, 2022.

[3] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[4] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

[5] Patrick Breheny and Jian Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *The annals of applied statistics*, 5(1):232–253, 2011.

[6] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[7] Sorathan Chaturapruek, John C Duchi, and Christopher Ré. Asynchronous stochastic convex optimization: the noise is in the noise and SGD don’t care. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[8] Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. LAG: Lazily Aggregated Gradient for Communication-Efficient Distributed Learning. In *Advances in Neural Information Processing Systems (NeurIPS 2018)*, volume 32, pages 5050–5060. Curran Associates, Inc., 2018.

[9] Yae Jee Cho, Jianyu Wang, Tarun Chirvolu, and Gauri Joshi. Communication-efficient and model-heterogeneous personalized federated learning via clustered knowledge transfer. *IEEE Journal of Selected Topics in Signal Processing*, 17(1):234–247, 2023.

[10] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *ArXiv*, abs/2003.13461, 2020.

[11] Aymeric Dieuleveut and Kumar Kshitij Patel. Communication trade-offs for local-sgd with large step size. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[12] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[13] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *Advances in Neural Information Processing Systems*, volume 33, pages 3557–3568. Curran Associates, Inc., 2020.

[14] Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.
[15] Hamid Reza Feyzmahdavian and Mikael Johansson. Asynchronous Iterations in Optimization: New Sequence Results and Sharper Algorithmic Guarantees. arXiv preprint arXiv:2109.04522, 2021.

[16] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 2017. PMLR.

[17] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An Efficient Framework for Clustered Federated Learning. In Advances in Neural Information Processing Systems (NeurIPS 2020), volume 34, pages 19586–19597. Curran Associates, Inc., 2020.

[18] David Hallac, Jure Leskovec, and Stephen Boyd. Network Lasso: Clustering and Optimization in Large Graphs. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’15, page 387–396, New York, NY, USA, 2015. ACM.

[19] Deren Han. A Survey on Some Recent Developments of Alternating Direction Method of Multipliers. Journal of the Operations Research Society of China, 10:1–52, 2022.

[20] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. arXiv, 2020.

[21] David Harrison and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. Journal of Environmental Economics and Management, 5(1):81–102, 1978.

[22] Toby Hocking, Jean-Philippe Vert, Francis Bach, and Armand Joulin. Clusterpath: an Algorithm for Clustering using Convex Fusion Penalties. In Proceedings of the 28th International Conference on Machine Learning (ICML2011), ICML ’11, pages 745–752, New York, NY, USA, June 2011. ACM.

[23] Shaoming Huang, Yong Zhou, Ting Wang, and Yuanming Shi. Byzantine-resilient federated machine learning via over-the-air computation. In 2021 IEEE International Conference on Communications Workshops (ICC Workshops), pages 1–6, 2021.

[24] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. Personalized Cross-Silo Federated Learning on Non-IID Data. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 7865–7873, May 2021.

[25] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonavitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D’Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayler Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jinyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and Open Problems in Federated Learning. Foundations and Trends® in Machine Learning, 14(1-2):1–210, 2021.
[26] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In Proceedings of the 37th International Conference on Machine Learning, volume 119 of Proceedings of Machine Learning Research, pages 5132–5143. PMLR, 13–18 Jul 2020.

[27] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. arXiv preprint arXiv:1610.05492, 2016.

[28] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. In 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), 2020.

[29] S.W. Lagakos. The challenge of subgroup analyses—reporting without distorting. New England Journal of Medicine, 354(16):1667–1669, 2006.

[30] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. ACM Trans. Program. Lang. Syst., 4(3):382–401, jul 1982.

[31] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In Proceedings of the IEEE, volume 86, pages 2278–2324, 1998.

[32] Qing Li and Byeong Man Kim. Clustering approach for hybrid recommender system. In Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003), pages 33–38, Halifax, NS, Canada, 2003.

[33] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated Learning: Challenges, Methods, and Future Directions. IEEE Signal Processing Magazine, 37(3):50–60, 2020.

[34] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated Optimization in Heterogeneous Networks. In Proceedings of machine learning and systems, volume 2, pages 429–450, 2020.

[35] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the Convergence of FedAvg on Non-IID Data. In International Conference on Learning Representations, 2020.

[36] Paul Pu Liang, Terrance Liu, Liu Ziyin, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. arXiv preprint arXiv:2001.01523, 2020.

[37] Siyun Lin, Yuze Han, Xiang Li, and Zhihua Zhang. Personalized Federated Learning towards Communication Efficiency, Robustness and Fairness. In Advances in Neural Information Processing Systems, 2022.

[38] Y. Liu, Y. Xu, and W. Yin. Acceleration of Primal-Dual Methods by Preconditioning and Simple Subproblem Procedures. Journal of Scientific Computing, 86(21), 2021.

[39] S. Ma, J. Huang, Z. Zhang, and M. Liu. Exploration of heterogeneous treatment effects via concave fusion. The International Journal of Biostatistics, 16(1):20180026, 2019.

[40] Shujie Ma and Jian Huang. A concave pairwise fusion approach to subgroup analysis. Journal of the American Statistical Association., 112(517):410–423, 2017.
[41] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three Approaches for Personalization with Applications to Federated Learning. *arXiv preprint arXiv:2002.10619*, 2020.

[42] O. Marfoq, G. Neglia, A. Bellet, L. Kameni, and R. Vidal. Federated Multi-Task Learning under a Mixture of Distributions. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, volume 35, 2021.

[43] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282, Fort Lauderdale, Florida, USA, 2017.

[44] Yinbin Miao, Ziteng Liu, Hongwei Li, Kim-Kwang Raymond Choo, and Robert H. Deng. Privacy-preserving byzantine-robust federated learning via blockchain systems. *IEEE Transactions on Information Forensics and Security*, 17:2848–2861, 2022.

[45] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic Federated Learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4615–4625. PMLR, 2019.

[46] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. volume 33, pages 8024–8035, 2019.

[47] K. Pelckmans, J. D. Brabanter, Jak Suykens, and B. D. Moor. Convex Clustering Shrinkage. In *PASCAL Workshop on Statistics and Optimization of Clustering Workshop*, 2005.

[48] Boris T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., New Yock, 1987.

[49] Amit Portnoy, Yoav Tirosh, and Danny Hendler. Towards federated learning with byzantine-robust client weighting. *Applied Sciences*, 12(17), 2022.

[50] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3710–3722, 2021.

[51] Donald Shenaj, Eros Fani, Marco Toldo, Debora Caldarola, Antonio Tavaera, Umberto Michieli, Marco Ciccone, Pietro Zanuttigh, and Barbara Caputo. "learning across domains and devices: Style-driven source-free domain adaptation in clustered federated learning. In *Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2023.

[52] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated Multi-Task Learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[53] Artin Spiridonoff, Alex Olshevsky, and Yannis Paschalidis. Communication-efficient sgd: From local sgd to one-shot averaging. In *Advances in Neural Information Processing Systems*, volume 34, pages 24313–24326. Curran Associates, Inc., 2021.
[54] Sebastian Urban Stich. Local SGD Converges Fast and Communicates Little. In *ICLR 2019-International Conference on Learning Representations*, New Orleans, USA, 2019.

[55] Canh T. Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21394–21405. Curran Associates, Inc., 2020.

[56] Q. Tran-Dinh, N. H. Pham, D. T. Phan, and L. M. Nguyen. FedDR – Randomized Douglas-Rachford Splitting Algorithms for Nonconvex Federated Composite Optimization. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, volume 35, 2021.

[57] Saeed Vahidian, Mahdi Morafah, Weijia Wang, Vyacheslav Kungurtsev, Chen Chen, Mubarak Shah, and Bill Lin. Efficient distribution similarity identification in clustered federated learning via principal angles between client data subspaces. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37:10043–10052, 06 2023.

[58] Laurens Van Der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2605):2579–2605, 2008.

[59] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv*, 2019.

[60] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[61] Xue Yu, Ziyi Liu, Yifan Sun, and Wu Wang. Clustered federated learning for heterogeneous data (student abstract). *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(13):16378–16379, Sep. 2023.

[62] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. Fully decentralized joint learning of personalized models and collaboration graphs. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 864–874. PMLR, 26–28 Aug 2020.

[63] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu. FedPD: A Federated Learning Framework With Adaptivity to Non-IID Data. *IEEE Transactions on Signal Processing*, 69:6055–6070, 2021.

[64] Yu Zhang, Duo Liu, Moming Duan, Li Li, Xianzhang Chen, Ao Ren, Yujuan Tan, and Chengliang Wang. Fedmds: An efficient model discrepancy-aware semi-asynchronous clustered federated learning framework. *IEEE Transactions on Parallel and Distributed Systems*, 34(3):1007–1019, 2023.
Appendix
Clustered Federated Learning based on Nonconvex Pairwise Fusion

A Details of Algorithm 1

We first reformulate (4) into a constrained problem. Next, we apply the classical DR splitting scheme to the constrained reformulation and present the standard ADMM algorithm. Then, we revise the \( \omega \)-step in a distributed and inexact manner. Finally, we randomize its local update step to obtain the final algorithm.

Constrained reformulation. By introducing a set of new parameters \( \theta_{ij} = \omega_i - \omega_j \), we can rewrite (4) as the following constrained optimization problem:

\[
\min_{\omega, \theta} \sum_{i=1}^{m} f_i(\omega_i) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \hat{g}(|\theta_{ij}|) \\
\text{s.t. } \omega_i - \omega_j = \theta_{ij}, \ i, j \in [m].
\]

Standard ADMM algorithm. The solution of the aforementioned problem can be obtained by minimizing the following augmented Lagrangian:

\[
\tilde{L}_\omega(\omega, \theta, v) = f(\omega) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ \hat{g}(|\theta_{ij}|) + \langle v_{ij}, \omega_i - \omega_j - \theta_{ij} \rangle + \frac{\rho}{2} \| \omega_i - \omega_j - \theta_{ij} \|^2 \right], \tag{A.1}
\]

where \( f(\omega) = \sum_{i=1}^{m} f_i(\omega_i) \), \( \theta = \{ \theta_{ij}^T, i, j \in [m] \}^T \), \( v = \{ v_{ij}^T, i, j \in [m] \}^T \), and \( \rho > 0 \) is the penalty parameter. We apply the DR splitting method adopted by ADMM to (8) to obtain the following iterations:

\[
\omega^{k+1} = \arg\min_{\omega} \sum_{i=1}^{m} f_i(\omega_i) + \frac{\rho}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \| \omega_i - \omega_j - \theta_{ij}^k + \frac{v_{ij}^k}{\rho} \|^2
\]

\[
\theta_{ij}^{k+1} = \arg\min_{\theta_{ij}} \hat{g}(|\theta_{ij}|) + \frac{\rho}{2} \| \omega_i^{k+1} - \omega_j^{k+1} + \frac{v_{ij}^k}{\rho} - \theta_{ij} \|^2 \tag{A.2}
\]

\[
v_{ij}^{k+1} = v_{ij}^k + \rho (\omega_i^{k+1} - \omega_j^{k+1} - \theta_{ij}^{k+1}) \tag{A.3}
\]

where the second and third steps are conducted for all \( i < j \). For \( i > j \), the updated quantities can be directly obtained via \( \theta_{ij} = -\theta_{ji} \) and \( v_{ij} = -v_{ji} \), and for \( i = j \), \( \theta_{ii} = v_{ii} = 0 \).

Distributed and inexact update of \( \omega \). In practice, the \( \omega \)-update step, can be conducted in a distributed manner, i.e., \( \omega^{k+1} \) can be obtained by running the following sub-iteration for sufficiently many steps, for \( c = 0, \ldots, C - 1 \):

\[
\omega_i^{k,:+1} = \arg\min_{\omega_i} f_i(\omega_i) + \frac{\rho}{2m} \sum_{j=1}^{m} \| \omega_i - \omega_j^{k,c} - \theta_{ij}^k + \frac{v_{ij}^k}{\rho} \|^2, \ i \in [m] \tag{A.4}
\]

where \( \omega_i^{k,0} = \omega_i^k \) and \( \omega_i^{k+1} \approx \omega_i^{k,C} \) for \( i \in [m] \). However, this parallel approach has three main drawbacks in the FL setting. First, the central server has to send all model parameters to all devices at each sub-iteration, resulting in large communication cost, especially when the number of devices or/sub-iteration rounds is large. Second, each step must solve a minimization problem exactly, which is impractical in the FL setup due to the heterogeneous computational power of devices.
Lastly, each device must know the current model parameters of all other devices, which may result in privacy leaks.

As such, to reduce communication cost, we only conduct sub-iteration \(\mathcal{A}_4\) for one step, i.e., \(C = 1\), instead of multiple steps. Considering the system heterogeneity of a federated network, we allow the minimization \(\mathcal{A}_4\) to be solved approximately. Specifically, we define \(h_i(\omega_i) = \frac{1}{3m} \sum_{j=1}^{m} \|\omega_i - \omega_j^k - \theta_{ij}^k + \frac{v_{ij}^k}{\rho}\|^2\). We replace the exact solution \(\hat{\omega}_i^{k+1} = \arg\min_{\omega_i} h_i(\omega_i)\) by an approximated solution \(\omega_i^{k+1} \approx h_i(\omega_i)\) up to a given accuracy \(\epsilon_i\) such that \(\|\omega_i^{k+1} - \hat{\omega}_i^{k+1}\| \leq \epsilon\|\omega_i^k - \hat{\omega}_i^k\|\). The approximated solution \(\omega_i^{k+1}\) can be obtained by running \(T_i\) epochs of local gradient decent updates

\[
\omega_i^{k,t+1} = \omega_i^{k,t} - \alpha (\nabla f_i(\omega_i^{k,t}) + \rho(\omega_i^{k,t} - \zeta_i^k)), \quad t = 0, \ldots, T_i - 1, \quad (A.5)
\]

where \(\alpha\) is the step size, \(\zeta_i^k = \frac{1}{m} \sum_{j=1}^{m} (\omega_j^k + \theta_{ij}^k - \frac{v_{ij}^k}{\rho})\) and \(\omega_i^{k,0} = \omega_i^k\). We set \(\omega_i^{k+1} = \omega_i^{k,T_i}\). Because \(\zeta_i^k\) is an average of parameters, the \(i\)-th device cannot infer the model parameters or private data of any other devices, which preserves the privacy of devices. In practice, \(\nabla f_i(\omega_i^{k,t})\) can be full or stochastic gradient of the local empirical loss \(f_i\) at \(\omega_i^{k,t}\).

Putting \(\mathcal{A}_2\), \(\mathcal{A}_3\), and \(\mathcal{A}_5\) together, we obtain the following distributed and inexact ADMM variant:

\[
\begin{align*}
\omega_i^{k,0} &= \omega_i^k, \forall i \in [m] \\
\omega_i^{k,t+1} &= \omega_i^{k,t} - \alpha (\nabla f_i(\omega_i^{k,t}) + \rho(\omega_i^{k,t} - \zeta_i^k)), \quad t = 0, \ldots, T_i - 1, \\
\omega_i^{k+1} &= \omega_i^{k,T_i}, \forall i \in [m] \\
\theta_{ij}^{k+1} &= \arg\min_{\theta_{ij}} \left\{ \frac{1}{2} \|\theta_{ij}\| + \frac{\rho}{2} \|\omega_i^{k+1} - \omega_j^{k+1} - \theta_{ij}^{k+1}\|^2 \right\}, \\
v_{ij}^{k+1} &= v_{ij}^{k} + \rho(\omega_i^{k+1} - \omega_j^{k+1} - \theta_{ij}^{k+1}),
\end{align*}
\]

and \(\theta_{ji}^{k+1} = -\theta_{ij}^{k+1}, v_{ji}^{k+1} = -v_{ij}^{k+1}, \forall i < j\).

Randomized-block update. Instead of performing updates for all devices as in \(\mathcal{A}_6\), we adopt a randomized-block strategy, where only a subset of devices \(\mathcal{A}_k\) perform local updates then send their local models to server. Then, for the inactive devices \(i \notin \mathcal{A}_k\), the local model is unchanged, i.e., \(\omega_i^{k+1} = \omega_i^k\). Hence, there is no need for communication between these inactive devices and the server. Moreover, for any two inactive devices \(i \perp j\), the parameters \(\theta_{ij}\) and \(v_{ij}\) are also unchanged, i.e., \(\theta_{ij}^{k+1} = \theta_{ij}^k\) and \(v_{ij}^{k+1} = v_{ij}^k\). As a result, the server only needs to update the parameters between two devices with at least one active device.

Full version of Algorithm 1. Finally, we obtain the full algorithm of FPFC (Algorithm 2). For completeness, we also provide the algorithm of FPFC-\(\ell_1\).

Algorithm of asyncFPFC. Here, we provide the full algorithm of asyncFPFC (Algorithm 3).

B Proof of Theorem 1

Proof. Because \(f_i\) is \(L_f\)-smooth and \(\mu = \rho - L_\mu > 0\), \(h_i\) is \(L_h\)-smooth with \(L_h = L_f + \rho\) and \(\mu\)-strongly convex. By the standard results of gradient descent \(48\), after \(t\) epochs, we have

\[
\|\omega_i^{k+1,t+1} - \hat{\omega}_i^{k+1}\|^2 \leq c^t \|\omega_i^{k} - \hat{\omega}_i^{k+1}\|^2,
\]
Algorithm 2 FPFC and FPFC-$\ell_1$

**Input:** number of devices $m$; number of communication rounds $K$; number of local gradient epochs $T_i$, $i \in [m]$; stepsize $\alpha$; parameters of $\tilde{g}()$: $\xi$, $\lambda$, $a$; and $\rho$.

**Output:** $\{\omega_i^k\}_{i \in [m]}$.

Initialize each device $i \in [m]$ with $\omega_i^0 = \ldots = \omega_m^0$. Initialize the server with $\zeta_i^0 = \omega_i^0$ for $i \in [m]$, $\theta_i^0 = 0$ and $v_i^0 = 0$ for $i, j \in [m]$.

for $k = 0$ to $K - 1$ do

1: [**Active devices**] Server randomly selects a subset of devices $A_k$.
2: [**Communication**] Each device $i \in A_k$ sends $\zeta_i^k$ to each device $i \in A_k$.
3: [**Local update**] For each device $i \in A_k$:

   $\omega_i^{k+1} = \omega_i^k$

   for $t = 0$ to $T_i - 1$ do

   $\omega_i^{k,t+1} = \omega_i^{k,t} - \alpha [\nabla f_i(\omega_i^{k,t}) + \rho (\omega_i^{k,t} - \zeta_i^k)]$

   end for

   $\omega_i^{k+1} = \omega_i^{k,T_i}$.

Each device $i \notin A_k$ does nothing, i.e., $\omega_i^{k+1} = \omega_i^k$.

4: [**Communication**] Each device $i \in A_k$ sends $\omega_i^{k+1}$ back to the server.
5: [**Server update**] For $i \in A_k$ or $j \in A_k$ ($i < j$):

Server computes $\delta_{ij}^{k+1} = \omega_i^{k+1} - \omega_j^{k+1} + \frac{v_{ij}^k}{\rho}$ and updates

**FPFC:**

$$\theta_{ij}^{k+1} = \begin{cases} \frac{\xi}{\lambda + \xi \rho} \delta_{ij}^{k+1}, & \|\delta_{ij}^{k+1}\| \leq \xi + \frac{\lambda}{\rho}, \\ \left(1 - \frac{\lambda}{\rho \|\delta_{ij}^{k+1}\|}\right)\delta_{ij}^{k+1}, & \xi + \frac{\lambda}{\rho} \leq \|\delta_{ij}^{k+1}\| \leq \lambda + \frac{\lambda}{\rho}, \\ \max\{0, 1 - \frac{a \lambda}{\|\delta_{ij}^{k+1}\|}\} \delta_{ij}^{k+1}, & \lambda + \frac{\lambda}{\rho} < \|\delta_{ij}^{k+1}\| \leq a \lambda, \\ \delta_{ij}^{k+1}, & \|\delta_{ij}^{k+1}\| > a \lambda. \end{cases}$$

**FPFC-$\ell_1$:**

$$\theta_{ij}^{k+1} = \max\{0, 1 - \frac{\lambda}{\rho \|\delta_{ij}^{k+1}\|}\} \delta_{ij}^{k+1}.$$ 

$v_{ij}^{k+1} = v_{ij}^k + \rho (\omega_i^{k+1} - \omega_j^{k+1} - \theta_{ij}^{k+1})$.

$\theta_{ji}^{k+1} = -\theta_{ij}^{k+1}$, $v_{ji}^{k+1} = -v_{ij}^{k+1}$.

For $i \notin A_k$ and $j \notin A_k$: $\theta_{ij}^{k+1} = \theta_{ij}^k$, $v_{ij}^{k+1} = v_{ij}^k$.

For $i \in [m]$, server updates $\zeta_i^{k+1} = \frac{1}{m} \sum_{j=1}^{m} (\omega_j^{k+1} + \theta_{ij}^{k+1} - \frac{v_{ij}^{k+1}}{\rho})$.

end for
Algorithm 3 Asynchronous FPFC (asyncFPFC)

Input: number of devices $m$; number of communication rounds $K$; number of local gradient epochs $T_i, i \in [m]$; stepsize $\alpha$; parameters of $\bar{g}(\cdot)$: $\xi, \lambda, a$; and $\rho$.

Output: $\{\omega_i^K\}_{i \in [m]}$.

Initialize each device $i \in [m]$ with $\omega_i^0 = \ldots = \omega_m^0$. Initialize the server with $\zeta_i^0 = \omega_i^0$ for $i \in [m]$, $\theta_i^0 = 0$ and $v_i^0 = 0$ for $i, j \in [m]$.

for $k = 0$ to $K - 1$ do

1: [Communication] Device $i_k$ completes its local update and then sends $\omega_i^{k+1}$ back to the server. Other devices maintain $\omega_i^{k+1} = \omega_i^k$ for $i \neq i_k$.

2: [Server update] Server computes $\delta_{ikj}^{k+1} = \omega_i^{k+1} - \omega_j^{k+1} + \frac{v_{ij}^k}{\rho}$ and updates

$$
\theta_{ikj}^{k+1} = \begin{cases} 
\xi \rho \frac{\delta_{ikj}^{k+1}}{\lambda + \rho}, & ||\delta_{ikj}^{k+1}|| \leq \xi + \frac{\lambda}{\rho} \\
(1 - \frac{\lambda}{\rho ||\delta_{ikj}^{k+1}||}) \delta_{ikj}^{k+1}, & ||\delta_{ikj}^{k+1}|| > \xi \leq \lambda + \frac{\lambda}{\rho} \\
\max(0,1-\frac{\lambda}{(a-1)\rho \delta_{ikj}^{k+1}}) \delta_{ikj}^{k+1}, & \lambda + \frac{\lambda}{\rho} < ||\delta_{ikj}^{k+1}|| \leq a\lambda \\
\delta_{ikj}^{k+1}, & ||\delta_{ikj}^{k+1}|| > a\lambda.
\end{cases}
$$

$$
v_{ij}^{k+1} = v_{ikj}^k + \rho(\omega_i^{k+1} - \omega_j^{k+1} - \theta_{ij,ikj}^{k+1}).
\theta_{ij,ikj}^{k+1} = -\theta_{ikj}^{k+1}, \quad v_{ij}^{k+1} = -v_{ikj}^k.
$$

For other devices $i \neq i_k$: $\theta_{ij,ikj}^{k+1} = 0$, $v_{ij}^{k+1} = v_{ikj}^k$.

Server updates $\zeta_i^{k+1} = \frac{1}{m} \sum_{j=1}^{m} (\omega_j^{k+1} + \theta_{ij,ikj}^{k+1} - \frac{v_{ij}^{k+1}}{\rho})$.

3: [Communication] Server sends $\zeta_i^{k+1}$ to device $i_k$.

end for

where $\hat{\omega}_i^{k+1}$ is the exact solution of $\min_{\omega_i} h_i(\omega_i)$ and $c = 1 - \alpha \frac{2\mu L}{\rho + \mu}$. Then, we get

$$\|\omega_i^{k+1,t+1} - \hat{\omega}_i^{k+1}\| \leq (\sqrt{c})^t \|\omega_i^k - \hat{\omega}_i^{k+1}\| \leq (\sqrt{c})^t (\|\omega_i^k - \omega_i^{k+1,t+1}\| + \|\omega_i^{k+1,t+1} - \hat{\omega}_i^{k+1}\|).
$$

Thus, we obtain

$$\|\omega_i^{k+1,t+1} - \hat{\omega}_i^{k+1}\| \leq \frac{(\sqrt{c})^t}{1 - (\sqrt{c})^t} \|\omega_i^{k+1,t+1} - \omega_i^k\|. \quad (B.1)
$$

For any $\epsilon_i > 0$, as long as $t \geq \frac{2\log(\frac{\epsilon_i}{\sqrt{c} + \epsilon_i})}{\log c}$, we have

$$\frac{(\sqrt{c})^t}{1 - (\sqrt{c})^t} \leq \epsilon_i. \quad (B.2)
$$

The proof can be completed by combing (B.1) and (B.2). □

C Proof of Theorem 2

To save space, we introduce a matrix

$$
A_{ij} = [0_d \times (i-1)d, I_d, 0_{(j-i)d}, -I_d, 0_d \times (m-j-1)d] \in \mathbb{R}^{d \times md},
$$

32
and then we have $\omega_i - \omega_j = A_{ij} \omega$ for $i, j \in [m]$. We define

\begin{align*}
\eta_{ij}^k &= \nu_{ij}^{k+1} + \rho (\theta_{ij}^{k+1} - \theta_{ij}^k + \omega_{ij}^{k+1} - \omega_{ij}^k), \\
\tilde{\eta}_{ij}^k &= \nu_{ij}^{k+1} + \rho (\theta_{ij}^{k+1} - \theta_{ij}^k), \quad i, j \in [m].
\end{align*}

(C.1)\hspace{1cm} (C.2)

Note that $\eta_{ij}^k = \tilde{\eta}_{ij}^k + \rho (\omega_{ij}^{k+1} - \omega_{ij}^k)$ and $\tilde{\eta}_{ij}^k = - \tilde{\eta}_{ji}^k$. At each communication round, we allow only a subset of devices $i \in A_k$ to perform local updates. The subset $A_k$ is an i.i.d. realization of $\hat{A}$, and $\hat{A}$ is the collection of all subsets of $[m]$. Let $F_k = \sigma(A_0, ..., A_k)$ be the $\sigma$-algebra generated by $A_0, ..., A_k$. Our analysis allows arbitrary sampling strategies that satisfy Assumption 3. For convenience, we assume that at each communication round of FPFC, all $v_{ij}$ and $\theta_{ij}(i, j \in [m])$ are updated by $\{A, 2\}$ and $\{A, 3\}$, respectively.

### C.1 Proof of Lemma 1

**Proof.** From the definition (8) of the augmented Lagrangian $\tilde{\mathcal{L}}_\rho(\omega, v, \{A, 3\}, \{C, 1\}, \{C, 2\})$, we have

\[
\tilde{\mathcal{L}}_\rho(\omega^k, \theta^k, v^k) - \tilde{\mathcal{L}}_\rho(\omega^{k+1}, \theta^k, v^k) = \left\{ f(\omega^k) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [ \langle \nu_{ij}^k, A_{ij} \omega - \theta_{ij}^k \rangle + \frac{\rho}{2} \| A_{ij} \omega - \theta_{ij}^k \|^2 ] - \{ f(\omega^{k+1}) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [ \langle \nu_{ij}^k, A_{ij} \omega - \theta_{ij}^k \rangle + \frac{\rho}{2} \| A_{ij} \omega - \theta_{ij}^k \|^2 ] \right\}
\]

\[
= f(\omega^k) - f(\omega^{k+1}) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [ \langle \nu_{ij}^k, A_{ij} \omega^k - A_{ij} \omega^{k+1} \rangle + \frac{\rho}{2} \| A_{ij} \omega^k - A_{ij} \omega^{k+1} \|^2 + \rho (A_{ij} \omega^{k+1} - \theta_{ij}^k, A_{ij} \omega^k - A_{ij} \omega^{k+1}) ]
\]

by $\{A, 3\}$,

\[
\tilde{\mathcal{L}}_\rho(\omega^k, \theta^k, v^k) - \tilde{\mathcal{L}}_\rho(\omega^{k+1}, \theta^k, v^k) = \left\{ f(\omega^k) - f(\omega^{k+1}) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [ \langle \nu_{ij}^k, \omega^k - \omega^{k+1} \rangle + \rho (\theta_{ij}^{k+1} - \theta_{ij}^k, \omega_{ij}^k - \omega_{ij}^{k+1}) ] + \frac{\rho}{2} \| (\omega_{ij}^k - \omega_{ij}^{k+1}) \|^2 - \frac{\rho}{2} \| A_{ij} \omega^k - A_{ij} \omega^{k+1} \|^2 
\]

\[
= f(\omega^k) - f(\omega^{k+1}) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [ \langle \nu_{ij}^k, \omega_{ij}^k - \omega_{ij}^{k+1} \rangle + \langle \nu_{ij}^k, \omega_{ij}^k - \omega_{ij}^{k+1} \rangle + \frac{\rho}{2} \| \omega_{ij}^k - \omega_{ij}^{k+1} \|^2 + \rho \theta_{ij}^{k+1} - \theta_{ij}^k, \omega_{ij}^k - \omega_{ij}^{k+1} \rangle + \frac{\rho}{2} \| \omega_{ij}^k - \omega_{ij}^{k+1} \|^2 
\]

\[
\frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \rho \| \omega_{ij}^k - \omega_{ij}^{k+1} \|^2 + \frac{\rho}{2} \| \omega_{ij}^k - \omega_{ij}^{k+1} \|^2 = \frac{\rho}{2} \sum_{i=1}^{m} \| \omega_{ij}^k - \omega_{ij}^{k+1} \|^2 
\]

(C.3)
to obtain the last equality. Using the relation between \( \eta^k_{ij} \) and \( \tilde{\eta}^k_{ij} \), we have

\[
\frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ (\eta^k_{ij}, \omega_i^k - \omega_j^{k+1}) + (\tilde{\eta}^k_{ij}, \omega_i^{k+1} - \omega_j^k) \right] \\
= \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left( \eta^k_{ij}, \omega_i^k - \omega_j^{k+1} \right) + (\eta^k_{ij}, \omega_i^{k+1} - \omega_j^k) - \rho \| \omega_j^{k+1} - \omega_j^k \|^2 \\
= \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ (\eta^k_{ij}, \omega_i^k - \omega_j^{k+1}) + (\eta^k_{ij}, \omega_i^{k+1} - \omega_j^k) + (\omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1}) - \rho \| \omega_j^{k+1} - \omega_j^k \|^2 \right] \\
= \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ 2 \left( \eta^k_{ij}, \omega_i^k - \omega_j^{k+1} \right) + (\eta^k_{ij}, \omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1}) \right] + \rho \left( \omega_j^{k+1} - \omega_j^k, \omega_i^{k+1} - \omega_i^k \right).
\]

Because

\[
\frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left( \eta^k_{ij}, (\omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1}) \right) \\
= \frac{1}{2m} \sum_{i<j} (\eta^k_{ij}, (\omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1})) + (\eta^k_{ij}, (\omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1})) \\
= \frac{1}{2m} \sum_{i<j} (\eta^k_{ij}, (\omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1})) + (\eta^k_{ij}, (\omega_i^{k+1} - \omega_j^k) - (\omega_i^k - \omega_j^{k+1})) = 0,
\]

we have

\[
\frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ (\eta^k_{ij}, \omega_i^k - \omega_j^{k+1}) + (\tilde{\eta}^k_{ij}, \omega_j^{k+1} - \omega_j^k) \right] = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ (\eta^k_{ij}, \omega_i^k - \omega_i^{k+1}) + \frac{\rho}{2} \left( \omega_i^{k+1} - \omega_i^k, \omega_j^{k+1} - \omega_j^k \right) \right].
\]

Combining (C.3) and (C.4), we obtain

\[
\bar{\mathcal{L}}_\rho(\omega^k, \theta^k, \nu^k) - \bar{\mathcal{L}}_\rho(\omega^{k+1}, \theta^k, \nu^k) = f(\omega^k) - f(\omega^{k+1}) + \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{m} \left[ (\eta^k_{ij}, \omega_i^k - \omega_i^{k+1}) + \frac{\rho}{2} \left( \omega_i^{k+1} - \omega_i^k, \omega_j^{k+1} - \omega_j^k \right) \right] + \frac{\rho}{2} \frac{m}{m} \left\| \omega_i^k - \omega_i^{k+1} \right\|^2.
\]

Let \( r^k_i = f_i(\omega^k_i) - f_i(\omega_i^{k+1}) + \frac{1}{m} \sum_{j=1}^{m} \left[ (\eta^k_{ij}, \omega_i^k - \omega_i^{k+1}) + \frac{\rho}{2} \left( \omega_i^{k+1} - \omega_i^k, \omega_j^{k+1} - \omega_j^k \right) \right] + \frac{\rho}{2} \left\| \omega_i^k - \omega_i^{k+1} \right\|^2.
\]

For \( i \notin \mathcal{A}_k \), \( \omega_i^{k+1} = \omega_i^k \); thus, we have

\[
\bar{\mathcal{L}}_\rho(\omega^k, \theta^k, \nu^k) - \bar{\mathcal{L}}_\rho(\omega^{k+1}, \theta^k, \nu^k) = \sum_{i \in \mathcal{A}_k} r^k_i.
\]

Define \( h_i(\omega_i) = f_i(\omega_i) + \frac{\rho}{2m} \sum_{j=1}^{m} \| \omega_i - \omega_j^k - \theta^k_{ij} + \frac{v^k_{ij}}{\rho} \|^2 \). For \( i \in \mathcal{A}_k \), using (C.1), we have

\[
\nabla h_i(\omega_i^{k+1}) = \nabla f_i(\omega_i^{k+1}) + \frac{\rho}{m} \sum_{j=1}^{m} (\omega_i^{k+1} - \omega_j^k - \theta^k_{ij} + \frac{v^k_{ij}}{\rho}) \nabla f_i(\omega_i^{k+1}) + \frac{m}{m} \eta^k_{ij}.
\]

Using the last expression and \( L_f \)-smoothness of \( f_i \), we can derive that

\[
r^k_i \geq \frac{\rho}{2m} \sum_{j=1}^{m} (\omega_i^{k+1} - \omega_j^k, \omega_j^{k+1} - \omega_i^k) + \frac{\rho - L_f}{2} \| \omega_i^k - \omega_i^{k+1} \|^2 + \left\langle \nabla h_i(\omega_i^{k+1}), \omega_i^k - \omega_i^{k+1} \right\rangle.
\]
Summing (C.7) for $i \in A_k$, we obtain
\[
\sum_{i \in A_k} r_i^k \geq \frac{\rho}{2m} \sum_{i \in A_k} (\omega_i^{k+1} - \omega_i^k)^2 + \sum_{i \in A_k} \frac{\rho - L_f}{2} \|\omega_i^k - \omega_i^{k+1}\|^2 + \sum_{i \in A_k} \langle \nabla h_i(\omega_i^{k+1}), \omega_i^k - \omega_i^{k+1} \rangle
\]
\[
\geq \sum_{i \in A_k} \frac{\rho - L_f}{2} \|\omega_i^k - \omega_i^{k+1}\|^2 - \sum_{i \in A_k} \|\nabla h_i(\omega_i^{k+1})\| \|\omega_i^k - \omega_i^{k+1}\|.
\]
(C.8)

According to Theorem 1, for each device $i \in A_k$, there exists an $\epsilon_i \in [0, 1]$ such that $T_i = \frac{2 \log(\frac{1}{4\epsilon_i})}{\log c}$, and we have $\|\omega_i^{k+1} - \omega_i^k\| \leq (e^{-\frac{T_i}{2}} - 1)^{-1} \|\omega_i^{k+1} - \omega_i^k\|$. Then, due to the $L_h$-smoothness of $h_i$, we have that for $i \in A_k$
\[
\|\nabla h_i(\omega_i^{k+1})\| \leq L_h \|\omega_i^{k+1} - \omega_i^k\| \leq (e^{-\frac{T_i}{2}} - 1)^{-1} L_h \|\omega_i^{k+1} - \omega_i^k\|.
\]
(C.9)

Inserting (C.9) into (C.8), we obtain
\[
\sum_{i \in A_k} r_i^k \geq \sum_{i \in A_k} \frac{\rho - L_f - 2(e^{-\frac{T_i}{2}} - 1)^{-1} L_h}{2} \|\omega_i^k - \omega_i^{k+1}\|^2.
\]
Together with (C.5), we then obtain
\[
\tilde{L}_\rho(\omega, \theta, v^k) - \tilde{L}_\rho(\omega, \theta^k, v^k) \geq \sum_{i \in A_k} \frac{\rho - L_f - 2(e^{-\frac{T_i}{2}} - 1)^{-1} L_h}{2} \|\omega_i^k - \omega_i^{k+1}\|^2.
\]
(C.10)

This completes the proof.

\[\square\]

### C.2 Proof of Lemma 2

**Proof.** From the definition of the augmented Lagrangian $\tilde{L}_\rho(\omega, \theta, v)$ (8), we have
\[
\tilde{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1}) = \tilde{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^k) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \langle v_i^{k+1} - v_i^k, \mathbf{A}_{ij} \omega^{k+1} - \theta_{ij}^{k+1} \rangle
\]
\[
\tilde{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^k) + \frac{1}{2m\rho} \sum_{i=1}^{m} \sum_{j=1}^{m} \|v_i^{k+1} - v_i^k\|^2
\]
(C.11)

and
\[
\tilde{L}_\rho(\omega^k, \theta^k, v^k) = \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \langle v_i^k - v_i^{k+1}, \mathbf{A}_{ij} \omega^k - \theta_{ij}^k \rangle + \frac{\rho}{2} \|\mathbf{A}_{ij} \omega^k - \theta_{ij}^k\|^2 - \frac{\rho}{2} \|\mathbf{A}_{ij} \omega^{k+1} - \theta_{ij}^{k+1}\|^2.
\]
(C.12)

Using (A.2), we obtain that
\[
\nabla \tilde{g}(\|\theta_{ij}^k\|) = v_i^k + \rho(\omega_i^k + \omega_j^k - \theta_{ij}^k) = v_i^{k+1}.
\]
(C.13)

From the definition of smoothed SCAD penalty, we know that $\tilde{g}$ is $L_{\tilde{g}}$-smooth with $L_{\tilde{g}} = \max\{\frac{1}{\hat{\tau}}, \frac{1}{a-1}\}$. Using the $L_{\tilde{g}}$-smoothness of $\tilde{g}$ and (C.13), we can derive from (C.12) that
\[
\tilde{L}_\rho(\omega^{k+1}, \theta^k, v^k) \geq \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \langle v_i^{k+1} - v_i^k, \theta_{ij}^k - \theta_{ij}^{k+1} \rangle + \frac{\rho}{2} \|\mathbf{A}_{ij} \omega^k - \theta_{ij}^k\|^2
\]
\[
- \frac{\rho}{2} \|\mathbf{A}_{ij} \omega^{k+1} - \theta_{ij}^{k+1}\|^2 - \frac{L_{\tilde{g}}}{2} \|\theta_{ij}^k - \theta_{ij}^{k+1}\|^2.
\]
(C.14)
By (A.3), we have
\[
\langle v_{ij}^{k+1} - v_{ij}^k, \theta_{ij}^k - \theta_{ij}^{k+1} \rangle + \frac{\rho}{2} \|A_{ij} \omega^{k+1} - \theta_{ij}^k\|^2 \overset{(A.3)}{=} \langle v_{ij}^{k+1} - v_{ij}^k, \theta_{ij}^k - \theta_{ij}^{k+1} \rangle + \frac{\rho}{2} \|v_{ij}^{k+1} - v_{ij}^k\|^2 + \frac{\rho}{2} \|\theta_{ij}^{k+1} - \theta_{ij}^k\|^2.
\]
Inserting the last equality into (C.14), we can derive that
\[
\hat{L}_\rho(\omega^{k+1}, \theta^k, v^k) - \hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^k) \geq \frac{1}{2m} \sum_{i=1}^m \sum_{j=1}^m \left( \frac{1}{2\rho} \|v_{ij}^{k+1} - v_{ij}^k\|^2 + \frac{\rho}{2} \|\theta_{ij}^{k+1} - \theta_{ij}^k\|^2 - \frac{L_{\tilde{g}}}{2} \|\theta_{ij}^k - \theta_{ij}^{k+1}\|^2 \right)
\]
Using the $L_{\tilde{g}}$-smoothness of $\tilde{g}$ and (C.13), we can bound $\|v_{ij}^{k+1} - v_{ij}^k\|$ as
\[
\|v_{ij}^{k+1} - v_{ij}^k\| \leq L_{\tilde{g}} \|\theta_{ij}^{k+1} - \theta_{ij}^k\|.
\]
By (C.15) and (C.16), we can further derive from (C.11) that
\[
\hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \geq \hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^k) + \frac{1}{2m} \sum_{i=1}^m \sum_{j=1}^m \|v_{ij}^{k+1} - v_{ij}^k\|^2 - \frac{L_{\tilde{g}}}{2} \sum_{i=1}^m \sum_{j=1}^m \|\theta_{ij}^k - \theta_{ij}^{k+1}\|^2.
\]
Together with Lemma 1, we have
\[
\hat{L}_\rho(\omega^{k}, \theta^k, v^k) - \hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \geq \sum_{i \in A_k} \frac{\rho - L_f - 2(\epsilon - \frac{\rho}{\rho} - 1) - L_h}{2} \|\omega_i^k - \omega_i^{k+1}\|^2
\]
\[
+ \frac{1}{2m} \left( \frac{\rho - L_{\tilde{g}}}{2} - \frac{L_{\tilde{g}}}{\rho} \right) \sum_{i=1}^m \sum_{j=1}^m \|\theta_{ij}^k - \theta_{ij}^{k+1}\|^2.
\]
This completes the proof. \qed

### C.3 Proof of Lemma 3

**Proof.** Let us define $u_{ij}^k = A_{ij} \omega^k$. We have
\[
f(\omega^k) + \frac{1}{2m} \sum_{i=1}^m \sum_{j=1}^m \tilde{g}(\|u_{ij}^k\|) \geq \hat{F}(\omega^*) \geq \min_{\omega, \theta_{ij}} \left\{ f(\omega) + \frac{1}{2m} \sum_{i=1}^m \sum_{j=1}^m \tilde{g}(\|\theta_{ij}\|) : A_{ij} \omega - \theta_{ij} = 0 \right\}.
\]
Using this expression and definition of \( \tilde{\mathcal{L}}_\rho(\omega, \theta, v) \), we have

\[
\tilde{\mathcal{L}}_\rho(\omega^k, \theta^k, v^k) = f(\omega^k) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|\theta^k_{ij}\|) + \langle v^k_{ij}, A_{ij}\omega^k - \theta^k_{ij} \rangle + \frac{\rho}{2} \|A_{ij}\omega^k - \theta^k_{ij}\|^2]
\]

\[
= f(\omega^k) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|\theta^k_{ij}\|) + \langle \nabla g(\|\theta^k_{ij}\|), u^k_{ij} - \theta^k_{ij} \rangle + \frac{\rho}{2} \|A_{ij}\omega^k - \theta^k_{ij}\|^2]
\]

\[
\geq f(\omega^k) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|u^k_{ij}\|) - \frac{L_\bar{g}}{2} \|A_{ij}\omega^k - \theta^k_{ij}\|^2 + \frac{\rho}{2} \|A_{ij}\omega^k - \theta^k_{ij}\|^2]
\]

\[
= f(\omega^k) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|u^k_{ij}\|) + \frac{\rho - L_\bar{g}}{2} \|A_{ij}\omega^k - \theta^k_{ij}\|^2] \geq F^*,
\]

where the first inequality holds from the \( L_{\bar{g}} \)-smoothness of \( \bar{g} \), and the last inequality holds because of \( \rho > \max\{\frac{1}{\xi}, \frac{1}{4-\xi}\} \), Assumption 2, and

\[
F^* - \bar{F}(\bar{\omega}^*) = f(\omega^*) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|\omega^*_i - \omega^*_j\|) - f(\omega^*) - \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \tilde{g}(\|\bar{\omega}^*_i - \bar{\omega}^*_j\|)]
\]

\[
\leq f(\bar{\omega}^*) + \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} [g(\|\bar{\omega}^*_i - \bar{\omega}^*_j\|) - f(\bar{\omega}^*) - \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \tilde{g}(\|\bar{\omega}^*_i - \bar{\omega}^*_j\|)] 
\]

The last inequality holds because of Proposition 1. \( \square \)

### C.4 Proof of Theorem 2

**Proof.** (a) First,

\[
\nabla_{\omega} \tilde{\mathcal{L}}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) = \nabla f_i(\omega_i^{k+1}) + \frac{1}{2m} \sum_{j=1}^{m} [(v_{ij}^{k+1} - v_{ij}^{k+1})] = \nabla f_i(\omega_i^{k+1}) + \frac{1}{m} \sum_{j=1}^{m} v_{ij}^{k+1}.
\]

Using (C.6), we can derive from the last equality that

\[
\nabla_{\omega} \tilde{\mathcal{L}}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) = \nabla h_i(\omega_i^{k+1}) - \frac{\rho}{m} \sum_{j=1}^{m} (\omega_j^{k+1} - \omega_j^k + \theta_j^{k+1} - \theta_j^k).
\]

Then, by (C.9) and the Jensen inequality, for \( i \in A_k \), we have

\[
\|\nabla_{\omega} \tilde{\mathcal{L}}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})\|^2 \leq 3\left( c^{-\frac{T}{2}} - 1 \right)^{-2} L_h^2 \|\omega_i^{k+1} - \omega_i^k\|^2 + \frac{\rho^2}{m} \sum_{j=1}^{m} \|\omega_j^{k+1} - \omega_j^k\|^2 + \frac{\rho^2}{m} \sum_{j=1}^{m} \|\theta_j^{k+1} - \theta_j^k\|^2.
\]

Furthermore,

\[
\sum_{i \in A_k} \|\nabla_{\omega} \tilde{\mathcal{L}}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})\|^2 \leq 3 \sum_{i \in A_k} \left( c^{-\frac{T}{2}} - 1 \right)^{-2} L_h^2 + \rho^2 \|\omega_i^{k+1} - \omega_i^k\|^2 + \frac{3\rho^2}{m} \sum_{i=1}^{m} \sum_{j=1}^{m} \|\theta_j^{k+1} - \theta_j^k\|^2.
\]

Because \( \rho > \max\{\frac{L_f}{1 - 2c^{-\frac{T}{2}}}, \frac{2\lambda}{\xi}, \frac{2}{4-\xi}\} \), we have \( \frac{\rho - L_f - 2(c^{-\frac{T}{2}} - 1)L_h}{\rho} > 0 \) and \( \frac{\rho - L_f - 2(c^{-\frac{T}{2}} - 1)L_h}{\rho^2} > 0 \). By Lemma 2

\[
\sum_{i \in A_k} \|\nabla_{\omega} \tilde{\mathcal{L}}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})\|^2 \leq \left[ \frac{6\left( (c^{-\frac{T}{2}} - 1)^{-2} L_h^2 + \rho^2 \right)}{\rho - L_f - 2(c^{-\frac{T}{2}} - 1)L_h} + \frac{12\rho^3}{\rho^2 - L_\bar{g}^2 - 2L_\bar{g}^2} \right][\tilde{\mathcal{L}}_\rho(\omega^k, \theta^k, v^k) - \tilde{\mathcal{L}}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1})].
\]
Taking the expectation of random variable $\mathcal{A}_k$ conditioned on $\mathcal{F}_{k-1}$, we then have
\[
\mathbb{E}\left[ \sum_{i \in \mathcal{A}_k} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 | \mathcal{F}_{k-1} \right]
\leq \frac{6\epsilon_k}{} \left[ \frac{(c - \frac{2}{k}) - 1)^{-2} L_h^2 + \rho^2}{\rho - L_f - 2(c - \frac{2}{k}) - 1)L_h} + \frac{12\rho^3}{\rho^2 - L_\theta \rho - 2L_\theta^2} \right] [\mathbb{E}[\hat{L}_\rho(\omega^k, \theta^k, v^k) - \mathbb{E}[\hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1}) | \mathcal{F}_{k-1}]].
\] (C.19)

Meanwhile,
\[
\mathbb{E}\left[ \sum_{i \in \mathcal{A}_k} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 | \mathcal{F}_{k-1} \right] = \sum_{\mathcal{A}} P(\mathcal{A}_k = \mathcal{A}) \sum_{i \in \mathcal{A}} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2
\]
\[
= \sum_{i=1}^m p_i \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \geq \tilde{\rho} \sum_{i=1}^m \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2.
\]

Inserting this inequality into (C.19) and taking the total expectation $\mathcal{F}_k$, we obtain
\[
\mathbb{E}\left[ \sum_{i \in \mathcal{A}_k} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \right] \leq \frac{1}{\tilde{\rho}} \left[ \frac{6\epsilon_k}{} \left[ \frac{(c - \frac{2}{k}) - 1)^{-2} L_h^2 + \rho^2}{\rho - L_f - 2(c - \frac{2}{k}) - 1)L_h} + \frac{12\rho^3}{\rho^2 - L_\theta \rho - 2L_\theta^2} \right] [\mathbb{E}[\hat{L}_\rho(\omega^k, \theta^k, v^k) - \mathbb{E}[\hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1})]].
\]

Summing this inequality from $k = 0$ to $K - 1$ and multiplying the result by $\frac{1}{K}$, we get
\[
\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}\left[ \sum_{i \in \mathcal{A}_k} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \right] \leq \frac{1}{K\tilde{\rho}} \left[ \frac{6\epsilon_k}{} \left[ \frac{(c - \frac{2}{k}) - 1)^{-2} L_h^2 + \rho^2}{\rho - L_f - 2(c - \frac{2}{k}) - 1)L_h} + \frac{12\rho^3}{\rho^2 - L_\theta \rho - 2L_\theta^2} \right] [\mathbb{E}[\hat{L}_\rho(\omega^k, \theta^k, v^k) - \mathbb{E}[\hat{L}_\rho(\omega^{k+1}, \theta^{k+1}, v^{k+1})]].
\]

where $\hat{L}_\rho^0 = \hat{L}_\rho(\omega^0, \theta^0, v^0)$. By Lemma 3, we know $\mathbb{E}[\hat{L}_\rho K] \geq F^*$. $\theta^0 = v^0 = 0$, $\omega_1 = ... = \omega_m$, $\hat{L}_\rho^0 = \hat{F}(\omega^0)$. By Proposition 1, we know $\hat{F}(\omega^0) \leq F(\omega^0) + \frac{m\xi\lambda}{4}$. Finally, we have
\[
\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}\left[ \sum_{i \in \mathcal{A}_k} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \right] \leq \frac{C_1(F(\omega^0) - F^* + m\xi\lambda/4)}{K},
\]

where $C_1 = \frac{1}{K\tilde{\rho}} \left[ \frac{6\epsilon_k}{} \left[ \frac{(c - \frac{2}{k}) - 1)^{-2} L_h^2 + \rho^2}{\rho - L_f - 2(c - \frac{2}{k}) - 1)L_h} + \frac{12\rho^3}{\rho^2 - L_\theta \rho - 2L_\theta^2} \right].$

(b) First,
\[
\sum_{i=1}^m \sum_{j=1}^m \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 = \frac{1}{4m^2} \sum_{i=1}^m \sum_{j=1}^m \| \nabla_{\theta_i} (\theta^{k+1})_i - v^{k+1}_i \|^2 = 0.
\]

Taking the total expectation $\mathcal{F}_k$, we obtain
\[
\mathbb{E}\left[ \sum_{i \in \mathcal{A}_k} \| \nabla_{\omega_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \right] = 0.
\]

(c) By the update of $v^{k+1}_i$ (A.3), we have
\[
\nabla_{v^{k+1}_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) = \frac{1}{2m} (A_{ij} \omega^{k+1} - \theta^{k+1}) + \frac{1}{2m\rho} (v^{k+1}_j - v^{k}_i).
\]

Then, by (C.16), we obtain
\[
\sum_{i=1}^m \sum_{j=1}^m \| \nabla_{v^{k+1}_i} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \leq \frac{L_g^2}{4m^2 \rho^2} \sum_{i=1}^m \sum_{j=1}^m \| \theta^{k+1} - \theta^{k} \|^2.
\]

38
By Lemma 2
\[ \sum_{i=1}^{m} \sum_{j=1}^{m} \| \nabla_{v_{ij}} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2 \leq \frac{L^2_0}{m \rho^2 - L^2_0 - 2L^2_0} \| \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \| \| \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \| \].

Now, taking the total expectation \( \mathcal{F}_k \),
\[ \mathbb{E}[\| \nabla_{v} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2] \leq \frac{L^2_0}{m \rho^2 - L^2_0 - 2L^2_0} \| \mathbb{E}[\hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})] \| \| \mathbb{E}[\hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})] \| ] \].
(C.20)

Summing this inequality from \( k = 0 \) to \( K \) and multiplying the result by \( \frac{1}{K} \),
\[ \frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\| \nabla_{v} \hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1}) \|^2] \leq \frac{L^2_0}{m \rho^2 - L^2_0 - 2L^2_0} \| \mathbb{E}[\hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})] \| \| \mathbb{E}[\hat{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})] \| ] \]
where \( C_2 = \frac{m \rho^2 - L^2_0 - 2L^2_0}{K} \).

For the smoothed SCAD penalty, define \( \hat{G}(\omega, \theta, v) = [\| \theta - \text{prox}_{\mathcal{L}_0}(\theta) \| \] By the definition of \( \hat{L}_0 \), we have
\[ \text{prox}_{\mathcal{L}_0}(\theta_{ij}) = \text{arg min}_{x} [\hat{g}(x) - x < v_{ij}, x > + \frac{1}{2} \| x - \theta_{ij} \|^2] \]
(C.21)
\[ = \text{arg min}_{x} [\hat{g}(x) + \frac{1}{2} \| x - (\theta_{ij} + v_{ij}) \|^2] = \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}). \]

Similarly, we have \( \text{prox}_{\mathcal{L}_0}(\theta_{ij}) = \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) \). Then, the difference between \( \hat{g}(\omega, \theta, v) \) and \( g(\omega, \theta, v) \) is
\[ \| \hat{G}(\omega, \theta, v) - G(\omega, \theta, v) \|^2 = \sum_{i=1}^{m} \sum_{j=1}^{m} \| \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) - \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) \|^2. \]
(C.22)

Because
\[ \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) = \text{arg min}_{x} [\tilde{g}(x) + \frac{1}{2} \| x - (\theta_{ij} + v_{ij}) \|^2] = \begin{cases} \xi \frac{\lambda}{\lambda + \xi} (\theta_{ij} + v_{ij}), & \| \theta_{ij} + v_{ij} \| \leq \lambda + \xi, \\ (1 - \frac{\lambda}{\| v_{ij} \|}) (\theta_{ij} + v_{ij}), & \lambda + \xi < \| \theta_{ij} + v_{ij} \| \leq \lambda + \lambda, \\ \max\{0, 1 - \frac{\lambda}{\| v_{ij} \|}\} (\theta_{ij} + v_{ij}), & \lambda + \lambda < \| \theta_{ij} + v_{ij} \| \leq a\lambda, \\ \theta_{ij} + v_{ij}, & \| \theta_{ij} + v_{ij} \| > a\lambda, \end{cases} \]
(C.23)

and
\[ \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) = \text{arg min}_{x} [\tilde{g}(x) + \frac{1}{2} \| x - (\theta_{ij} + v_{ij}) \|^2] = \begin{cases} (1 - \frac{\lambda}{\| v_{ij} \|}) (\theta_{ij} + v_{ij}), & \| \theta_{ij} + v_{ij} \| \leq \lambda + \lambda, \\ \frac{\lambda}{\max\{0, 1 - \frac{\lambda}{\| v_{ij} \|}\}} (\theta_{ij} + v_{ij}), & \lambda + \lambda < \| \theta_{ij} + v_{ij} \| \leq a\lambda, \\ \theta_{ij} + v_{ij}, & \| \theta_{ij} + v_{ij} \| > a\lambda, \end{cases} \]
(C.24)
we have
\[ \| \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) - \text{prox}_{\tilde{g}}(\theta_{ij} + v_{ij}) \| = \begin{cases} \frac{\xi \lambda}{\lambda + \xi} (\theta_{ij} + v_{ij}), & \| \theta_{ij} + v_{ij} \| \leq \lambda + \xi, \\ 0, & \text{else}. \end{cases} \]
(C.25)

Consequently, we obtain
\[ \| \hat{G}(\omega, \theta, v) - G(\omega, \theta, v) \|^2 \leq \frac{m^2 \lambda^2 \xi^2}{(\lambda + \xi)^2}. \]
(C.26)
Since \( \mathbb{E}[\nabla \tilde{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})] = 0 \), which is equivalent to \( \mathbb{E}[\tilde{G}(\omega^{k+1}, \theta^{k+1}, v^{k+1})] = 0 \), together with Proposition 1, we obtain

\[
\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\|\nabla \omega \mathcal{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})\|^2] \leq \frac{C_1(F(\omega^0) - F^* + m\xi/4)}{K},
\]

\[
\mathbb{E}[\|\tilde{G}(\omega^{k+1}, \theta^{k+1}, v^{k+1})\|^2] \leq \frac{m^2\lambda^2\xi^2}{(\lambda + \xi)^2},
\]

\[
\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[\|\nabla \omega \mathcal{L}_0(\omega^{k+1}, \theta^{k+1}, v^{k+1})\|^2] \leq \frac{C_2(F(\omega^0) - F^* + m\xi/4)}{K}.
\]

\[\square\]

D Statistical convergence rate under a linear model

D.1 Notations and preparation

We provide the prerequisite notations, definitions, and conditions in the following. Note that some of them have been provided in the main text, but we also provide them here for completeness.

Denote the true values of parameters as \( \omega^{true} = (\omega_1^{true}, \ldots, \omega_m^{true})^\top \). Let \( \alpha^{true} = (\alpha_1^{true}, \ldots, \alpha_L^{true})^\top \) be the distinct values of \( \omega^{true} \), and then \( \omega_i^{true} = \alpha_i^{true} \) for \( i \in G_l \). Define the minimal differences of the common model parameters between two clusters as

\[
b = \min_{i \in G_l, j \in G_{l'} \neq l} \|\omega_i^{true} - \omega_j^{true}\| = \min_{i \neq i'} \|\alpha_i^{true} - \alpha_i^{true}\|.
\]

Let \( n = \sum_{i=1}^m n_i \), \( n_{Gi} = \sum_{i \in G_l} n_i \), \( n_{min} = \min_{l \in [L]} n_{Gi} \) and \( n_{max} = \max_{l \in [L]} n_{Gi} \). Write \( |G_{min}| = \min_{l \in [L]} |G_l| \), where \( | \cdot | \) is the cardinality of the set. For a matrix \( A \), denote its smallest / largest eigenvalue as \( \lambda_{\min}(A) / \lambda_{\max}(A) \), \( \ell_2 \)-norm as \( \|A\|_2 = \sqrt{\text{tr}(A^\top A)} \), and Frobenius norm as \( \|A\|_F = \sqrt{\text{tr}(A^\top A)} \).

Define \( \mathcal{M}_G = \{ \omega \in \mathbb{R}^{md} : \omega_i = \omega_j, \text{for any } i, j \in G_l, l \in [L] \} \). Let \( \tilde{Z} = \{ Z_{il} \} \) be a \( m \times L \) matrix with \( Z_{il} = 1 \) for \( i \in G_l \) and \( Z_{il} = 0 \) otherwise. Let \( \bar{Z} = \tilde{Z} \otimes I_d \). Therefore, each \( \omega \in \mathcal{M}_G \) can be rewritten as \( \omega = Z\alpha \), where \( \alpha = (\alpha_1^\top, \ldots, \alpha_L^\top)^\top \) and \( \alpha_l \) is the \( d \times 1 \) vector of the \( l \)-th cluster-specific parameters for \( l = 1, \ldots, L \). For \( i \in [m] \), write \( X_i = ((X_i^1)^\top, \ldots, (X_i^L)^\top)^\top \), \( y_i = (y_{i1}^1, \ldots, y_{in}^1)^\top \), and \( \tau_i = (\tau_{i1}^1, \ldots, \tau_{in}^1)^\top \). Define \( \tau_{Gi} = \{ \tau_i^l \} \) for \( l \in [L] \). Denote the \( j \)-th column of \( X_i \) as \( X_{ij} \), and then \( X_i = (X_{i1}, \ldots, X_{id}) \). Let \( X = \text{diag}(n_{1-1/2}^{-1}X_1, \ldots, n_{m-1/2}^{-1}X_m) \), \( y = (n_{1-1/2}^{-1}y_1^1, \ldots, n_{m-1/2}^{-1}y_m^1)^\top \), \( \tau = (n_{1-1/2}^{-1}\tau_1^1, \ldots, n_{m-1/2}^{-1}\tau_m^1)^\top \). Then, the linear model (9) and loss function \( f(\omega) \) can be written compactly as follows: \( y = X\omega^{true} + \tau \) and \( f(\omega) = \|y - X\omega\|^2 \).

Consider the oracle estimator \( \hat{\omega}^{or} \) for \( \omega \), under which the underlying clusters of devices \( G_1, \ldots, G_L \) are known. Specifically,

\[
\hat{\omega}^{or} = \arg\min_{\omega \in \mathcal{M}_G} \|y - X\omega\|^2. \tag{D.1}
\]

Correspondingly, the oracle estimator for the common parameters \( \alpha \) is

\[
\hat{\alpha}^{or} = \arg\min_{\alpha \in \mathbb{R}^{ld}} \|y - XZ\alpha\|^2.
\]

For simplicity, we denote the scaled SCAD penalty by \( \rho(t) = \lambda^{-1}P_\lambda(t) \). Note that \( \rho(t) \) is nondecreasing and concave in \( t \) for \( t \in [0, \infty) \) and \( \rho(0) = 0 \). When \( |t| > \alpha\lambda \), \( \rho(t) \) is a constant. \( \rho'(t) \) exists and is continuous except for a finite number of \( t \) and \( \rho'(0+) = 1 \).
Let $P(\omega, \lambda) = \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{m} \rho(\|\omega_i - \omega_j\|)$. Then, we can rewrite the objective function $F(\omega)$ as

$$F(\omega) = f(\omega) + P(\omega, \lambda).$$

(D.2)

The following conditions are assumed:

1. There exists a constant $C > 0$ such that $\lambda_{\min}(X_i^\top X_i) \geq Cn_i$ for $i \in [m]$.
2. $\|X_{ij}\| = \sqrt{n_i}$ for $i \in [m], j \in [d]$.
3. For $l \in [L]$, the noise vector $\tau_{G_l}$ has sub-Gaussian tails such that $P(|\langle c, \tau_{G_l} \rangle| > \|c\|x) \leq 2e^{-c_1 x^2}$ for any vector $c \in \mathbb{R}^{|G_l|}$ and $x > 0$, where $0 < c_1 < \infty$ is a constant.

These three conditions are common assumptions in penalized regression in high-dimensional settings. Under these mild assumptions, we can establish the statistical convergence rate.

D.2 Proof of Theorem 3

The proof of Theorem 3 is equivalent to establishing the following two results.

**Result 1:** Suppose Conditions (1)-(3) hold. Then, we have

$$P(\sup_{i \in [m]} \|\hat{\omega}_i^\text{or} - \omega_i^{\text{true}}\| \leq \Lambda_n) \geq 1 - 2Ldn^{-p_{\min}n^{-1}},$$

where $\Lambda_n = C^{-1} |G_{\min}|^{-1} c_1^{-1/2} \sqrt{2n_{\min} Ld \log n}$.

**Result 2:** Suppose $b > a\lambda$, $\lambda \gg m |G_{\min}|^{-1} \sqrt{n_{\min} Ld^2 \log n} + m \sqrt{n_{\min} d \log n}$, and the conditions in Result 1 hold. Then, there exists a local minimizer $\omega^*$ of $F(\omega)$ such that

$$P(\omega^* = \hat{\omega}_i^\text{or}) \geq 1 - 2Ldn^{-p_{\min}n^{-1}} - 2dn^{-1}.$$

**Proof of Result 1.** Recall that

$$\hat{\alpha}_i^\text{or} = \arg \min_{\alpha \in \mathbb{R}^L} \|y - XZ\alpha\|^2 = (Z^\top X^\top XZ)^{-1}(Z^\top X^\top X) y.$$

Then, using the relation $y = X\omega + \tau$, we have

$$\hat{\alpha}_i^\text{or} - \alpha_i^{\text{true}} = (Z^\top X^\top XZ)^{-1}Z^\top X^\top \tau,$$

and

$$\|\hat{\alpha}_i^\text{or} - \alpha_i^{\text{true}}\| \leq \|(Z^\top X^\top XZ)^{-1}\| \|Z^\top X^\top \tau\|. \quad \text{(D.3)}$$

By Condition (1),

$$\|(Z^\top X^\top XZ)^{-1}\| = \lambda_{\min}^{-1}(Z^\top X^\top XZ) \leq C^{-1} |G_{\min}|^{-1}. \quad \text{(D.4)}$$

Moreover,

$$\|Z^\top X^\top \tau\|_\infty = \sup_{j \in [d], l \in [L]} \sum_{i=1}^{m} X_{ij}^\top \tau_l (i \in G_l). \quad \text{(D.5)}$$
Write \( X_{G_i,j} = \{ X_{ij}, i \in G_i \} \) for \( l \in [L] \). By \( (D.5) \), we have

\[
P(\| Z^T X^T \|_\infty > c_1^{-\frac{1}{2}} \sqrt{2n_{min}^{-1} \log n}) \overset{(D.5)}{=} \sum_{j=1}^d \sum_{l=1}^L P(\| n_{G_i,j}^{-1} X_{ij}^T \| > c_1^{-\frac{1}{2}} \sqrt{2n_{min}^{-1} \log n}) \leq \sum_{j=1}^d \sum_{l=1}^L P(\| n_{G_i,j}^{-1} X_{ij} \| > c_1^{-\frac{1}{2}} \sqrt{2n_{min}^{-1} \log n}) \leq \sum_{j=1}^d \sum_{l=1}^L P(\| X_{G_i,j} \| > \| X \| \sqrt{n_{min}^{-1/2} c_1^{-1/2} \sqrt{\log n}}) \leq 2Ldc(-n_{min}^{-1/2} n_{max}^{-1} \log n) = 2Ld n_{min}^{-n_{max}^{-1}} n_{max}^{-1},
\]

where the third inequality follows from \( \| X_{G_i,j} \| \leq \sum_{i \in G_i} \| X \| \leq \sqrt{2n_{max}} \).

Note that \( \| Z^T X^T \| \leq \sqrt{Ld} \| Z^T X^T \|_\infty \). Then,

\[
P(\| Z^T X^T \| > C_0 \sqrt{2Ld n_{min}^{-1} \log n}) \leq P(\| Z^T X^T \|_\infty > c_1^{-\frac{1}{2}} \sqrt{2n_{min}^{-1} \log n}) \leq 2Ld n_{min}^{-n_{max}^{-1}} n_{max}^{-1}.
\]

By \( (D.3) \), \( (D.4) \), and \( (D.6) \), with probability at least \( 1 - 2Ld n_{min}^{-n_{max}^{-1}} n_{max}^{-1} \), we have

\[
\sup_{i \in [m]} \| \hat{\omega}_i \| - \| \omega_i \| \leq \sup_{l \in [L]} \| \hat{\alpha}_l \| - \| \alpha_l \| \leq \| \hat{\alpha} \| - \| \alpha \| \leq \Lambda_n,
\]

where \( \Lambda_n = C^{-1} |G_{min}|^{-1} c_1^{-1/2} \sqrt{2Ld n_{min}^{-1} \log n} \). This completes the proof.

**Proof of Result 2** Let \( T : M \rightarrow \mathbb{R}^{L \times d} \) be the mapping such that \( T(\omega) \) is the \( L \times d \) matrix consisting of \( L \) vectors with dimension \( d \), and its \( l \)-th vector component equals the common value of \( \omega_i \) for \( i \in G_l \). Let \( T^* : \mathbb{R}^{md} \rightarrow \mathbb{R}^{L \times d} \) be the mapping such that \( T^*(\omega) = \{ \sum_{i \in G_l} \frac{n_{G_i}}{n_{G_l}} \omega_i^T, l \in [L] \}^T \), where \( n_{G_l} = \sum_{i \in G_l} n_i \). Obviously, when \( \omega \in M_G \), \( T(\omega) = T^*(\omega) \). For any \( \omega \in \mathbb{R}^{md} \), let \( T^*(\omega) = \alpha = (\alpha_1^T, \ldots, \alpha_L^T) \) and \( \omega_{inv} = T^{-1}(T^*(\omega)) = T^{-1}(\alpha) \).

Consider the neighborhood of \( \omega_{true} \):

\[
\Theta = \{ \omega \in \mathbb{R}^{md} : \sup_{i \in [m]} \| \omega_i - \omega_i^{true} \| \leq \Lambda_n \}.
\]

By Result 1, there exists an event \( E_1 \) such that \( \sup_{i \in [m]} \| \hat{\omega}_i - \omega_i^{true} \| \leq \Lambda_n \) and \( P(E_1^C) \leq 2Ld n_{min}^{-n_{max}^{-1}} n_{max}^{-1} \).

Thus, \( \hat{\omega} \in \Theta \) in \( E_1 \). We show that \( \hat{\omega} \) is a strict local minimizer of the objective function \( F(\omega) \) with probability at least \( 1 - 2Ld n_{min}^{-n_{max}^{-1}} - 2d n^{-1} \) through the following two steps:

(i) In the event \( E_1 \), \( F(\omega_{inv}) > F(\hat{\omega}) \) for any \( \omega \in \Theta \) and \( \omega_{inv} \neq \hat{\omega} \).

(ii) There is an event \( E_2 \) such that \( P(E_2^C) < 2d n^{-1} \). In \( E_1 \cap E_2 \), \( F(\omega) \geq F(\omega_{inv}) \) for any \( \omega \in \Theta \) for sufficiently large \( n \).

With the results in (i) and (ii), for any \( \omega \in \Theta \) and \( \omega \neq \hat{\omega} \) in \( E_1 \cap E_2 \), we have \( F(\omega) > F(\hat{\omega}) \), so \( \hat{\omega} \) is a strict local minimizer of \( F(\omega) \) over the event \( E_1 \cap E_2 \) with \( P(E_1 \cap E_2) \geq 1 - 2Ld n_{min}^{-n_{max}^{-1}} - 2d n^{-1} \) for sufficiently large \( n \).

We first prove the results in (i). Because \( \hat{\omega} \) is the unique global minimizer of \( f(\omega) \) for \( \omega \in M_G \), \( f(\omega_{inv}) > f(\hat{\omega}) \), so we need to consider \( P(\omega_{inv}, \lambda) \). Because

\[
\| \alpha_l - \alpha_l' \| \geq \| \alpha_l^{true} - \alpha_l'^{true} \| - \| \alpha_l - \alpha_l^{true} \| - \| \alpha_l^{true} - \alpha_l'^{true} \| - 2 \sup_{l \in [L]} \| \alpha_l - \alpha_l^{true} \| \]

42
and
\[ \sup_{l \in [L]} \| \omega_l - \omega_l^{\text{true}} \|^2 = \sup_{l \in [L]} \| G_l \|^{-1} \sum_{i \in G_l} (\omega_i - \omega_i^{\text{true}})^2 \]
\[ = \sup_{l \in [L]} \| G_l \|^{-1} \sum_{i \in G_l} (\omega_i - \omega_i^{\text{true}})^2 \]
\[ = \sup_{l \in [L]} \| G_l \|^{-2} \sum_{i \in G_l} (\omega_i - \omega_i^{\text{true}})^2 \]
\[ \leq \sup_{l \in [L]} \| G_l \|^{-1} \sum_{i \in G_l} (\omega_i - \omega_i^{\text{true}})^2 \]
\[ \leq \sup_{l \in [L]} \| \omega_l - \omega_l^{\text{true}} \|^2 \leq \Lambda_n^2, \]
for all \( l, l' \in [L] \), we have
\[ \| \omega_l - \omega_{l'} \| \geq b - 2\Lambda_n \geq a\lambda, \]
where the last inequality follows from the assumption that \( b > a\lambda \gg \Lambda_n \). Consequently, \( \rho(\| \alpha_l - \alpha_{l'} \|) \) is a constant by the property of the SCAD function. Also note that
\[ P(\omega^{\text{inv}}, \lambda) = \frac{\lambda}{2m} \sum_{i=1}^{m} \rho(\| \omega_i^{\text{inv}} - \omega_j^{\text{inv}} \|) = \frac{\lambda}{2m} \sum_{i \neq j} |G_i| |G_j| \rho(\| \alpha_l - \alpha_{l'} \|). \]
So \( P(\omega^{\text{inv}}, \lambda) \) is a constant. Therefore, \( f(\omega^{\text{inv}}) + P(\omega^{\text{inv}}, \lambda) > f(\hat{\omega}^{\text{OR}}) + P(\hat{\omega}^{\text{OR}}, \lambda) \), i.e., \( F(\omega^{\text{inv}}) > F(\hat{\omega}^{\text{OR}}) \) for all \( \omega^{\text{inv}} \neq \hat{\omega}^{\text{OR}} \), and the result in (i) is proved.

Next we prove the result in (ii). For \( \omega \in \Theta \), by Taylor’s expansion, we have
\[ F(\omega) - F(\hat{\omega}^{\text{OR}}) = \Gamma_1 + \Gamma_2, \]
where \( \Gamma_1 = -(y - X\hat{\omega}, X(\omega - \omega^{\text{inv}})) \), \( \Gamma_2 = \frac{\partial P(\omega, \lambda)}{\partial \omega} |_{\omega=\hat{\omega}} (\omega - \omega^{\text{inv}}) \), and \( \hat{\omega} = c\omega + (1-c)\omega^{\text{inv}} \) for some constant \( c \in (0,1) \).

First, we consider \( \Gamma_2 \). We have
\[ \Gamma_2 = \frac{\partial P(\omega, \lambda)}{\partial \omega} |_{\omega=\hat{\omega}} (\omega - \omega^{\text{inv}}) = \frac{\lambda}{2m} \sum_{i \neq j} \rho'(\| \hat{\omega}_i - \hat{\omega}_j \|) \frac{\hat{\omega}_i - \hat{\omega}_j}{\| \hat{\omega}_i - \hat{\omega}_j \|} \left[ (\omega_i - \omega_i^{\text{inv}}) - (\omega_j - \omega_j^{\text{inv}}) \right]. \]

When \( i, j \in G_l \), \( \omega_i^{\text{inv}} = \omega_j^{\text{inv}} \) and \( \hat{\omega}_i - \hat{\omega}_j = c(\omega_i - \omega_j) \). Thus
\[ \Gamma_2 = \frac{\lambda}{2m} \sum_{l=1}^{L} \sum_{\{i,j \in G_l, i \neq j\}} \rho'(\| \hat{\omega}_i - \hat{\omega}_j \|) \frac{\hat{\omega}_i - \hat{\omega}_j}{\| \hat{\omega}_i - \hat{\omega}_j \|} \left[ (\omega_i - \omega_i^{\text{inv}}) - (\omega_j - \omega_j^{\text{inv}}) \right]. \]

Note that \( \sup_{i \in [m]} \| \omega_i^{\text{inv}} - \omega_i^{\text{true}} \| = \| \alpha_l - \alpha_l^{\text{true}} \| \leq \Lambda_n \), which follows from (D.7). Then,
\[ \sup_{i \in [m]} \| \hat{\omega}_i - \omega_i^{\text{true}} \| \leq c \sup_{i \in [m]} \| \omega_i - \omega_i^{\text{true}} \| + (1-c) \sup_{i \in [m]} \| \omega_i^{\text{inv}} - \omega_i^{\text{true}} \|. \]

Hence, for \( i \in G_l, j \in G_{l'}, l \neq l' \),
\[ \| \hat{\omega}_i - \omega_j \| \geq \min_{i \in [m]} \| \omega_i^{\text{true}} - \omega_j^{\text{true}} \| - 2\sup_k \| \hat{\omega}_k - \omega_k^{\text{true}} \| \]
\[ = \| \alpha_l^{\text{true}} - \alpha_{l'}^{\text{true}} \| - 2\sup_k \| \hat{\omega}_k - \omega_k^{\text{true}} \| \geq b - 2\Lambda_n > a\lambda, \]
and thus, $\rho'(||\tilde{\omega}_i - \tilde{\omega}_j||) = 0$. Consequently,

$$
\Gamma_2 = \frac{\lambda}{2m} \sum_{l=1}^{L} \sum_{i,j \in G, i \neq j} \rho'(||\tilde{\omega}_i - \tilde{\omega}_j||)\frac{(\tilde{\omega}_i - \tilde{\omega}_j)\top}{||\tilde{\omega}_i - \tilde{\omega}_j||} \left[ (\omega_i^{\text{inv}} - (\omega_j^{\text{inv}}) \right]
$$

$$
\quad = \frac{\lambda}{2m} \sum_{l=1}^{L} \sum_{i,j \in G, i \neq j} \rho'(||\tilde{\omega}_i - \tilde{\omega}_j||)||\omega_i - \omega_j||,
$$

where the last equality follows from $\tilde{\omega}_i - \tilde{\omega}_j = c(\omega_i - \omega_j)$. Furthermore, by the same reasoning as (D.7), we have

$$
\sup_{i \in [m]} ||\omega_i^{\text{inv}} - \omega_i^{\text{true}}|| = \sup_{l \in [L]} ||\alpha_l - \alpha_l^{\text{true}}|| \leq \sup_{i \in [m]} ||\omega_i - \tilde{\omega}_i^{\text{true}}||.
$$

Then, it can be shown that

$$
\sup_{i \in [m]} ||\tilde{\omega}_i - \tilde{\omega}_j|| \leq 2 \sup_{i \in [m]} ||\tilde{\omega}_i - \omega_i^{\text{true}}|| \leq 2 \sup_{i \in [m]} ||\omega_i - \omega_i^{\text{true}}||
$$

$$
\leq 2(\sup_{i \in [m]} ||\omega_i - \tilde{\omega}_i^{\text{true}}|| + \sup_{i \in [m]} ||\omega_i^{\text{true}} - \omega_i^{\text{true}}||) \leq 2 \sup_{i \in [m]} ||\omega_i - \omega_i^{\text{true}}|| \leq 4 \sup_{i \in [m]} ||\omega_i - \omega_i^{\text{true}}|| \leq 4\Lambda_n.
$$

Hence, $\rho'(||\tilde{\omega}_i - \tilde{\omega}_j||) \geq \rho'(4\Lambda_n)$ due to the concavity of $\rho(\cdot)$. As a result,

$$
\Gamma_2 \geq \frac{\lambda}{2m} \sum_{l=1}^{L} \sum_{i,j \in G, i \neq j} \rho'(4\Lambda_n)||\omega_i - \omega_j|| \geq \frac{\lambda}{m} \sum_{l=1}^{L} \sum_{i,j \in G, i \neq j} \rho'(4\Lambda_n)||\omega_i - \omega_j||. \quad (D.10)
$$

Next, we consider $\Gamma_1$. Let $\bar{v} = (\bar{v}_1^\top, \cdots, \bar{v}_m^\top)^\top = ((\bar{y} - \bar{X}\tilde{\omega})^\top \bar{X})^\top$. Then, we have

$$
\Gamma_1 = -\bar{v}^\top(\omega - \omega^{\text{inv}}) = -\sum_{i=1}^{m} \bar{v}_i^\top(\omega_i - \omega_i^{\text{inv}})
$$

$$
\quad = -\sum_{l=1}^{L} \sum_{i,j \in G, i \neq j} \frac{(n_j \bar{v}_i - n_i \bar{v}_j)^\top(\omega_i - \omega_j)}{n_{G_i}} \quad \text{(D.11)}
$$

$$
\quad \geq -\sum_{l=1}^{L} \sum_{i,j \in G, i \neq j} \frac{(n_j ||\bar{v}_i|| + n_i ||\bar{v}_j||) ||\omega_i - \omega_j||}{n_{G_i}},
$$

where the second equality follows from $\omega^{\text{inv}} = T^{-1}(T^*(\omega))$ and $\sum_{i=1}^{m} \bar{v}_i^\top \omega_i^{\text{inv}} = \sum_{i=1}^{L} \sum_{i,j \in G_i} \frac{n_j}{n_{G_i}} \bar{v}_i^\top \omega_j$. Moreover,

$$
\bar{v}_i = n_i^{-1}(\bar{X}_i^\top \bar{y}_i - \bar{X}_i^\top \bar{X}_i \bar{w}_i) = n_i^{-1}(\bar{X}_i^\top \bar{X}_i (\omega_i^{\text{true}} - \tilde{\omega}_i) + \bar{X}_i^\top \tau_i),
$$

and then

$$
||\bar{v}_i|| \leq n_i^{-1}(||\bar{X}_i^\top \bar{X}_i||_F ||\omega_i^{\text{true}} - \tilde{\omega}_i|| + ||\bar{X}_i^\top \tau_i||) \leq n_i^{-1}(||\bar{X}_i^\top \bar{X}_i||_F \Lambda_n + ||\bar{X}_i^\top \tau_i||). \quad (D.12)
$$

By Condition (2),

$$
||\bar{X}_i^\top \bar{X}_i||_F = \sqrt{\sum_{l=1}^{d} \sum_{k=1}^{d} (\bar{X}_{ik})^2} \leq \sum_{l=1}^{d} \sum_{k=1}^{d} (\bar{X}_{ik})^2 \leq \sum_{k=1}^{d} d \leq d n_i.
$$
By Condition (3),

\[
P(\|X_i^\top \tau_i\|_\infty > \sqrt{c_1^{-1} n_i \log n}) \leq \sum_{j=1}^d P(\|X_i^\top \tau_i\| > \sqrt{c_1^{-1} n_i \log n}) \leq \sum_{j=1}^d P(\|X_i^\top \tau_i\| > \sqrt{c_1^{-1} n_i \log n}) \leq 2dn^{-1}.
\]

(D.13)

Note that \(\|X_i^\top \tau_i\| \leq \sqrt{d}\|X_i^\top \tau_i\|_\infty\), so \(P(\|X_i^\top \tau_i\| > \sqrt{c_1^{-1} n_i \log n}) \leq 2dn^{-1}\). Thus, there is an event \(E_2\) such that \(P(E_2^c) \leq 2dn^{-1}\), and over the event \(E_2\),

\[
\|\tilde{v}_i\| \leq \sqrt{d}(\sqrt{d\Lambda_n} + \sqrt{c_1^{-1} n_i^{-1} \log n}) = \sqrt{d}(C^{-1}|G_{\min}|^{-1} c_1^{-1/2} \sqrt{2n_{\min}^{-1}} L \log n + \sqrt{c_1^{-1} n_i^{-1} \log n})
\]

\[
= \mathcal{O}(|G_{\min}|^{-1} \sqrt{n_{\min}^{-1}} L d^3 \log n + \sqrt{n_{\min}^{-1}} d \log n)
\]

(D.14)

Because \(n_j \leq n_{G_i}\) for \(j \in G_i\), we have

\[
n_{j} n_{G_i}^{-1} |G_{\min}|^{-1} \sqrt{n_{\min}^{-1} L d^3 \log n} + mn_{j} n_{G_i}^{-1} \sqrt{n_{\min}^{-1} d \log n} \leq |G_{\min}|^{-1} \sqrt{n_{\min}^{-1} L d^3 \log n} + m \sqrt{n_{\min}^{-1} d \log n}.
\]

By the assumption that \(\lambda \gg m|G_{\min}|^{-1} \sqrt{L d^3 n_{\min}^{-1} \log n} + m \sqrt{n_{\min}^{-1} d \log n}\), we have

\[
\frac{\lambda}{2m} \gg \frac{n_j \|\tilde{v}_i\| + n_i \|\tilde{v}_j\|}{n_{G_i}}
\]

for \(i, j \in G_i, i \neq j\), and \(l \in [L]\). Note that \(\rho'(4\Lambda_n) \to 1\) for the SCAD penalty. Therefore, for sufficiently large \(n\), we have

\[
F(\omega) - F(\omega^{inv}) = \Gamma_1 + \Gamma_2 \geq \sum_{l=1}^L \sum_{\{i,j \in G_i, i < j\}} \|\omega_i - \omega_j\| \left(\frac{\lambda \rho'(4\Lambda_n)}{m} - \frac{n_j \|\tilde{v}_i\| + n_i \|\tilde{v}_j\|}{n_{G_i}}\right) > 0.
\]

Therefore, in \(E_1 \cap E_2\), \(F(\omega) \geq F(\omega^{inv})\) for any \(\omega \in \Theta\). The result in (ii) is proved.

E  Simulation details and additional numerical experiments

In this section, we provide more details on the settings of numerical experiments and more extensive experimental results to demonstrate the performance of our method compared to others.

E.1 Datasets and models

**Synthetic.** Regarding cluster structures, we consider five scenarios: (S1) balanced with \(L = 4\), where each cluster has 25 devices; (S2) unbalanced with \(L = 4\), where the cluster sizes are 10, 40, 10 and 40; (S3) balanced with \(L = 2\), where each cluster has 50 devices; (S4) unstructured with \(L = 1\), where all \(m = 50\) devices have an identical optimal model and parameters; and (S5) personalized with \(L = m = 50\), where each device owns a local personalized model. We use the cross-entropy loss functions.

**Housing and Body fat (H&BF).** The Housing dataset [8,21] has 506 samples and 13 features. The response vector \(y\) is the median house price, and features include some factors that may affect the price of housing, such as per capita crime rate. The Body fat dataset contains 252 samples and
Table E.1: Experimental results on synthetic datasets under scenario S2.

| Methods | LOCAL | FedAvg | LG | Per-FedAvg | IFCA | CFL | PACFL | FPFC-ℓ | FPFC |
|---------|-------|--------|----|------------|------|-----|-------|--------|------|
| Acc     | 85.49%±0.03 | 40.63%±0.01 | 73.25%±0.05 | 65.67%±0.06 | 58.30%±0.18 | 89.31%±0.03 | 91.36%±0.02 | 84.91%±0.04 | 90.22%±0.03 |
| Num     | ×     | ×      | ×   | ×          | 2.00±0.82 | 36.00±6.48 | 8.00±5.10 | 9.67±4.19 | 11.67±8.81 |
| ARI     | ×     | ×      | ×   | ×          | 0.39±0.41 | 0.19±0.02 | 0.78±0.15 | 0.94±0.04 | 0.97±0.03 |

Table E.2: Experimental results on synthetic datasets under scenario S3.

| Methods | LOCAL | FedAvg | LG | Per-FedAvg | IFCA | CFL | PACFL | FPFC-ℓ | FPFC |
|---------|-------|--------|----|------------|------|-----|-------|--------|------|
| Acc     | 85.60%±0.01 | 50.74%±0.03 | 77.62%±0.08 | 67.86%±0.02 | 64.11%±0.18 | 90.52%±0.01 | 91.64%±0.01 | 87.12%±0.02 | 90.96%±0.01 |
| Num     | ×     | ×      | ×   | ×          | 1.33±0.47 | 9.67±2.05 | 2.00±0.00 | 2.00±0.00 | 2.00±0.00 |
| ARI     | ×     | ×      | ×   | ×          | 0.33±0.47 | 0.52±0.10 | 1.00±0.00 | 1.00±0.00 | 1.00±0.00 |

14 features. The response vector \( y \) is the percentage of body fat, and features include various body measurements. For consistency of the number of features, we randomly generate a vector from a normal distribution as the 14-th feature of the housing dataset.

**MNIST & FMNIST.** We use two public benchmark data sets, MNIST [31] and FMNIST (Fashion-MNIST) [60]. We followed the procedures described in [50] to create the practical non-IID data setting, which has a clear cluster structure. Let us take MNIST as an example. We first partitioned 60000 training images and 10000 testing images into \( m = 20 \) devices by a Dirichlet distribution, each of which belonging to one of \( L = 4 \) clusters. We then modify every devices’ data by randomly swapping two labels, depending on which cluster a device belongs to. Specifically, the first cluster consists of devices 1-5, where each devices’ data labeled as "0" could be relabeled as "8" and vice versa. The second cluster consists of devices 6-10, where each devices’ data labeled as "1" and "7" could be switched out similarly, and so on. The testing data are processed in the same way. We use a multilayer convolutional neural network with cross-entropy loss. To account for some common properties of the data from all devices, we adopt the weight sharing technique from multi-task learning [6], as used in IFCA [17]. Specifically, when we train the neural network models, we share the weights for the first three layers and run our method only on the last layer, producing a predicted label for each image.

**Machines and libraries.** All the methods are implemented in PyTorch [46] version 1.9.0 running on a public computing cloud with an Intel(R) Xeon Gold 5218 CPU and 192GB memory.

### E.2 Additional experiments and results

#### E.2.1 Synthetic data

Table E.1 reports the results for scenario S2, i.e., unbalanced cluster structure with \( L = 4 \). FPFC has favorable clustering performance. It can be observed that only FPFC can identify the general cluster structure and the overestimation of the number of clusters is due to the over-subdivision of two small clusters. The results of scenario S3 are presented in Table E.2. FPFC has the best or close to the best performance in prediction and clustering. Tables E.3 and E.4 list the results of scenarios S4 and S5. It can be seen that in the two worst scenarios, i.e., unstructured with \( L = 1 \) and personalized with \( L = m \), our FPFC still has competitive performance; in particular, it still can
Table E.4: Experimental results on synthetic datasets under scenario S5.

| Methods | LOCAL | FedAvg | LG | Per-FedAvg | IFCA | CFL | PACFL | FPFC-ℓ | FPFC |
|---------|-------|--------|----|------------|------|-----|-------|--------|------|
| Acc     | 81.83 ± 0.02 | 9.44 ± 0.01 | 63.41 ± 0.01 | 30.56 ± 0.07 | 60.17 ± 0.03 | 65.47 ± 0.02 | 81.16 ± 0.02 | 80.27 ± 0.02 | 80.27 ± 0.02 |
| Num     | x     | x      | x  | x          | 28.33 ± 2.05 | 37.67 ± 1.25 | 50.00 ± 0.00 | 50.00 ± 0.00 | 50.00 ± 0.00 |
| ARI     | x     | x      | x  | x          | 0.00 ± 0.00  | 0.00 ± 0.00  | 1.00 ± 0.00  | 1.00 ± 0.00  | 1.00 ± 0.00  |

Figure E.1: Distribution of testing accuracy and ARI over 50 random initializations on synthetic datasets.

Identify the cluster structure exactly, suggesting that FPFC is a safe choice in practice.

E.2.2 Sensitivity to Initial Values

Due to the nonconvexity of our objective function, the results of FPFC would rely on the initial values. As such, we randomly generate 50 initial values to test the initial value sensitivity. In Fig. E.1, we present the distributions of testing accuracy and ARI for a random replicate under the scenarios S1 and S3. As we can see, FPFC is very robust to the initial values, while IFCA, an iterative optimization algorithm that is also dependent on the initial values, displays large fluctuations in the quality of results.

Figure E.2: Robustness comparison of different methods on synthetic dataset.
E.2.3 Complete Results on Robustness

Fig. E.2 and Fig. E.3 show the robustness results on the synthetic and MNIST datasets, respectively, for different methods under the three Byzantine attacks. FPFC consistently exhibits higher robustness compared to other algorithms under various attack scenarios.

E.2.4 Effect of Regularization Parameter

In FPFC, the regularization parameter $\lambda$ controls the strength of fusion penalty. When $\lambda = 0$, the penalty is absent, and the original model will degenerate into the fully personalized model; When $\lambda \to \infty$, all devices will be shrunk into one cluster, and the original model will degenerate into the global model in standard FL. Fig. E.4 presents the testing RMSE/accuracy as a function of $\lambda$ on different datasets. Take synthetic data as an example, it can be observed that as $\lambda$ increases, the testing accuracy increases at first, peaks at approximately $\lambda = 0.6$, and then decreases, implying that a suitable $\lambda$ can improve prediction performance.
E.2.5 Heterogeneous training

Due to the different local storage and computational power, different devices may perform varied epochs of local updates. To simulate such heterogeneous training, we fix a global number of epochs $T$ and assign $T_i$ epochs (chosen uniformly at random in the range $[1, T]$) to each participating device. Fig. E.5 shows the testing RMSE/accuracy versus $T$. We observe that FPFC has a high tolerance to heterogeneous epochs as long as the number of epochs is not too small.

E.2.6 Effect of $\xi$

Due to the non-differentiability of the SCAD penalty at all points, we approximate the non-differentiable SCAD penalty by a continuously differentiable function, smoothed SCAD, and $\xi (\xi < \lambda)$ is a newly introduced parameter. Fig. E.6 shows the performance of different choices of $\xi$. We can see that the parameter $\xi$ will not affect the results as long as it is small enough.