

# Smoothing ADMM for Non-convex and Non-smooth Hierarchical Federated Learning

Reza Mirzaeifard, Stefan Werner

Department of Electronic Systems, Norwegian University of Science and Technology, Norway

E-mails: {reza.mirzaeifard, stefan.werner}@ntnu.no

**Abstract**—This paper presents a hierarchical federated learning (FL) framework that extends the alternating direction method of multipliers (ADMM) with smoothing techniques, tailored for non-convex and non-smooth objectives. Unlike traditional hierarchical FL methods, our approach supports asynchronous updates and multiple updates per iteration, enhancing adaptability to heterogeneous data and system settings. Additionally, we introduce a flexible mechanism to leverage diverse regularization functions at each layer, allowing customization to the specific prior information within each cluster and accommodating (possibly) non-smooth penalty objectives. Depending on the learning goal, the framework supports both consensus and personalization: the total variation norm can be used to enforce consensus across layers, while non-convex penalties such as minimax concave penalty (MCP) or smoothly clipped absolute deviation (SCAD) enable personalized learning. Experimental results demonstrate the superior convergence rates and accuracy of our method compared to conventional approaches, underscoring its robustness and versatility for a wide range of FL scenarios.

**Index Terms**—Federated learning, non-convex and non-smooth sparse penalties, smoothing techniques

## I. INTRODUCTION

The rapid expansion of cyber-physical systems within the internet of things (IoT) ecosystem has necessitated a shift from centralized data processing to federated frameworks. This transition is driven by the significant computational demands, bandwidth constraints, and privacy concerns associated with traditional centralized methods [1], [2]. Federated learning (FL) has emerged as a pivotal solution, enabling collaborative model training across distributed devices while keeping data localized, thereby addressing critical privacy challenges and communication overhead [3], [4]. Recent advancements have further enhanced FL's scalability through hierarchical architectures, where two layers of aggregation are performed: edge servers handle intra-set computations for individual client groups, while a cloud server manages inter-set coordination across clusters [4], [5]. This hierarchical approach optimizes resource utilization and facilitates parallel computations, making it indispensable for large-scale distributed systems.

Despite these benefits, hierarchical FL environments face substantial heterogeneity among participating clients, necessitating flexible update strategies. First, data heterogeneity arises from distribution skew, label skew, feature skew, and variations in data quality (e.g., quantity skew and noise levels) [6], [7]. Second, model heterogeneity occurs when clients

must tailor model architectures or optimizers to accommodate various hardware or domain requirements [8], [9]. Third, task heterogeneity emerges when clients solve distinct yet related problems [10], [11]. Fourth, communication heterogeneity stems from inconsistent network conditions and bandwidth constraints [12], [13]. Finally, device heterogeneity introduces disparities in computational power and storage capacities [14], [15]. These factors underscore the need for a flexible regularization strategy, support of asynchronous updates (to handle communication delays), and multiple updates per iteration (to leverage resource-rich clients) in hierarchical FL.

In addition to these challenges, non-smooth and non-convex formulations have gained increasing importance in FL due to real-world complexities such as noise, outliers, and the need for personalization [16]–[19]. Non-smooth objectives offer robustness against outliers and heavy-tailed distributions, improving the reliability of the learned models. Non-smooth regularization can induce sparsity (e.g., using an  $l_1$  penalty) or similarity constraints (e.g., total variation norms) to promote interpretable structures [17], [20]. Meanwhile, non-convex regularizers like smoothly clipped absolute deviation (SCAD) [21] and minimax concave penalty (MCP) [22] mitigate the bias of convex penalties and enable personalized learning by capturing intricate data relationships across heterogeneous clients [16]. However, state-of-the-art methods often struggle with non-convexity and non-smoothness, especially when asynchronous updates, multiple local iterations, and scalability are required [23], [24]. Consequently, there is a need for a new optimization framework that efficiently manages these complexities to produce robust, interpretable, and personalized solutions for large-scale hierarchical FL environments.

This work introduces the hierarchical federated smoothing ADMM (HFSAD), a framework tailored for hierarchical FL that addresses non-convex, non-smooth optimization challenges while accommodating heterogeneous network and computational conditions. HFSAD leverages smoothing techniques selectively on consensus or personalization penalties (e.g., total variation norms), making them tractable within an ADMM-based scheme. At the same time, it retains non-smooth penalties (e.g.,  $l_1$ , MCP, SCAD) for sparsity and inductive knowledge about parameters, thereby promoting interpretability and robust regularization. This dual approach balances global alignment and local customization, while enabling principled handling of outliers or heavy-tailed data distributions. HFSAD supports asynchronous updates and

This work was supported by the Research Council of Norway.

multiple local iterations, accommodating heterogeneous network and computational conditions. By uniting smoothing-driven linearization with ADMM's decomposition capabilities, our method maintains scalability across distributed clusters, ensuring robustness to outliers, interpretability via sparse solutions, and convergence guarantees. Experimental results demonstrate that HFSAD surpasses existing approaches in accuracy, speed, and flexibility, establishing a new standard for distributed data analysis under diverse system constraints.

**Mathematical Notations:** Scalars are denoted by lowercase letters, column vectors by bold lowercase letters, and matrices by bold uppercase letters. The transpose of a matrix  $\mathbf{A}$  is indicated by  $\mathbf{A}^\top$ . The  $j$ th column of  $\mathbf{A}$  is denoted by  $\mathbf{a}_j$ , and the element in the  $i$ th row and  $j$ th column of  $\mathbf{A}$  is denoted by  $a_{ij}$ . The proximal operator of a function  $f(\cdot)$ , scaled by a parameter  $\gamma$ , is defined as  $\text{Prox}_f(w; \gamma) = \arg \min_x \left\{ f(x) + \frac{1}{2\gamma} \|x - w\|_2^2 \right\}$ . Finally,  $\partial f(u)$  denotes the sub-gradient of the function  $f(\cdot)$  at point  $u$ .

## II. PROBLEM FORMULATION

Assume that the system consists of one cloud server and  $L$  edge servers, where each edge server, indexed by  $l$  with  $l = 1, 2, \dots, L$ , is associated with a unique set of clients  $V_l$ . Each set  $V_l$  contains  $N_l$  clients, and every client  $j$  in  $V_l$  possesses its own local dataset  $D_{lj} = \{\mathbf{X}_{lj}, \mathbf{y}_{lj}\}$ . Consequently, the global objective function is defined as follows:

$$\min_{\{\{\mathbf{w}_l^j\}_{j \in V_l}\}_{l=1}^L} \sum_{l=1}^L \sum_{j \in V_l} f_{lj}(\mathbf{w}_l^j) + \mathcal{R}_c(\{\mathbf{w}_l^j\}) + \mathcal{R}_w(\{\mathbf{w}_l^j\}), \quad (1)$$

where  $\{\{\mathbf{w}_l^j\}_{j \in V_l}\}_{l=1}^L$  denotes the local model parameters across clients. Further, within each cluster  $V_l$ , there is a cluster-level parameter  $\mathbf{w}_l$  to which individual client parameters are aligned. The regularization function  $\mathcal{R}_c(\cdot)$  is designed to enforce global consensus or allow for personalization within a cluster, while  $\mathcal{R}_w(\cdot)$  imposes prior knowledge about the parameters (e.g., based on historical data or domain expertise).

We now extend our formulation by introducing a set of cluster-level parameters  $\{\mathbf{w}_l\}_{l=1}^L$  and an overall global parameter  $\mathbf{w}_0$ . With these additional parameters, the objective function is rewritten as:

$$\min_{\mathbf{W}} \sum_{l=1}^L \sum_{j \in V_l} f_{lj}(\mathbf{w}_l^j) + \mathcal{R}_c(\{\mathbf{w}_l^j\}_{j=1}^{N_l}, \mathbf{w}_l) + \mathcal{R}_c(\{\mathbf{w}_l\}_{l=1}^L, \mathbf{w}_0) + \mathcal{R}_w(\mathbf{w}_0) + \mathcal{R}_w(\{\mathbf{w}_l\}_{l=1}^L). \quad (2)$$

Here,  $\mathbf{W}$  denotes the collection of all parameters, including the local client parameters  $\{\mathbf{w}_l^j\}$ , the cluster-level parameters  $\{\mathbf{w}_l\}_{l=1}^L$ , and the overall global parameter  $\mathbf{w}_0$ . In this formulation, the first regularization term enforces intra-cluster relationships (which may include personalization), while the second term captures the relationship between the cluster-level parameters and the overall global parameter.

To further refine the model, we decompose the regularization into per-cluster components by defining  $\mathcal{R}_c^l(\mathbf{w}_l, \mathbf{w}_0)$ , that

quantifies the discrepancy between the cluster-level parameter  $\mathbf{w}_l$  and the global parameter  $\mathbf{w}_0$  for the  $l$ th cluster. Moreover, we allow the intra-cluster regularization to be flexible enough to enable personalization for each client by denoting it as  $\mathcal{R}_c^{l,j}(\mathbf{w}_l^j, \mathbf{w}_l)$ . This leads to the equivalent formulation:

$$\min_{\mathbf{W}} \sum_{l=1}^L \left[ \sum_{j \in V_l} f_{lj}(\mathbf{w}_l^j) + \mathcal{R}_c^{l,j}(\mathbf{w}_l^j, \mathbf{w}_l) + \mathcal{R}_c^l(\mathbf{w}_l, \mathbf{w}_0) + \mathcal{R}_w^l(\mathbf{w}_l) \right] + \mathcal{R}_w^0(\mathbf{w}_0). \quad (3)$$

This formulation decouples the regularization terms, allowing separate control the intra-cluster consensus (or personalization) between local and cluster-level models (via  $\mathcal{R}_c^{l,j}$ ) and the inter-cluster relationship with the global model (via  $\mathcal{R}_c^l$ ), while incorporating prior knowledge through the  $\mathcal{R}_w$  terms.

Motivated by the need to efficiently handle non-smooth regularizers that promote sparsity, our approach leverages the Alternating Direction Method of Multipliers (ADMM). In our formulation, the regularization function  $\mathcal{R}_w$  is often non-smooth to induce sparsity in the parameters, which enhances interpretability and leverages prior knowledge. ADMM naturally accommodates such non-smooth terms through its proximal update steps, thereby eliminating the need for ad hoc modifications. Moreover, ADMM has been demonstrated to improve learning performance in terms of both convergence speed and accuracy.

In order to apply ADMM more efficiently, we introduce auxiliary variables  $\mathbf{Z}$  and  $\mathbf{Q}$  that mirror the structure of  $\mathbf{W}$ . Specifically, we define  $\mathbf{Z} = \{\{\mathbf{z}_l^j\}_{j \in V_l, l=1}^L \cup \{\mathbf{z}_l\}_{l=1}^L\}$ ,  $\mathbf{Q} = \{\{\mathbf{q}_l^j\}_{j \in V_l, l=1}^L \cup \{\mathbf{q}_l\}_{l=1}^L\}$ , where the variables  $\mathbf{z}_l^j$  and  $\mathbf{q}_l^j$  are copies of the local parameters  $\mathbf{w}_l^j$  and the cluster-level parameters  $\mathbf{w}_l$ , respectively, and  $\mathbf{z}_l$  and  $\mathbf{q}_l$  are copies of  $\mathbf{w}_l$  and the global parameter  $\mathbf{w}_0$ . With these auxiliary variables, our ADMM-based formulation is modified as follows:

$$\min_{\mathbf{W}, \mathbf{Z}, \mathbf{Q}} \sum_{l=1}^L \left[ \sum_{j \in V_l} f_{lj}(\mathbf{w}_l^j) + \mathcal{R}_c^{l,j}(\mathbf{z}_l^j, \mathbf{q}_l^j) + \mathcal{R}_c^l(\mathbf{z}_l, \mathbf{q}_l) + \mathcal{R}_w^l(\mathbf{w}_l) \right] + \mathcal{R}_w^0(\mathbf{w}_0)$$

subject to:  $\mathbf{z}_l^j = \mathbf{w}_l^j, \quad \mathbf{q}_l^j = \mathbf{w}_l, \quad \forall j \in [N_l], \forall l \in [L],$

$$\mathbf{z}_l = \mathbf{w}_l, \quad \mathbf{q}_l = \mathbf{w}_0, \quad \forall l \in [L]. \quad (4)$$

However, in non-convex settings ADMM generally requires that the variables updated in its second block be smooth in order to ensure convergence [18], [25]–[28], as well as to support asynchronous and multiple updates [29]. To address this issue, we iteratively apply smoothing techniques to those non-smooth components that are updated in the second block and that could potentially impede the update process. This reformulation preserves the sparsity-inducing properties of  $\mathcal{R}_w$  while ensuring that the ADMM updates remain stable and convergent. In the next section, we provide further details.

### III. HIERARCHICAL FEDERATED SMOOTHING ADMM

To address the challenges in optimizing non-smooth functions within the ADMM framework, we employ smoothing techniques to approximate a function  $g$  by a family of smooth upper-bound functions  $\tilde{g}$  [30]. These approximations are beneficial in locally Lipschitz settings, ensuring continuity, differentiability, and well-behaved gradients—features that are critical for efficient optimization. Importantly, as the smoothing parameter  $\mu$  tends to zero,  $\tilde{g}(\mathbf{x}; \mu)$  converges to the original function  $g(\mathbf{x})$ , thereby preserving its essential properties [31].

Specifically, we approximate the intra-cluster regularization functions  $\mathcal{R}_c^{l,j}(\mathbf{z}_l^j, \mathbf{q}_l^j)$  and the inter-cluster regularization functions  $\mathcal{R}_c^l(\mathbf{z}_l, \mathbf{q}_l)$  by their smooth counterparts  $\tilde{\mathcal{R}}_c^{l,j}(\cdot; \mu)$  and  $\tilde{\mathcal{R}}_c^l(\cdot; \mu)$ , respectively. Moreover, in cases where the loss functions  $f_{lj}(\cdot)$  are not proximal-friendly, we employ smooth approximations  $\tilde{f}_{lj}(\cdot; \mu)$  to enable a linearized update step. We emphasize that these smooth approximations serve as upper bounds to the original functions, converging to the true function as  $\mu$  decreases.

With these definitions, the approximate augmented Lagrangian for our problem can be written as

$$\begin{aligned} \bar{\mathcal{L}}_{\sigma, \mu}(\mathbf{W}, \mathbf{Z}, \mathbf{Q}, \mathbf{\Lambda}, \mathbf{\Gamma}) = & \sum_{l=1}^L \left\{ \sum_{j \in V_l} f_{lj}(\mathbf{w}_l^j) + \tilde{\mathcal{R}}_c^{l,j}(\mathbf{z}_l^j, \mathbf{q}_l^j; \mu_l^j) \right. \\ & \left. + \tilde{\mathcal{R}}_c^l(\mathbf{z}_l, \mathbf{q}_l; \mu_l) + \mathcal{R}_w^l(\mathbf{w}_l) \right\} + \mathcal{R}_w^0(\mathbf{w}_0) + \sum_{l=1}^L \left\{ \sum_{j \in V_l} \left[ \right. \right. \\ & \left. \langle \mathbf{\Lambda}_l^j, \mathbf{w}_l^j - \mathbf{z}_l^j \rangle + \frac{\sigma_l^j}{2} \|\mathbf{w}_l^j - \mathbf{z}_l^j\|^2 + \langle \mathbf{\Gamma}_l^j, \mathbf{w}_l - \mathbf{q}_l^j \rangle + \frac{\sigma_l^j}{2} \|\mathbf{w}_l - \mathbf{q}_l^j\|^2 \right] + \\ & \left. \langle \mathbf{\Lambda}_l, \mathbf{w}_l - \mathbf{z}_l \rangle + \frac{\sigma_l}{2} \|\mathbf{w}_l - \mathbf{z}_l\|^2 + \langle \mathbf{\Gamma}_l, \mathbf{w}_0 - \mathbf{q}_l \rangle + \frac{\sigma_l}{2} \|\mathbf{w}_0 - \mathbf{q}_l\|^2 \right\} \quad (5) \end{aligned}$$

Here,  $\mathbf{\Lambda} = \{\mathbf{\Lambda}_l^j, \mathbf{\Lambda}_l\}$  and  $\mathbf{\Gamma} = \{\mathbf{\Gamma}_l^j, \mathbf{\Gamma}_l\}$  are the collections of dual variables and  $\{\sigma_l^j, \sigma_l\}$  are the penalty parameters. In our algorithm, each client  $j \in V_l$  and each cluster head  $l$  maintains its own update counter—denoted by  $k_l^j$  for clients and  $k_l$  for cluster heads—that reflects the number of updates it has performed. Nodes are allowed to update asynchronously and can execute multiple updates within each global iteration  $k_0$ . This design enables each node to progress at its own pace based on available computational resources and the characteristics of its local data, a flexibility that is essential for accommodating the inherent heterogeneity in federated learning environments.

To progressively adjust the approximations, the parameters  $\mu_l^j, \sigma_l^j$  and  $\mu_l, \sigma_l$  are updated at each local iteration  $k_l^j$  and  $k_l$  respectively as following, where  $c, d, \alpha, \beta > 0$ :

$$\sigma_l^j(k_l^j) = c\sqrt{k_l^j}, \mu_l^j(k_l^j) = \frac{\alpha}{\sqrt{k_l^j}}, \sigma_l(k_l) = d\sqrt{k_l}, \mu_l(k_l) = \frac{\beta}{\sqrt{k_l}} \quad (6)$$

By assuming that each function  $f_{lj}(\cdot)$  is proximable and weakly convex, we update each  $\mathbf{w}_l^j$  as:

$$\mathbf{w}_l^j(k_l^j) = \mathbf{Prox}_{f_{lj}}(\mathbf{u}_l; \frac{1}{\sigma_l^j(k_l^j)}) \quad (7)$$

where  $\mathbf{u}_l = \mathbf{z}_l^j(k_l^j-1) - \frac{\mathbf{\Lambda}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)}$ . If a closed-form solution for the proximal operator of any  $f_{lj}(\cdot)$  is not available, we can employ an upper-bound smooth approximation, as suggested in [32], and linearize the update step.

Subsequently, in the next step, we update each  $\mathbf{z}_l^j$  and  $\mathbf{q}_l^j$  in a single step as:

$$\begin{aligned} \begin{bmatrix} \mathbf{z}_l^j(k_l^j) \\ \mathbf{q}_l^j(k_l^j) \end{bmatrix} = & \arg \min_{\mathbf{z}_l^j, \mathbf{q}_l^j} \tilde{\mathcal{R}}_c^{l,j}(\mathbf{z}_l^j, \mathbf{q}_l^j; \mu_l^j(k_l^j)) + \frac{\sigma_l^j(k_l^j)}{2} \times \left( \left\| \mathbf{z}_l^j - \right. \right. \\ & \left. \left. \mathbf{w}_l^j(k_l^j) - \frac{\mathbf{\Lambda}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} \right\|_2^2 + \left\| \mathbf{q}_l^j - \mathbf{w}_l(k_l) - \frac{\mathbf{\Gamma}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} \right\|_2^2 \right) \quad (8) \end{aligned}$$

Following the simplification provided in [33], we obtain:

$$\begin{aligned} \begin{bmatrix} \mathbf{z}_l^j(k_l^j) \\ \mathbf{q}_l^j(k_l^j) \end{bmatrix} = & \frac{1}{2} \begin{bmatrix} \mathbf{w}_l^j(k_l^j) + \frac{\mathbf{\Lambda}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} + \mathbf{w}_l(k_l) + \frac{\mathbf{\Gamma}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} \\ \mathbf{w}_l^j(k_l^j) + \frac{\mathbf{\Lambda}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} + \mathbf{w}_l(k_l) + \frac{\mathbf{\Gamma}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} \end{bmatrix} \\ & + \frac{1}{2} \begin{bmatrix} -\mathbf{e} \\ \mathbf{e} \end{bmatrix} \quad (9) \end{aligned}$$

where  $\mathbf{e} = \mathbf{Prox}_{\tilde{\mathcal{R}}_c^{l,j}(\cdot, \cdot; \mu_l^j(k_l^j))} \left( -\mathbf{w}_l^j(k_l^j) - \frac{\mathbf{\Lambda}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)} + \mathbf{w}_l(k_l) + \frac{\mathbf{\Gamma}_l^j(k_l^j-1)}{\sigma_l^j(k_l^j)}; \frac{2}{\sigma_l^j(k_l^j)} \right)$ . When  $\mathcal{R}_c^{l,j}(\cdot)$  is defined using  $\ell_1$ , MCP, or SCAD penalties, the proximal operator  $\mathbf{Prox}_{\tilde{\mathcal{R}}_c^{l,j}(\cdot, \cdot; \mu_l^j)}$  is decomposable and can be efficiently computed.

Next, we update each  $\mathbf{\Lambda}_l^j$  and  $\mathbf{\Gamma}_l^j$  as follows:

$$\mathbf{\Lambda}_l^j(k_l^j) = \mathbf{\Lambda}_l^j(k_l^j-1) + \sigma_l^j(k_l^j) \left( \mathbf{w}_l^j(k_l^j) - \mathbf{z}_l^j(k_l^j) \right) \quad (10)$$

$$\mathbf{\Gamma}_l^j(k_l^j) = \mathbf{\Gamma}_l^j(k_l^j-1) + \sigma_l^j(k_l^j) \left( \mathbf{w}_l(k_l) - \mathbf{q}_l^j(k_l^j) \right) \quad (11)$$

Next, we transmit the updated values  $k_l^j, \mathbf{q}_l^j(k_l^j)$ , and  $\mathbf{\Gamma}_l^j(k_l^j)$  to the cluster head.

At the cluster head  $l$ , we update  $\mathbf{w}_l$  as:

$$\mathbf{w}_l(k_l) = \mathbf{prox}_{\mathcal{R}_w^l}(\bar{\mathbf{r}}_l, v_l), \quad (12)$$

where  $v_l = \frac{1}{\sigma_l^{k_l} + \sum_{j \in V_l} \sigma_l^j(k_l^j+1)}$  and  $\bar{\mathbf{r}}_l = v_l \left( \sigma_l(k_l) \mathbf{z}_l^{(k_l-1)} - \mathbf{\Lambda}_l^{(k_l-1)} + \sum_{j \in V_l} (\sigma_l^j(k_l^j+1) \mathbf{q}_l^j(k_l^j) - \mathbf{\Gamma}_l^j(k_l^j)) \right)$ . The cluster head sends its variable to each client.

Then, we update each  $\mathbf{z}_l$  and  $\mathbf{q}_l$  in a parallel as:

$$\begin{aligned} \begin{bmatrix} \mathbf{z}_l^{(k_l)} \\ \mathbf{q}_l^{(k_l)} \end{bmatrix} &= \arg \min_{\mathbf{z}_l, \mathbf{q}_l} \tilde{\mathcal{R}}_c^l(\mathbf{z}_l, \mathbf{q}_l; \mu_l^{(k_l)}) + \frac{\sigma_l^{(k_l)}}{2} \times \left( \left\| \mathbf{z}_l - \mathbf{w}_l^{(k_l)} - \frac{\Lambda_l^{j(k_l-1)}}{\sigma_l^{(k_l)}} \right\|_2^2 + \left\| \mathbf{q}_l - \mathbf{w}_0^{(k_0)} - \frac{\Gamma_l^{(k_l-1)}}{\sigma_l^{(k_l)}} \right\|_2^2 \right), \end{aligned} \quad (13)$$

which can be driven in closed form solution similar to (9).

The updates for dual variables  $\Lambda_l$  and  $\Gamma_l$  are given by

$$\Lambda_l^{(k_l)} = \Lambda_l^{(k_l-1)} + \sigma_l^{(k_l)} \left( \mathbf{w}_l^{(k_l)} - \mathbf{z}_l^{(k_l)} \right) \quad (14)$$

$$\Gamma_l^{(k_l)} = \Gamma_l^{(k_l-1)} + \sigma_l^{(k_l)} \left( \mathbf{w}_0^{(k_0)} - \mathbf{q}_l^{(k_l)} \right) \quad (15)$$

Accordingly, we send the updated values  $k_l$ ,  $\mathbf{q}_l^{(k_l)}$ , and  $\Gamma_l^{(k_l)}$  to server.

The update step for  $\mathbf{w}_0$  is performed globally, based on aggregated information from all cluster head data after updating their respective variables. The update rule for  $\mathbf{w}_0$  is given by:

$$\mathbf{w}_0^{(k_0)} = \text{Prox}_{R_w^0} \left( \xi \sum_{l=1}^L (\sigma_l^{(k_l+1)} \mathbf{q}_l^{(k_l)} + \Gamma_l^{(k_l)}); \xi \right), \quad (16)$$

where  $\xi = \frac{1}{\sum_{l=1}^L \sigma_l^{(k_l+1)}}$ . The server sends its variable to each cluster head accordingly.

We emphasize that the asynchronicity and multiple updates inherent in our framework do not compromise convergence. This robustness stems from the fact that both the client-level and cluster-head dual variables are updated via an ascent step only after their corresponding primal variables (e.g.,  $\mathbf{z}_l$  and  $\mathbf{q}_l$ ) have been updated. Such an update order minimizes the disruptive effect of the dual updates and the associated penalty parameters, introducing only minor, summable noise due to smoothing. Moreover, although clients and cluster heads update asynchronously and independently, we enforce periodic synchronization. In particular, during a global update of  $\mathbf{w}_0$ , local updates (for  $\mathbf{z}_l$ ,  $\mathbf{q}_l$ ,  $\Lambda_l$ , and  $\Gamma_l$ ) are temporarily paused, and similarly, the global update step is suspended while local nodes perform their updates. We further assume the existence of an integer  $K_a$  such that, within every  $K_a$  global iteration, every client and cluster head updates its variables at least once. The following theorem is provided in case consensus is needed for hierarchical learning.

**Theorem 1.** Assume that the each function  $f_{lj}(\cdot)$ ,  $R_w^l(\cdot)$ ,  $\forall l \in [L] \cup 0$  has bounded gradients indicated by  $\nu_f$  and  $\nu_r$  respectively, parameters  $\sigma_l^j, \mu_l^j, \sigma_l, \mu_l$  are updated according to (6),  $\alpha c \geq \sqrt{20}\nu_f$  and  $\beta d \geq \sqrt{20}(\omega)$  where  $\omega = \max_l |N_l| \nu_f + \nu_r$ . By choosing  $\mathcal{R}_c^{l,j}(\cdot)$  and  $\mathcal{R}_c^l(\cdot)$  are total variation norm  $\nu_f \|\cdot\|_1$  and  $\omega \|\cdot\|_1$ , HFSAD reaches consensus and converges to a stationary point satisfying the following optimality conditions:

$$0 \in \sum_{l \in [L]} \sum_{j \in N_l} f_{lj}(\mathbf{w}_l^{j*}) + R_w^l(\mathbf{w}_l^*) + R_w^0(\mathbf{w}^*) \quad (17a)$$

$$\mathbf{w}_l^* = \mathbf{w}_l^{j*}, \quad \forall j \in N_l \quad \forall l \in [L] \quad (17b)$$

$$\mathbf{w}_0^* = \mathbf{w}_l^*, \quad \forall l \in [L] \quad (17c)$$

## IV. SIMULATION RESULTS

We consider the restricted-domain SCAD-penalized robust phase retrieval, where each client  $j \in V_l$  optimizes:  $f_{lj}(\mathbf{w}) = |y_{lj} - |\langle \mathbf{x}_{lj}, \mathbf{w} \rangle|^2| + \mathcal{I}(\mathbf{w})$ , with observations  $\mathbf{x}_{lj} \in \mathbb{R}^M$ , intensity  $y_{lj} \in \mathbb{R}$ , and the indicator function  $\mathcal{I}(\mathbf{w})$  equals 0 if  $|w_m| \leq 5$  for all  $m \in [M]$ , and  $+\infty$  otherwise. The SCAD penalty is  $\eta_l P_{\lambda, \gamma}(\cdot)$  with  $\lambda = 0.1$ ,  $\gamma = 2.4$ , and weighting  $\eta_l = |N_l| - \frac{1}{L}$  for  $R_w^l(\cdot)$  and  $\eta_l = 1$  for  $R_w^0(\cdot)$ . Consensus is enforced via total variation regularization:  $\mathcal{R}_c^{l,j}(\cdot) = \omega \|\cdot\|_1$ ,  $\mathcal{R}_c^l(\cdot) = \omega_l \|\cdot\|_1$ , where  $\omega$  is set by  $5 \max_{l,j} \|\mathbf{x}_{lj}\|$ , and  $\omega_l = \max_l |N_l| \omega + \eta_l \lambda \gamma$ . We benchmark against the centralized sub-gradient method in [34], evaluating performance with relative error:  $\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2}$ , averaged over 100 trials. Simulation parameters include:  $c = \omega$ ,  $d = \frac{\omega_0}{25}$ ,  $\alpha = \sqrt{20}$ , and  $\beta = 25\sqrt{20}$ . Measurements are generated as:  $y_{lj} = |\langle \mathbf{x}_{lj}, \mathbf{w} \rangle|^2 + \epsilon_{lj}$ ,  $\mathbf{x}_{ij} \sim \mathcal{N}(0, \mathbf{I}) \odot \mathbf{B}_h^l$ ,  $\mathbf{w} \sim \mathcal{N}(0, \mathbf{I}) \odot \mathbf{B}_s$ , where  $\odot$  denotes the Hadamard product, diagonal matrices  $\mathbf{B}_h^l, \mathbf{B}_s \in \mathbb{R}^{M \times M}$  have sparsity ratios  $p = 0.8$ ,  $s = 0.3$ , and noise  $\epsilon_{lj}$  is drawn i.i.d. from a mixture exponential distribution with parameters  $c_1 = 0.9$ ,  $c_2 = 0.1$ , and  $\lambda_2 = \frac{\lambda_1}{10}$ . The SNR ( $\gamma = -20\text{dB}$ ) determines  $\lambda_1$  via:  $\lambda_1 = \sqrt{\frac{(\sum_{l=1}^L N_l) \times 21.8 \times 10^{\frac{\gamma}{10}}}{\sum_{l=1}^L \sum_{j=1}^{N_l} \|\mathbf{x}_{lj}\|^2}}$ . In simulations, we set  $(L, N, M) = (5, 50, 25)$ , with  $N_l = N$ .

In scenario one (Fig. 1), HFSAD with  $K_m = 10$  local updates per iteration shows superior convergence and lower relative error compared to Sub. Scenario two (Fig. 2) compares varying local update steps ( $K_m = 1, 5, 10, 20$ ), demonstrating fastest convergence with  $K_m = 20$ , while fewer steps yield slower but adequate convergence. Scenario three (Fig. 3) examines HFSAD performance under asynchronicity with probabilities  $p_c = 0.3, 0.5, 0.7, 1$  when  $K_m = 1$ , showing comparable and acceptable performance results between different level of collaboration probabilities.

## V. CONCLUSION

This paper introduced the Hierarchical Federated Smoothing ADMM (HFSAD), a novel federated learning framework designed to address non-convex and non-smooth optimization challenges in hierarchical federated learning scenarios. The proposed method integrates smoothing techniques with ADMM, effectively accommodating asynchronous updates and supporting multiple local updates per iteration, which makes it well-suited for heterogeneous network and computational environments. Our experiments on SCAD-penalized robust phase retrieval problems demonstrated that HFSAD achieves improved convergence behavior and higher accuracy compared to traditional centralized methods. These results highlight HFSAD's potential to efficiently handle complex optimization landscapes while offering robustness and flexibility required for real-world federated learning applications.

## REFERENCES

- [1] X. Chen, W. Liu, and Y. Zhang, "Quantile regression under memory constraint," *The Annals of Statistics*, vol. 47, no. 6, pp. 3244–3273, Dec. 2019.

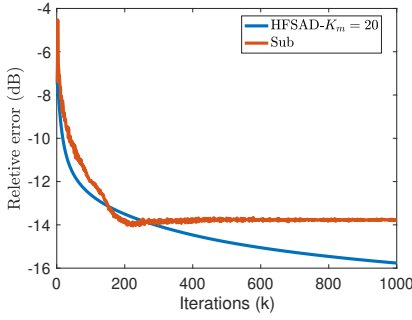


Fig. 1: Relative error comparison of HF-SAD and Sub

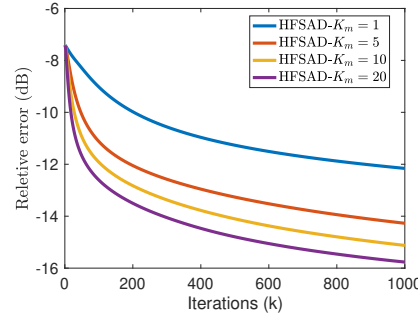


Fig. 2: Relative error comparison across different update steps

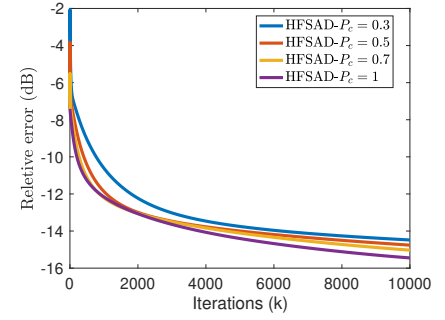


Fig. 3: Relative error comparison across different collaboration probability

- [2] X. Zhou and Y. Xiang, "ADMM-based differential privacy learning for penalized quantile regression on distributed functional data," *Mathematics*, vol. 10, no. 16, p. 2954, Aug. 2022.
- [3] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–36, July 2021.
- [4] X. Zhou, W. Liang, I. Kevin, K. Wang, Z. Yan, L. T. Yang, W. Wei, J. Ma, and Q. Jin, "Decentralized p2p federated learning for privacy-preserving and resilient mobile robotic systems," *IEEE Wireless Communications*, vol. 30, no. 2, pp. 82–89, Apr. 2023.
- [5] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *IEEE International Conference on Big Data*, 2020, pp. 15–24.
- [6] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji *et al.*, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.
- [7] J. Zhang, Z. Li, B. Li, J. Xu, S. Wu, S. Ding, and C. Wu, "Federated learning with label distribution skew via logits calibration," in *Proceedings of the 39th International Conference on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 162, 2022, pp. 26 311–26 329.
- [8] E. Diao, J. Ding, and V. Tarokh, "HeteroFL: Computation and communication efficient federated learning for heterogeneous clients," in *International Conference on Learning Representations (ICLR)*, 2020.
- [9] D. Li and J. Wang, "FedMD: Heterogeneous federated learning via model distillation," *arXiv preprint arXiv:1910.03581*, 2019.
- [10] V. Smith, C. Chiang, M. Sanjabi, and A. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems (NIPS)* 30, 2017, pp. 4424–4434.
- [11] 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 34 (NeurIPS 2021)*, 2021.
- [12] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE International Conference on Communications (ICC)*, 2019.
- [13] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2021, pp. 19–35.
- [14] J. Shin, Y. Li, Y. Liu, and S.-J. Lee, "Fedbalancer: Data and pace control for efficient federated learning on heterogeneous clients," in *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services*, 2022, pp. 436–449.
- [15] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, May 2020.
- [16] X. Yu, Z. Liu, W. Wang, and Y. Sun, "Clustered federated learning based on nonconvex pairwise fusion," *Information Sciences*, vol. 678, p. 120956, 2024.
- [17] R. Mirzaeifard, D. Ghaderyan, and S. Werner, "Decentralized smoothing ADMM for quantile regression with non-convex sparse penalties," *arXiv preprint arXiv:2408.01307*, 2024.
- [18] R. Mirzaeifard, N. K. Venkatesowda, and S. Werner, "Robust phase retrieval with non-convex penalties," in *56th IEEE Asilomar Conference on Signals, Systems, and Computers*, 2022, pp. 1291–1295.
- [19] R. Mirzaeifard, V. C. Gogineni, N. K. Venkatesowda, and S. Werner, "Dynamic graph topology learning with non-convex penalties," in *30th IEEE European Signal Processing Conference*, 2022, pp. 682–686.
- [20] Y. Sarcheshmehpour, Y. Tian, L. Zhang, and A. Jung, "Clustered federated learning via generalized total variation minimization," *IEEE Transactions on Signal Processing*, vol. 71, pp. 4240–4256, 2023.
- [21] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *The Annals of Statistics*, vol. 38, no. 2, pp. 894–942, Apr. 2010.
- [22] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, Dec. 2001.
- [23] S. M. Azimi-Abarghouy, N. Bastianello, K. H. Johansson, and V. Fodor, "Hierarchical federated admm," *IEEE Networking Letters*, 2025.
- [24] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Hierarchical federated learning with quantization: Convergence analysis and system design," *IEEE Transactions on Wireless Communications*, vol. 22, no. 1, pp. 2–18, 2022.
- [25] Y. Wang, W. Yin, and J. Zeng, "Global convergence of admm in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, Jan. 2019.
- [26] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Apr. 2015, pp. 3836–3840.
- [27] M. Yashtini, "Convergence and rate analysis of a proximal linearized ADMM for nonconvex nonsmooth optimization," *Journal of Global Optimization*, vol. 84, no. 4, pp. 913–939, Dec. 2022.
- [28] A. Themelis and P. Patrinos, "Douglas–Rachford splitting and ADMM for nonconvex optimization: Tight convergence results," *SIAM Journal on Optimization*, vol. 30, no. 1, pp. 149–181, Jan. 2020.
- [29] Y. Miao, D. Kuang, X. Li, S. Xu, H. Li, K.-K. R. Choo, and R. H. Deng, "Privacy-preserving asynchronous federated learning under non-iid settings," *IEEE Transactions on Information Forensics and Security*, 2024.
- [30] R. Mirzaeifard, N. K. Venkatesowda, V. C. Gogineni, and S. Werner, "Smoothing admm for sparse-penalized quantile regression with non-convex penalties," *IEEE Open Journal of Signal Processing*, 2023.
- [31] X. Chen, "Smoothing methods for nonsmooth, nonconvex minimization," *Mathematical Programming*, vol. 134, pp. 71–99, Aug. 2012.
- [32] R. Mirzaeifard, N. K. Venkatesowda, V. C. Gogineni, and S. Werner, "ADMM for sparse-penalized quantile regression with non-convex penalties," in *30th IEEE European Signal Processing Conference*, 2022, pp. 2046–2050.
- [33] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," in *Proc. 23th Int. Conf. Knowl. Discovery and Data Mining*, 2017, pp. 205–213.
- [34] D. Davis, D. Drusvyatskiy, K. J. MacPhee, and C. Paquette, "Subgradient methods for sharp weakly convex functions," *Journal of Optimization Theory and Applications*, vol. 179, pp. 962–982, Dec. 2018.