Personalized Federated Learning via Convex Clustering

Aleksandar Armacki[1], Dragana Bajovic[2], Dusan Jakovetic[3], and Soummya Kar[1]

[1]Carnegie Mellon University, Pittsburgh, PA
Email: {aarmacki, soummyak}@andrew.cmu.edu

[2]Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia
Email: dbajovic@uns.ac.rs

[3]Faculty of Sciences, University of Novi Sad, Novi Sad, Serbia
Email: dusan.jakovetic@dmi.uns.ac.rs

Abstract—We propose a parametric family of algorithms for personalized federated learning with locally convex user costs. The proposed framework is based on a generalization of convex clustering in which the differences between different users' models are penalized via a sum-of-norms penalty, weighted by a penalty parameter λ. The proposed approach enables "automatic" model clustering, without prior knowledge of the hidden cluster structure, nor the number of clusters. Analytical bounds on the weight parameter, that lead to simultaneous personalization, generalization and automatic model clustering are provided. The solution to the formulated problem enables personalization, by providing different models across different clusters, and generalization, by providing models different than the per-user models computed in isolation. We then provide an efficient algorithm based on the Parallel Direction Method of Multipliers (PDMM) to solve the proposed formulation in a federated server-users setting. Numerical experiments corroborate our findings. As an interesting byproduct, our results provide several generalizations to convex clustering.

I. Introduction

Federated learning (FL) is a paradigm in which many users collaborate, with the goal of learning a joint model [1]. FL has become a mainstream research topic and has been applied in various domains, including, e.g., finance, manufacturing, health, and smart cities [2], [3].

With a FL system, each user has a local dataset, with private and possibly sensitive data. The data distribution across users is typically highly heterogeneous. There may be a huge amount of users, wherein each user contributes with a proportionally small local dataset. Therefore, the federation provides users with the benefit of training on the joint data, effectively offering broader knowledge and better generalization. However, due to the highly heterogeneous nature of the data, it is nontrivial to design a FL system where individual users achieve better performance though the federation when compared with models trained on their own local data. In fact, the authors in [4] show that in many tasks, users may actually not benefit from participating in federated learning. The globally trained model underperforms on their local data, compared to the model solely trained on the local data. Moreover, applying privacy preserving techniques further deteriorates the performance. On the other hand, users with very small datasets suffer from overfitting and poor generalization of models trained only on their local data.

To amend these problems and reap the benefits of both worlds – the abundance of data and better learning that the federation offers, as well as adapting the models to perform well on the local data, *personalized federated learning* is introduced. Unlike the standard federated learning, the goal of personalized federated learning is to learn multiple models. In particular, let N be the number of participating users, with $f : \mathbb{R}^d \mapsto \mathbb{R}$ a given cost function. Then, the goal of standard FL is to solve

$$\arg\min_{x \in \mathbb{R}^d} F_{\text{global}}(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x), \qquad (1)$$

where $f_i(x)$ is the cost function f evaluated on the local dataset of the i-th user. Contrary to this approach, the (broad) goal of personalized FL is to learn N models, by solving

$$\arg\min_{x_1, \ldots, x_N \in \mathbb{R}^d} F_{\text{local}}(x_1, \ldots, x_N) = \frac{1}{N} \sum_{i=1}^{N} f_i(x_i), \qquad (2)$$

subject to appropriately defined constraints. Depending on the constraints imposed and the approach taken to solving (2), the literature on personalized FL adopts different approaches to personalization, including multi-task learning [5], [6], fine-tuning [7], [8], knowledge distillation [9], [10], [11], [12] and clustering-based approaches [13], [14], [15].

In this paper, we propose a novel approach to personalized federated learning that enables simultaneous personalization, generalization, and model clustering. The approach is based on the following novel personalized FL (convex problem)

formulation:

$$\operatorname*{arg\,min}_{x_1,\ldots,x_N \in \mathbb{R}^d} F(x_1,\ldots,x_N) = \frac{1}{N}\sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j \neq i} \|x_i - x_j\|, \tag{3}$$

where $\lambda > 0$ is a penalty parameter, and $\|\cdot\|$ stands for the Euclidean norm. Compared with (2), the formulation (3) has a regularization term controlled with $\lambda > 0$, that penalizes the differences between the local nodes' solutions via a sum-of-norms penalty.

Formulation (3) may be seen as a generalization of convex clustering, e.g., [16], where, instead of the (quadratic) distance functions, which penalize the departure from a local data point, we use arbitrary convex local losses.

Problem (3) is also related with the personalized FL formulation in [17], which, instead of the sum of norms of pairwise distances uses the sum of *squared* norms of distances of local solutions to their average. As we show in the paper, although the two formulations are resembling, the solution structures of the two formulations are qualitatively very different.

Specifically, as we show in the paper, the solution to (3) has several interesting properties, which, to the best of our knowledge, are not (jointly) exhibited with any of the previous personalized FL formulations. Namely, a solution to (3) has a clustered structure: depending on the penalty parameter λ and similarity of the local functions, local solutions $x_i^\star$ of (3) are equal across certain users' groups (clusters). The number of groups (clusters) K is automatically determined as part of the solution. To further illustrate benefits of this feature, suppose that the users exhibit an (unknown) clustered structure such that each user's data within a cluster comes from the same distribution, while the distributions that correspond to different clusters are mutually different. If the different clusters' distributions are sufficiently far apart, the proposed method (3) uncovers the unknown cluster structure and hence allocates the same models to all users within the same cluster. This allows within-cluster generalization, i.e., users to effectively enlarge their training data by harnessing data of all users from the same cluster. In addition, depending on the distance between the different clusters' distributions, the method (3) allows for a controlled across-clusters generalization; intuitively, it allows a user to harness data from another clusters' distribution, but with a different (reduced) "weight" when compared to within-cluster data. If, at an extreme, the difference between different clusters' distributions is negligible (but this information is unknown), then users should clearly use the global model (1). This is again captured by (3), because (as shown in the paper) it matches (1) for λ above a threshold.

In summary, our contributions are the following: First, we propose a novel formulation for personalized federated learning (3), the solution of which has a clustering structure while at the same time preserving generalization abilities.

Second, we provide a condition on the penalty parameter λ, with theoretical guarantees, for discovering the "hidden structure" underlying the models; this condition is expressed in terms of the well-established diversity of the local functions, hence making a strong connection and justifying analytically the use of this quantity.

Third, the proposed solution "automatically" determines the number of models K, i.e., K need not be known in advance.

Fourth, in order to show that the proposed formulation (3) is amenable to federated server-client settings, we provide an efficient algorithm to solve the formulation that is based upon the Parallel Direction Method of Multipliers (PDMM) [18].

Finally, we demonstrate by simulation examples, on a supervised binary classification problem, that the proposed solution exhibits 1) generalization, i.e., improves testing accuracy with respect to the users models trained in isolation; 2) personalization, i.e., improves testing accuracy with respect to the global FL model (1); and 3) achieves a comparable (or better) generalization and personalization (in the sense of 1) and 2)) than [6], while at the same time uncovering cluster structure, hence reducing the number of distinct models from N to K.

With respect to existing personalized FL approaches discussed above, the works [13], [14], [15], [12] also account for users' clustering in a certain way, but very differently from our approach. Most notably, existing approaches aim to uncover "cluster identities" first and subsequently provide loss minimizations across cluster groups in isolation from other groups. This within-clusters isolation may reduce overall generalization ability of the models. In contrast, the proposed approach allows also for across-clusters generalization that is further controlled by the penalty parameter λ. It is worth noting that reference [19] introduces formulation similar to (3), but in a different context of distributed consensus optimization. Most importantly, they are only concerned with the question when (3) matches (1), i.e., when (3) leads to a global consensus across local models; they are not concerned, nor they study personalization (clustering) abilities of (3).

Our results are also of direct interest to convex clustering, e.g., [16], as they provide recovery guarantees for generalized convex clustering, when the squared quadratic loss $f_i(x) := \|x_i - a_i\|^2$ per data point $a_i \in \mathbb{R}^d$ is replaced with an arbitrary differentiable convex loss, e.g., the Huber loss.

Paper organization. The rest of the paper is organized as follows. Section II describes the problem of interest and outlines the assumptions used in the analysis. Section III presents the recovery guarantees of the method. Section IV outlines an efficient algorithm for solving the proposed problem in the federated setting. Section V presents numerical experiments, and Section VI concludes the paper. Supporting proofs, practical considerations, and additional numerical experiments can be found in [20]. The next paragraph introduces the notation used throughout the paper.

Notation. The set of real numbers is denoted by $\mathbb{R}$, while $\mathbb{R}^d$ denotes the corresponding d-dimensional vector space; $\|\cdot\|$ represents the standard Euclidean norm. $\langle \cdot, \cdot \rangle$ represents the standard vector product over the space of real vectors. $[N]$ denotes the set of integers up to and including N, i.e., $[N] = \{1, \ldots, N\}$.

II. PROBLEM FORMULATION

Consider a collection of N users, $i = 1, \ldots, N$, that participate in a federated learning activity. Each user i holds a function $f_i : \mathbb{R}^d \to \mathbb{R}$. Function f_i may correspond, e.g., to

an empirical loss with respect to the local data set available at user i. We make the following assumptions throughout the paper.

Assumption 1. *For each* $i = 1, \ldots, N$, *function* $f_i : \mathbb{R}^d \to \mathbb{R}$ *is convex and coercive, i.e.,* $f_i(x) \to +\infty$ *whenever* $\|x\| \to +\infty$.

Assumption 2. *For each* $i = 1, \ldots, N$, *function* $f_i : \mathbb{R}^d \to \mathbb{R}$ *has Lipschitz continuous gradients, i.e. the following holds*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \text{ for all } x, y \in \mathbb{R}^d.$$

Note that, under the above assumptions, problems (1) and (3) are solvable. We denote by $y^\star \in \mathbb{R}^d$ a solution to (1) and by $\{x_i^\star\}$, $i = 1, \ldots, N$, $x_i^\star \in \mathbb{R}^d$, a solution to (3).

There are many machine learning models that satisfy Assumptions 1 and 2, such as supervised binary classification problems studied in Section V.

The high-level goal in personalized federated learning is that each user i finds a local model, say $x_i^\bullet \in \mathbb{R}^d$, that performs well on the local data (i.e., the value $f_i(x_i^\bullet)$ is low), but that also exhibits a generalization ability with respect to data available at other users $j \neq i$. In addition, a desirable feature of personalized federated learning is that the users are able to classify other users into two categories. The first category corresponds to those users $j \in \{1, 2, \ldots, N\}$ that have similar data (similar f_j's) to their own; the second category corresponds to those users whose local data is "sufficiently different" from theirs. With this classification in place, each user i can fully harness the data from "similar users" for an improved personalization while avoiding overfitting; e.g., when user i has a very few data points of its own, it effectively enlarges its data set while preserving personalization. On the other hand, the data from "sufficiently different users" should still be harnessed in a controlled way to further improve generalization abilities.

To account for the effects above, we provide a novel personalized learning formulation, where each user i wants to obtain the local model $x_i^\star \in \mathbb{R}^d$ such that $x^\star := ((x_1^\star)^\top, \ldots, (x_N^\star)^\top)^\top \in \mathbb{R}^{Nd}$ is a minimizer of (3), where $\lambda > 0$ is a tuning parameter. Intuitively, the term $\sum_{i=1}^N f_i(x_i)$ forces the local models x_i's to behave well with respect to local costs f_i's; the term $\lambda \sum_{j \neq i} \|x_i - x_j\|$ makes the local models be mutually close, hence enabling generalization. The penalization term $\sum_{j \neq i} \|x_i - x_j\|$ is known to enforce sparsity in other contexts, in the sense that it forces many of the x_i's to be mutually equal at a solution of (3).

It is interesting to compare our novel formulation (3) with the personalized federated learning formulation in [6]:

$$\arg\min_{x_1, \ldots, x_N \in \mathbb{R}^d} \frac{1}{N} \sum_{i=1}^N f_i(x_i) + \gamma \sum_{j \neq i} \|x_i - x_j\|^2, \quad (4)$$

where $\gamma > 0$ is a penalty parameter.

The difference of (4) with respect to (3) is that, in (4), the differences of local models x_i and x_j are penalized via the squared Euclidean norm, while with our formulation, the 2-norm appears without squares. There are several important implications of this difference with respect to the resulting personalized learning models. Most importantly, in contrast with (3), formulation (4) in general does not lead to model clustering for any $\lambda > 0$. In addition, as a side comment, the solutions to (1) and (4) are in general mutually different for any $\lambda > 0$ (in the sense that the solution $\{y_i^\star\}$ to (4) does not obey $y_i^\star = y_j^\star$, for all i, j, irrespective of the choice of λ). In contrast, with (3), we recover global model learning as in (1) for $\lambda > \widehat{\lambda}$.

We also connect (3) with convex clustering. Convex clustering, e.g. [16], is an appealing method to cluster N data points $a_i \in \mathbb{R}^d$, $i = 1, \ldots, N$. The method corresponds to solving problem (3) with $f_i(x) = \frac{1}{2}\|x - a_i\|^2$. Intuitively, to each data point i, we associate a candidate cluster center x_i, and then we enforce a (soft) constraint that many x_i's should be mutually equal. There are several efficient algorithms and cluster recovery guarantees results available for convex clustering, but only when $f_i(x) = \frac{1}{2}\|x - a_i\|^2$. Our results make a direct generalization of convex clustering to other loss metrics, e.g., the "distance" of a candidate cluster x_i from data point a_i may be measured through the Huber loss.

III. THEORETICAL GUARANTEES FOR OPTIMAL CLUSTER RECOVERY

In this section, we state and prove our main results on characterization of solutions to (3).

We start by defining the following auxiliary optimization problem associated to a certain (predefined) partition of users $C_1, \ldots, C_K$, $\cup_{k=1}^K C_k = [N]$ and $C_k \cap C_j = \emptyset$:

$$\arg\min_{w_1, \ldots, w_N \in \mathbb{R}^d} \frac{1}{N} \sum_{k=1}^K n_k g_k(w_k) + \lambda \sum_{l \neq k} n_k n_l \|w_k - w_l\|, \quad (5)$$

where $g_k(w) := 1/n_k \sum_{i \in C_k} f_i(w)$, for $w \in \mathbb{R}^d$, and $n_k = |C_k|$ is the number of elements in C_k, for $k = 1, \ldots, K$. Let $w_k^\star = w_k^\star(\lambda)$, $k = 1, \ldots, K$, denote a solution to (5). Note that problem (5) is solvable by Assumption 1.

Theorem 1. *Consider problem* (3). *Assume that, for some node partition* $C_1, C_2, \ldots, C_K$, *and parameter* λ, *there holds*

$$\lambda \geq \max_{k=1, \ldots, K} \max_{i, j \in C_k} \frac{\|\nabla f_i(w_k^\star) - \nabla f_j(w_k^\star)\|}{n_k}. \quad (6)$$

Next, let $\{x_i^\star\}$, $i = 1, \ldots, N$, *be defined as follows: for each* $i \in C_k$, *we let* $x_i^\star = w_k^\star$, *for* $k = 1, \ldots, K$, *where* $\{w_k^\star = w_k^\star(\lambda)\}$, $k = 1, \ldots, K$, *is a solution to* (5), *defined for the same partition* $C_1, \ldots, C_K$ *that verifies* (6). *Then,* $\{x_i^\star\}$, $i = 1, \ldots, N$, *is a solution of* (3).

Remark 1. *Note that Theorem 1 guarantees that at least one solution of* (3) *exhibits the clustered structure with respect to partition* $C_1, \ldots, C_K$, *while it does not preclude a scenario that there might be another solution of* (3) *that may not exhibit this cluster structure. However, when each of the* f_i's *is in addition assumed to be strictly convex, then* $\{x_i^\star\}$ *is unique, and it necessarily has the clustered structure.*

We next prove Theorem 1.

This can be easily seen based on Theorem 1 in [19].

Proof. The proof is in spirit similar to Theorem 1 in [21]. From the first order optimality conditions for (5), we obtain that, for each $k = 1, \ldots, K$, there must hold:

$$\nabla g_k(w_k^\star) + \lambda \sum_{l \neq k} n_l r_{kl}^\star = 0, \tag{7}$$

where $r_{kl}^\star$ is a subgradient of $\|w_k - w_l\|$ with respect to w_k, computed at the solution. For each $k, l = 1, \ldots, K$, $r_{kl}^\star$ must satisfy:

$$r_{kl}^\star = \begin{cases} \frac{w_k^\star - w_l^\star}{\|w_k^\star - w_l^\star\|}, & \text{for } w_k^\star \neq w_l^\star \\ \text{a vector } r \in \mathbb{R}^d \text{ s.t. } \|r\| \leq 1, & \text{otherwise} \end{cases} \tag{8}$$

We now turn to first order optimality conditions for the original problem (3):

$$\nabla f_i(x_i) + \lambda \sum_{j \neq i} s_{ij} = 0, \tag{9}$$

where s_{ij} is a subgradient of $\|x_i - x_j\|$ computed with respect to x_i. Similarly as in the above, at the solution, s_{ij} must satisfy:

$$s_{ij} = \begin{cases} \frac{x_i - x_j}{\|x_i - x_j\|}, & \text{for } x_i \neq x_j \\ \text{a vector } s \in \mathbb{R}^d \text{ s.t. } \|s\| \leq 1, & \text{otherwise} \end{cases} \tag{10}$$

It can be verified that, when condition (6) is fulfilled, then the following choice of $x_i^\star$ and $s_{ij}^\star$ satisfy the first order optimality conditions in (9) and (10)

$$x_i^\star = w_k^\star, \ i \in C_k, \tag{11}$$

$$s_{ij}^\star = \begin{cases} r_{kl}^\star, & j \in C_l, k \neq l \\ \frac{\nabla f_j(w_k^\star) - \nabla f_i(w_k^\star)}{\lambda n_k}, & j \in C_k \end{cases}, \tag{12}$$

hence proving the result. $\qquad\square$

Theorem 1 guarantees the existence of a solution of (3) that exhibits the desired clustering structure. However, if the parameter λ is chosen too large, it can actually coarsen the clustering structure $C_1, \ldots, C_K$ and provide a solution with $1 \leq M < K$ groups (clusters). The following theorem ensure the correct clustering structure is recovered.

Theorem 2. *Consider problem* (5). *If for some node partition* $C_1, C_2, \ldots, C_K$ *and parameter* $\lambda > 0$ *there holds*

$$\lambda < \frac{\min_{k,l \in [K], k \neq l} \|\nabla g_k(w_k^\star) - \nabla g_l(w_k^\star)\|}{2 \max_{k \in [K]} \sum_{l \neq k} n_l}, \tag{13}$$

then for each $k, l \in [K], k \neq l$, *we have* $w_k^\star \neq w_l^\star$, *where* $w_k^\star = w_k^\star(\lambda)$, $k = 1, \ldots, K$, *is a solution to* (5), *defined for the same partition* $C_1, \ldots, C_K$, *that verifies* (13).

Proof of Theorem 2 can be found in [20].

IV. ALGORITHM FOR PERSONALIZED FEDERATED LEARNING

In this section, we introduce an algorithm to solve (3) in a federated server-users setting. The algorithm is adapted from the parallel direction method of multipliers (PDMM) in [18].

We start by reformulating problem (3) as follows:

$$\min \sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j \neq i} \|z_{ij}\| \tag{14}$$
$$\text{s.t. } x_i - x_j = z_{ij}, \ i \neq j.$$

That is, each of the $N(N-1)$ terms $\|x_i - x_j\|$ in (3) are replaced with $\|z_{ij}\|$, where $z_{ij} \in \mathbb{R}^d$ is an auxiliary (primal) variable. Then, for equivalence of (3) and (14), we add for each ordered pair (i, j), $i \neq j$, the constraint $x_i - x_j = z_{ij}$. Next, introduce the augmented Lagrangian $L : \mathbb{R}^{Nd} \times \mathbb{R}^{N(N-1)d} \times \mathbb{R}^{N(N-1)d} \to \mathbb{R}$, defined by:

$$L(\{x_i\}, \{z_{ij}\}, \{\mu_{ij}\}) = \sum_{i=1}^{N} f_i(x_i) + \lambda \sum_{j \neq i} \|z_{ij}\|$$
$$+ \sum_{j \neq i} \mu_{ij}^\top (x_i - x_j - z_{ij}) + \frac{\rho}{2} \sum_{j \neq i} \|x_i - x_j - z_{ij}\|^2, \tag{15}$$

where $\{x_i\}$, $i = 1, \ldots, N$, and $\{z_{ij}\}$, $i = 1, \ldots, N$, $j = 1, \ldots, N$, $i \neq j$, are the primal variables, and $\{\mu_{ij}\}$, $i = 1, \ldots, N$, $j = 1, \ldots, N$, $i \neq j$, are the dual variables, and $\rho > 0$ is a penalty parameter. Abstracting details, PDMM proceeds as follows. First, it updates at each iteration $t = 0, 1, \ldots$, a randomly selected subset of primal variables $\{x_i, z_{ij}\}$ by minimizing a surrogate of L with the rest of primal and dual variables fixed. Then, a randomly selected subset of dual variables $\{\mu_{ij}\}$ is updated, while also bookkeeping a set of auxiliary dual variables $\{\widehat{\mu}_{ij}\}$. See equations (29)–(31) in [22] for a detailed definition of the generic PDMM.

Here, we apply and adapt PDMM to solve (14), and hence, solve (3), in the federated server-users setting. To facilitate presentation of the algorithm, we enumerate all primal variables x_i's and z_{ij}'s through a common index set $\mathcal{S}_P$ with N^2 elements, such that the i-th element of $\mathcal{S}_P$, $i = 1, \ldots, N$, corresponds to x_i, and the remaining $N(N-1)$ subsequent elements correspond to z_{ij}'s, where the ordered pairs $\ell \sim (i, j)$ are positioned lexicographically in $\mathcal{S}_P$. For example, $(N+1)$-th element of $\mathcal{S}_P$ corresponds to variable z_{12}, $(N+2)$-nd element of $\mathcal{S}_P$ corresponds to z_{13}, etc. Similarly, we let $\mathcal{S}_D$ be the $N(N-1)$-sized index set, such that its ℓ-th element corresponds to the dual variable μ_{ij}, $\ell \sim (i, j)$, $\ell = 1, \ldots, N(N-1)$. The PDMM-based personalized FL method is shown in Algorithm 1.

Functions $B_i(\cdot, \cdot)$ in (16) and $B_{ij}(\cdot, \cdot)$ in (17) are instances of Bregman divergence, e.g., [18]; for example, they can be taken as $B(u, v) = \frac{1}{2}\|u - v\|^2$. The choice of functions $B_i(\cdot, \cdot)$ and $B_{ij}(\cdot, \cdot)$ also affect the computational cost of updates (16) and (17), respectively. For example, for $B_{ij}(u, v) = \frac{1}{2}\|u - v\|^2$, update (17) corresponds to evaluating a proximal operator of the 2-norm that is done via block soft-thresholding. See also subsection 2.1 in [18] for the choices of $B_i(\cdot, \cdot)$ that make update (16) computationally cheap. The positive parameters η_i in (16) and η_{ij} in (17) weigh the Bregman divergence terms;

It is possible to halve the number of constraints in (14) by imposing the constraint $x_i - x_j = z_{ij}$ only for $i < j$. This approach reduces the number of variables at the cost of additional coordination of users on the server's side. We present here the approach with the larger number of variables and less coordination required.

Algorithm 1 A PDMM-based algorithm for personalized FL and model clustering

---

for $t = 0, \ldots, T - 1$ **do**

(S1) The server randomly selects a subset $\mathcal{S}_P^{(t)} \subset \mathcal{S}_P$ of S_P, $S_P < N^2$, primal variables;

(S2) Each user $i \in \{1, \ldots, N\}$, such that $i \in \mathcal{S}_P^{(t)}$, performs the update of $x_i^{(t)}$ as follows:

$$x_i^{(t+1)} = \arg\min_{x_i \in \mathbb{R}^d} f_i(x_i) + \sum_{j \neq i} (\widehat{\mu}_{ij}^{(t)})^\top x_i$$
$$+ \quad \frac{\rho}{2} \sum_{j \neq i} \|x_i - x_j^{(t)} - z_{ij}^{(t)}\|^2 \qquad (16)$$
$$+ \quad \eta_i \, B_i(x_i, x_i^{(t)});$$

(S3) Each user $i \in \{1, \ldots, N\}$, such that $\ell \in \mathcal{S}_P^{(t)}$, $\ell \sim (i,j)$, performs the update of $z_{ij}^{(t)}$ as follows:

$$z_{ij}^{(t+1)} = \arg\min_{z_{ij} \in \mathbb{R}^d} \lambda \, \|z_{ij}\| - (\widehat{\mu}_{ij}^{(t)})^\top z_{ij}$$
$$+ \quad \frac{\rho}{2} \|x_i^{(t)} - x_j^{(t)} - z_{ij}\|^2 \qquad (17)$$
$$+ \quad \eta_{ij} \, B_{ij}(z_{ij}, z_{ij}^{(t)});$$

(S4) For each $s \in \{1, \ldots, N\}$, and ℓ, $\ell \sim (i,j)$, such that $s, \ell \notin \mathcal{S}_P^{(t)}$, set $x_s^{(t+1)} = x_s^{(t)}$, and $z_{ij}^{(t+1)} = z_{ij}^{(t)}$;

(S5) Each user $i \in \{1, \ldots, N\}$, such that $i \in \mathcal{S}_P^{(t)}$, sends $x_i^{(t+1)}$ to the server;

(S6) Each user $i \in \{1, \ldots, N\}$, such that $\ell \in \mathcal{S}_P^{(t)}$, $\ell \sim (i,j)$, sends $z_{ij}^{(t+1)}$ to the server;

(S7) The server collects $\{x_i^{(t+1)}\}$, $i \in \mathcal{S}_P^{(t)}$, and broadcasts this $(S_P\, d) \times 1$ vector to all users $i \notin \mathcal{S}_P^{(t)}$;

(S8) The server picks a random subset $\mathcal{S}_D^{(t)}$, $\mathcal{S}_D^{(t)} \subset \mathcal{S}_D$, of S_D dual variables, and performs the following update for $\ell \in \mathcal{S}_D$, $\ell \sim (i,j)$:

$$\mu_{ij}^{(t+1)} = \mu_{ij}^{(t)} \qquad (18)$$
$$+ \quad \tau\,\rho\,\left(x_i^{(t+1)} - x_j^{(t+1)} - z_{ij}^{(t+1)}\right);$$

(S9) The server sets $\mu_{ij}^{(t+1)} = \mu_{ij}^{(t)}$, for $\ell \notin \mathcal{S}_D^{(t)}$, $\ell \sim (i,j)$;

(S10) For each $\ell \in \mathcal{S}_D^{(t)}$, $\ell \sim (i,j)$, the server sends $\mu_{ij}^{(t+1)}$ to user i;

(S11) Each user i, such that $(i,j) \sim \ell$, $\ell \in \mathcal{S}_D^{(t)}$, performs the following update:

$$\widehat{\mu}_{ij}^{(t+1)} = \mu_{ij}^{(t+1)} \qquad (19)$$
$$- \quad \nu\,\rho\,\left(x_i^{(t+1)} - x_j^{(t+1)} - z_{ij}^{(t+1)}\right);$$

end for

---

the larger η_i is, the closer $x_i^{(t+1)}$ is to $x_i^{(t)}$, i.e., the smaller steps the algorithm makes. A similar effect is achieved with η_{ij} in (17). Quantity $\widehat{\mu}_{ij}^{(t)}$ in (19) is an auxiliary dual variable associated with the dual variable $\mu_{ij}^{(t)}$. The update step (19) is a backward dual step that is introduced for improving the algorithm's stability that may otherwise be violated due to the parallel and randomized nature of primal variable updates; see [18] for details. Similarly, parameters $\tau > 0$ and $\nu > 0$ in (18) and (19), respectively, are "damping" factors in dual variable updates, that are again used to stabilize the algorithm trajectory.

With Algorithm 1's initialization, we can set the $z_{ij}^{(0)}$'s, $\mu_{ij}^{(0)}$'s, and and $\widehat{\mu}_{ij}^{(0)}$'s arbitrarily. For example, they can be all set to zero. For the initialization of the x_i's, we need that $x_j^{(0)}$, $j \neq i$, is available at each user i. This can be achieved by, e.g., setting $x_i^{(0)} = 0$, for all i, or by letting the server send a common initial point $y^{(0)} \in \mathbb{R}^d$ to all users prior to the algorithm start, so that each user i sets $x_i^{(0)} = y^{(0)}$. The initial point $y^{(0)}$ may also be obtained by (approximately) solving (1) via a (non-personalized, standard) FL algorithm, e.g., FedAvg.

With Algorithm 1, the server maintains and updates the $N(N-1)$ $d \times 1$-sized dual variables $\mu_{ij}^{(t+1)}$, $i, j = 1, \ldots, N$, $i \neq j$. Each user i maintains and updates the $d \times 1$-sized primal variable $x_i^{(t)}$; $N - 1$ $d \times 1$-sized primal variables $z_{ij}^{(t)}$, $j \neq i$; and $N - 1$ $d \times 1$-sized auxiliary dual variables $\widehat{\mu}_{ij}^{(t)}$, $j \neq i$.

With Algorithm 1, communication from users to the server ("uplink") takes place at steps (S5) and (S6). Note that, at each t, during steps (S5) and (S6), the server receives exactly S_P $d \times 1$-sized (real vectors) messages. Here, S_P is the design parameter that can be taken to be much smaller than N, hence the uplink communication does not incur high overhead. Communication from the server to users ("downlink") takes place at steps (S7) and (S10). At step (S7), the server brodcasts S_D $d \times 1$-sized messages. At step (S10), the server transmits a total of S_D $d \times 1$-sized messages to different users. (Specifically, variable $\mu_{ij}^{(t+1)}$ is sent to user i, for $\ell \sim (i,j)$, $\ell \in \mathcal{S}_D^{(t)}$.) Therefore, the downlink communication involves a total of $(S_P + S_D)$ $d \times 1$-sized messages per iteration t. Quantity S_D is also a design parameter that can be set to be much smaller than N; hence, the downlink communication does not incur a significant communication overhead.

By applying Theorem 3 in [18] it can be shown that, under appropriately chosen tuning parameters, $\mathbb{E}\left[F(x^{(t)}) - F^\star\right] = O(1/t)$, where we recall that F is the objective function defined in (3), $F^\star = \inf_x F(x)$, and $x^{(t)} = ((x_1^{(t)})^\top, \ldots, (x_N^{(t)})^\top)^\top$ is generated by Algorithm 1.

V. NUMERICAL RESULTS

We now present numerical simulations. In the first set of experiments, we evaluate the cluster recovery abilities, as well as personalization and generalization abilities of the proposed formulation (3) and compare it with alternatives.

We now describe the first set of experiments. We consider a supervised binary classification problem. The generated data contains $K = 3$ clusters. For each cluster, for each given label/class (± 1), the data comes from a uniform distribution over an ellipse in $\mathbb{R}^2$. The two ellipses that correspond to

different classes for a given cluster overlap, so that the data in each cluster is not linearly separable. We generate 200 (training) data points from the distributions from each cluster, 100 points per class. Then, we associate to each cluster 20 FL users. Each FL user samples 10 data points out of the 200 data points available in its cluster. Hence, the data for all users within a cluster comes from the same distribution. See also Figure 1 in [20] for illustration of the data. The squared Hinge loss is used throughout the simulations, as the local loss function of each user i, and is given by

$$f_i(x) = \frac{c\|w\|^2}{2} + \frac{1}{m}\sum_{j=1}^{m} \max\left\{0, 1 - \ell_{ij}(\langle w, a_{ij}\rangle - b)\right\}^2,$$

where m represents the number of (local) data points, $(a_{ij}, \ell_{ij})_{j=1}^{m}$, represent the data points and class labels at user i, $c > 0$ is a penalty parameter that controls the regularization, while $x = [w, b] \in \mathbb{R}^{d+1}$, $d = 2$, represents the vector that defines the classifier. That is, the classifier based on vector $x = [w, b]$ takes a feature vector $a \in \mathbb{R}^2$ as input and predicts its label as $\ell = \text{sign}(\langle w, a\rangle - b)$. Throughout the experiments, we set the parameter $c = 10^{-3}$, in order to put more weight on the classification performance of the method.

We compare the proposed formulation (3) with the alternatives in (1), (2) and (4), referred to here as the global model, local models, and squared penalty, respectively. Formulation (1) corresponds to a standard, non-personalized FL solution. That is, the classifier vectors x_i's with (1) are equal for all users, i.e., $x_i = y^*$, where we recall that y^* is the solution to (1). With formulation (2), each user i's classifier vector equals $x_i = y_i^* = \arg\min f_i(x)$. In addition, we compare the proposed formulation (3) with an oracle model that knows beforehand the clustering structure of the users; then, for each user i within a cluster C_k, the oracle lets user i's classifier vector be $x_i = \arg\min_x \sum_{j \in C_k} f_k(x)$ We expect that the oracle model performs best in the considered setup among all methods, as it has an unfair advantage of knowing the cluster structure beforehand, and the data distributions of different clusters are very different, so data from a different cluster confuses another cluster's classifier. To evaluate solutions (1)-(4), we used CVXPY [23], [24].

In order to evaluate generalization and personalization abilities of different methods, we evaluate testing accuracy of the corresponding classifiers with respect to a newly generated test data. More precisely, let x_i be a classifier vector for user i obtained through training via any of the methods (1)-(4). For each user i, we then evaluate the testing accuracy of the classifier x_i with respect to the full testing data set for the cluster to which user i belongs to. We then average the testing accuracy across all users $i = 1, \ldots, N$. For each cluster, the testing data is generated by drawing new samples (new with respect to training data) from the same distributions according to which the training data is generated. Methods (3) or (4) then exhibit generalization if the average testing accuracy is above the average testing accuracy of local models; they exhibit personalization if their average testing accuracy is above that of the global model. The performance of the models, for different values of λ, is presented in Figure 1. Additionally, we

present the average Euclidean distance between the classifier vectors x_i's belonging to the same cluster. The results are summarized in Figure 2.
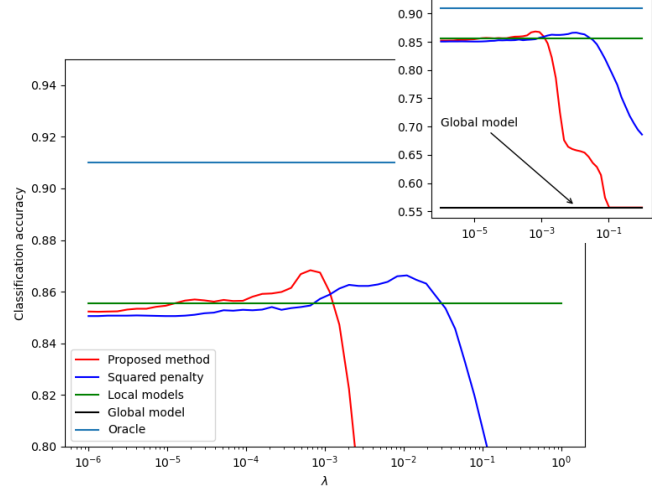

Fig. 1: The average classification accuracy across all users and all clusters. We can see that the two personalization methods achieve both personalization, as they outperform the global model, as well as generalization, as they outperform the strictly local model, for certain values of λ.
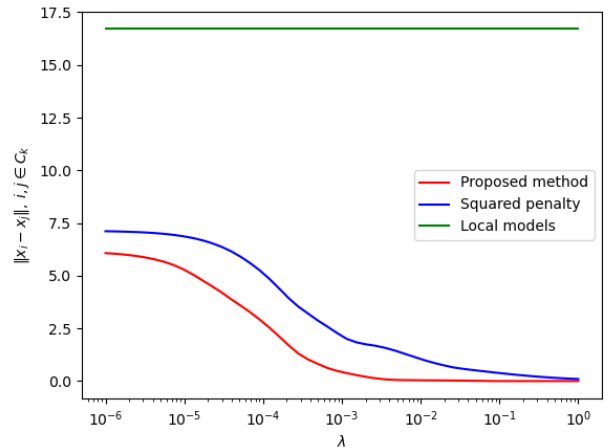

Fig. 2: The average distance between models across all users and all clusters. We can see that distance between models increases, as λ decreases, which is to be expected, as the models fit better to their local data. However, the proposed method results in more compact solutions within clusters, compared to the squared penalty one.

Figures 1 and 2 show the following. For λ sufficiently large, our method enforces consensus, and effectively performs as the standard FL method (1). For λ sufficiently small, the proposed method achieves both personalization, as it significantly outperforms the global model, as well as generalization, as the performance on the the complete cluster data is better than the strictly local models. Compared to the squared penalty model (4), we note that our method recovers the global model

for sufficiently large λ, while the squared penalty method can recover the global model only asymptotically, as λ tends to infinity. The highest average accuracy is achieved by our method, being at 86.8%, compared to the highest average accuracy of the squared penalty method, being at 86.6%. Figure 2 shows that our method constantly produces more compact clusters, i.e. the average distance between solutions within clusters is constantly smaller than the one produced by the squared penalty method .

Hence, we can see that the proposed formulation (3) achieves a comparable or slightly better peak accuracy with respect to (4), while producing more compact models, i.e., significantly reducing the number of distinct models that need to be kept in the overall FL system. We report that the proposed model (3) exactly recovers the cluster structure (produces equal user models within clusters and finds 3 clusters) for $\lambda \in (0.0892, 0.0919)$.

Finally, we evaluated the performance of PDMM for solving (3). This result, as well as some additional numerical simulations, can be found in [20].

VI. CONCLUSION

We proposed a novel formulation for personalized federated learning wherein we minimize the sum of local users' costs with respect to their local models, coupled with an additive term that penalizes the local models' differences via a sum-of-norms penalty. Future work directions include the following. First, while the convergence of the proposed solver follows from that of PDMM, it is of interest to study in more detail its computational and communication costs (complexity), for example in terms of properly defined oracles that determine how the primal variable "argmin" updates are carried out, e.g., in terms of the number of stochastic gradient descent iterations. Second, while the provided synthetic data experiments show merits of the proposed formulation, future work will include more elaborate experiments on real data sets, as well as further comparisons with other personalized FL approaches. Finally, towards practical implementations of the proposed approach, its management and deployment mechanisms will be developed.

REFERENCES

[1] 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*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[2] P. K. et al., "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, 2021.

[3] X. H. et al., "Advances and open problems in federated learning."

[4] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," *arXiv preprint arXiv:2002.04758*, 2020.

[5] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[6] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," 2021.

[7] A. Fallah, A. Mokhtari, and A. E. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[8] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," 2019.

[9] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[10] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4320–4328.

[11] I. Bistritz, A. Mann, and N. Bambos, "Distributed distillation for on-device learning," in *NeurIPS*, 2020.

[12] Y. J. Cho, J. Wang, T. Chiruvolu, and G. Joshi, "Personalized federated learning for heterogeneous clients with clustered knowledge transfer," *ArXiv*, vol. abs/2109.08119, 2021.

[13] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," 2021.

[14] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3710–3722, 2021.

[15] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *ArXiv*, vol. https://arxiv.org/abs/2002.10619, 2020.

[16] D. Sun, K.-C. Toh, and Y. Yuan, "Convex clustering: Model, theoretical guarantee and efficient algorithm," *Journal of Machine Learning Research*, vol. 22, 2021.

[17] F. Hanzely, S. Hanzely, S. Horvath, and P. Richtarik, "Lower bounds and optimal algorithms for personalized federated learning," in *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020.

[18] H. Wang, A. Banerjee, and Z.-Q. Luo, "Parallel direction method of multipliers," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[19] W. Ben-Ameur, P. Bianchi, and J. Jakubowicz, "Robust distributed consensus using total variation," *IEEE Trans. Aut. Contr.*, vol. 61, no. 6, 2016.

[20] A. Armacki, D. Bajovic, D. Jakovetic, and S. Kar, "Personalized federated learning via convex clustering," *arXiv preprint, available at: https://arxiv.org/abs/2202.00718*, 2022.

[21] A. Panahi, D. Dubhashi, F. D. Johansson, and C. Bhattacharyya, "Clustering by sum of norms: Stochastic incremental algorithm, convergence and cluster recovery," in *Proceedings of the 34th International Conference on Machine Learning, PMLR 70*, 2017, pp. 2769–2777.

[22] M. Hong, M. Razaviyayn, Z.-Q. Luo, and J.-S. Pang, "A unified algorithmic framework for block-structured optimization involving big data: With applications in machine learning and signal processing," *IEEE Sig. Proc. Mag.*, vol. 33, no. 1, pp. 57–77, 2016.

[23] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[24] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.

Note that the considered simulation setup is such that the inter cluster generalization does not help. That is, data distributions across different clusters are so far apart, that the oracle, that ignores the data from other clusters, works best. Clearly, if the data distributions across different clusters are very close, another oracle that exploits inter-cluster generalization may be considered. Consider the extreme case when the data distributions across the three clusters are mutually (almost) equal, i.e., the distribution difference is negligible, but this is not known beforehand. Then, clearly, the global model (1) becomes best-performing and an "oracle" model, in the sense that it implicitly utilizes the unknown cluster structure. Our formulation (3) can uncover this and actually match the global model for λ above a threshold, and hence perform optimally. On the other hand, (4) gets closer to the global model as λ increases, but never matches it. This illustration presents a scenario when there is clear merit in across-cluster generalization that the proposed formulation (3) is able to harness.