A Unifying Framework for Federated Learning

Saber Malekmohammadi, Kiarash Shaloudegi, Zeou Hu, and Yaoliang Yu

Abstract There have been multiple federated learning (FL) algorithms proposed in the FL community during the recent years. However, a thorough comparison of these algorithms has not been done, and our understanding of the theory of FL is still limited. The lack of a unifying view in practice has also led to the reinvention of the same algorithms under different names. Motivated by this gap, we develop a unifying scheme for FL and demonstrate that many of the algorithms that exist in the FL literature are special cases of this scheme. The unification allows us to get a deeper understanding of different FL algorithms, to compare them easier, to improve the previous results for their convergence analysis and to find new FL algorithms. In particular, we demonstrate the important role that step size plays in the convergence of FL algorithms. Further, based on our unifying scheme, we propose an efficient and economic method for accelerating FL algorithms. This streamlined acceleration method does not incur any communication overheads. We evaluate our findings by performing extensive experiments on both nonconvex and convex problems.

1 Introduction

Federated Learning (FL) is a massively distributed framework consisting of some users and the data distributed among them. This framework enables training of a shared or personalized model based on the users private data. Since the work of

S. Malekmohammadi (✉) · Z. Hu · Y. Yu
School of Computer Science, University of Waterloo, 200 University Ave West, Waterloo, ON, Canada
e-mail: saber.malekmohammadi@uwaterloo.ca

Z. Hu
e-mail: zeou.hu@uwaterloo.ca

Y. Yu
e-mail: yaoliang.yu@uwaterloo.ca

K. Shaloudegi
Noah's Ark Lab, Huawei Technologies, Montreal, QC, Canada
e-mail: kiarash.shaloudegi@gmail.com

R. Razavi-Far et al. (eds.), *Federated and Transfer Learning*, Adaptation, Learning, and Optimization 27, https://doi.org/10.1007/978-3-031-11748-0_5

[1], much progress has been made in different ways, including proposing novel algorithms [2–8], convergence analysis [9–13], fairness [14–16], privacy protection [17, 18], personalization [19–23], model robustness [24–27], standardization [28, 29] and applications [30], just to name a few. A summary of the current state of the art in FL can be found in [31–33].

In this work, we study some FL algorithms, which are among the most popular algorithms in the community, including FedAvg [1], FedSplit [5] and FedProx [2]. By relating these algorithms to the operator splitting theory in optimization, we get a deeper understanding of them. In particular, FedAvg belongs to forward-backward splitting. Also, there is a trade-off between the number of local epochs and the step size in FedAvg. A similar observation has also been reported in [5, 12, 34]. Similarly, we show that FedProx belongs to backward-backward splitting. Interestingly, FedProx also belongs to forward-backward splitting applied to a regularized problem. We also show that FedProx can find the exact global solution when the step size diminishes by time (sublinearly fast). This new understanding is against the observation in [5] for a constant step size. We also show that the convergence of FedSplit, which corresponds to Peaceman-Rachford splitting [35, 36], heavily depends on the objective functions being strongly convex. Hence, FedSplit is less stable for nonconvex problems.

Adapting the existing FL algorithms and connecting them with operator splitting theory allows us to propose new algorithms. Based on Douglas-Rachford splitting [36, 37] (more specifically the partial inverse method [38]), we propose a new algorithm FedPi, which is essentially a more stable (but slower) variant of FedSplit. Also, by combining the projection (averaging) in FedAvg and FedProx with the reflector in FedSplit, we propose a new algorithm FedRP, which essentially extends the algorithm of [39]. We analyze the convergence of FedRP and show with empirical experiments that it is indeed competitive with the other FL algorithms.

As an important contribution, we propose a unifying framework for FL algorithms. We show that the aforementioned FL algorithms are special cases of a unifying grand scheme, which is an important progress for standardizing FL algorithms. Furthermore, our proposed unification also allows us to accelerate different FL algorithms in an efficient way without incurring any overhead. We use Anderson-acceleration, which was proposed in [40, 41] for nonlinear fixed-point iteration, for the first time to accelerate existing FL algorithms. We also show how to implement the acceleration practically without incurring any overhead. To validate our findings, we perform extensive experiments on both nonconvex and convex problems, and compare the previously mentioned FL algorithms.

We proceed as follows. We introduce the background and some basic tools in Sect. 2. In Sect. 3, we report our main contributions: We draw a connection between FL and operator splitting theory, provide new understandings about some of the well-known algorithms, refine their convergence analysis, show some experimental results and propose a new algorithm. In Sect. 4, we unify the discussed algorithms into a grand scheme and based on that, we propose an economic way for accelerating them. We finally conclude in Sect. 5.

2 Background

We consider a federated learning (FL) setting with m users, where user i has an objective function $f_i : \mathbb{R}^d \to \mathbb{R}$, and $\mathbf{w} \in \mathbb{R}^d$ is the model parameter shared between the users. The objective function f_i of user i is computed on the corresponding user's private data ($\mathscr{D}_i$). In FL, the goal is to *efficiently and collectively* optimize users objective functions $\{f_i\}$ in a decentralized way. The optimization should also be privacy-preserving and with low communication overhead. Many existing FL algorithms, e.g. FedAvg [1], optimize the (arithmetic) average performance:

$$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}), \quad \text{where} \quad f(\mathbf{w}) := \sum_{i=1}^{m} \lambda_i f_i(\mathbf{w}). \tag{1}$$

In this problem, λ_i is the nonnegative weight corresponding to user i. We assume these weights are determined *beforehand* and sum to 1. For a related algorithm where the weights λ_i are adapted in each iteration, see [16].

The proximal map and Moreau envelope of a function[1] $f : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$, are respectively, defined as:

$$\begin{aligned} \mathsf{P}_f^{\eta}(\mathbf{w}) &= \operatorname*{argmin}_{\mathbf{z}} \ \tfrac{1}{2\eta}\|\mathbf{z}-\mathbf{w}\|_2^2 + f(\mathbf{z}), \\ \mathsf{M}_f^{\eta}(\mathbf{w}) &= \min_{\mathbf{z}} \ \tfrac{1}{2\eta}\|\mathbf{z}-\mathbf{w}\|_2^2 + f(\mathbf{z}). \end{aligned} \tag{2}$$

The parameter $\eta > 0$ acts as the step size. We note that the Moreau envelope M_f^{η} is a real-valued function while the proximal map $\mathsf{P}_f^{\eta} : \mathbb{R}^d \rightrightarrows \mathbb{R}^d$ is a vector-valued map that may take multiple values at a given input. When f is lower bounded by some quadratic function[2] and η is sufficiently small, the Moreau envelope M_f^{η} is finite-valued, whereas the proximal map P_f^{η} is nonempty-valued if f is additionally lower semicontinuous. For convex functions, P_f^{η} is always a singleton (for any $\eta > 0$) while it may take multiple values when f is nonconvex. The Moreau envelope (and its related variants) have long been used as a smoothing device to (approximately) turn a nonsmooth function into a smooth one, see e.g. [43]. Based on (2), we use the reflector operator as:

$$\mathsf{R}_f^{\eta}(\mathbf{w}) = 2\mathsf{P}_f^{\eta}(\mathbf{w}) - \mathbf{w}. \tag{3}$$

The following theorem, due to [44], reveals the conditions under which a single-valued map $\mathsf{P} : \mathbb{R}^d \to \mathbb{R}^d$ coincides with the proximal map P_f^{η} of some convex function f.

[1] If the input to the function f is not in our domain of interest, we allow f to take ∞ value.

[2] Such functions are called prox bounded, see [42, Definition 1.23].

Theorem 1 ([44]) *Let* $\mathsf{P} : \mathbb{R}^d \to \mathbb{R}^d$ *be an everywhere single-valued map. Fix any* $\eta > 0$. *Then,* $\mathsf{P} = \mathsf{P}_f^\eta$ *for some convex function* f *iff* P *is nonexpansive and* $\mathsf{P} = \nabla g$ *for some differentiable convex function* g. *Moreover,* $\mathsf{P}_f^\eta = \mathsf{P}_h^\eta$ *iff* $f = h + c$ *for some constant* c.

Recall that a map $\mathsf{P} : \mathbb{R}^d \to \mathbb{R}^d$ is nonexpansive (w.r.t. the ℓ_2 norm) if for all $\mathbf{x}, \mathbf{y}$:

$$\|\mathsf{P}(\mathbf{x}) - \mathsf{P}(\mathbf{y})\|_2 \leq \|\mathbf{x} - \mathbf{y}\|_2.$$

When $d = 1$, the condition $\mathsf{P} = \nabla g$ reduces to the usual (increasing) monotonicity while for $d \geq 2$ it is known as maximal cyclic monotonicity [42].

We reformulate the problem (1) as an inner product [38]:

$$\min_{\mathsf{w} \in H} \mathsf{f}(\mathsf{w}) = \langle \mathbf{1}, \mathbf{f}(\mathsf{w}) \rangle . \tag{4}$$

In the above inner product, $\mathbf{f}(\mathsf{w}) := \big(f_1(\mathbf{w}_1), \ldots, f_m(\mathbf{w}_m)\big)$ is the vector of local objective function values. $\mathbf{1}$ is a vector with its all elements equal to 1,

$$H := \{\mathsf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_m) \in \mathbb{R}^{dm} : \ \mathbf{w}_1 = \cdots = \mathbf{w}_m\}. \tag{5}$$

The reformulation (4) is based on the following inner product, which we use in our notations.

$$\langle \mathsf{w}, \mathsf{z} \rangle := \sum_i \lambda_i \mathbf{w}_i^\top \mathbf{z}_i. \tag{6}$$

We remark that we have incorporated the users corresponding weights λ_i into the inner product. Based on the product (6), we have the following complement for the equality subspace H defined above:

$$H^\perp := \{\mathsf{w} : \langle \mathsf{w}, \mathsf{z} \rangle = 0, \forall \mathsf{z} \in H\} = \{\mathsf{w} : \sum_i \lambda_i \mathbf{w}_i = \mathbf{0}\}. \tag{7}$$

Also, we define the projection and reflection operators:

$$\mathsf{P}_H(\mathsf{w}) = (\bar{\mathbf{w}}, \ldots, \bar{\mathbf{w}}), \tag{8}$$

$$\mathsf{R}_H(\mathsf{w}) = (2\bar{\mathbf{w}} - \mathbf{w}_1, \ldots, 2\bar{\mathbf{w}} - \mathbf{w}_m). \tag{9}$$

In both of the projection and reflection operators, $\bar{\mathbf{w}} := \sum_i \lambda_i \mathbf{w}_i$ is the weighted average of $\{\mathbf{w}_i\}_{i=1}^m$. Given a (sub)differentiable function f, the (forward) gradient update map w.r.t. f is defined as:

$$\mathsf{G}_f^\eta := \mathrm{id} - \eta \cdot \partial f. \tag{10}$$

Note that for a convex and differentiable function f, we have:

$$\nabla \mathsf{M}_f^\eta = \frac{\mathrm{id} - \mathsf{P}_f^\eta}{\eta} \quad \text{and} \quad \mathsf{R}_f^\eta = \mathsf{G}_f^\eta \circ \mathsf{P}_f^\eta. \tag{11}$$

We also introduce another technical tool, the proximal average [44, 45], which we use to get a deeper understanding of `FedAvg`. Given a set of positive weights $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$ and a set of functions $\mathbf{f} = (f_1, \ldots, f_m)$, the so-called proximal average function $\mathsf{A} = \mathsf{A}_{\mathbf{f}}^\eta$ is given implicitly by:

$$\mathsf{P}_{\mathsf{A}}^\eta = \left\langle \mathbf{1}, \mathsf{P}_{f_i}^\eta \right\rangle = \sum_{i=1}^m \lambda_i \mathsf{P}_{f_i}^\eta, \tag{12}$$

i.e., the proximal map of the implicit proximal average is equal to the weighted average (with the weights $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$) of proximal maps. When $\{f_i\}$ are convex, the existence and uniqueness (up to addition of a constant) of the proximal average can be shown [45].

Corollary 1 *Given a set of positive weights $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$ and convex functions $\mathbf{f} = (f_1, \ldots, f_m)$, the proximal average function $\mathsf{P}_{\mathsf{A}}^\eta$ exists and is unique (up to addition of a constant).*

Proof The existence and uniqueness of the proximal average immediately follow from Theorem 1. Indeed, invoking the only if part in Theorem 1, the right-hand side of (12) is easily verified to be nonexpansive and each $\mathsf{P}_{f_i}^\eta = \nabla g_i$ for some convex function g_i, hence $\sum_i \lambda_i \mathsf{P}_{f_i}^\eta = \sum_i \lambda_i \nabla g_i = \nabla(\sum_i \lambda_i g_i)$. The if part in Theorem 1 then implies the existence of the proximal average function A whose uniqueness (up to addition of constant) also follows from Theorem 1. □

3 Federated Learning as Operator Splitting

We take a close examination of existing `FL` algorithms (`FedAvg`, `FedProx`, `FedSplit`) in this section. In details, we provide some new understandings of them, reveal the important effect of step size on their convergence, obtain new guarantees for their convergence, and propose some new `FL` and acceleration algorithms. We also perform numerical experiments to validate our theoretical developments, with two convex experiments on least squares and logistic regression, as well as two non-convex image classification experiments on CIFAR-10 and MNIST datasets with convolution neural networks (CNNs).

3.1 Experimental Setups

Before proceeding to examining the FL algorithms, let us first specify the experimental details. We follow [5] to simulate the convex problems of least squares and logistic regression.

Experimental setup for least squares regression

In this convex FL problem, we consider m users with user i having $f_i(\mathbf{w}) := \frac{1}{2}\|A_i\mathbf{w} - \mathbf{b}_i\|_2^2$ as its local objective function. We need to solve the convex problem:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) := \sum_{i=1}^{m} f_i(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m} \|A_i\mathbf{w} - \mathbf{b}_i\|_2^2.$$

The optimization variable is $\mathbf{w} \in \mathbb{R}^d$. The matrix $A_i \in \mathbb{R}^{n_i\times d}$ and the vector $\mathbf{b}_i \in \mathbb{R}^{n_i}$ are related as follows:

$$\mathbf{b}_i = A_i\mathbf{w}_\star + \boldsymbol{\varepsilon}_i$$

In this linear model, $\boldsymbol{\varepsilon}_i$ is the noise vector and each of its elements are sampled from the normal distribution $N(0, \sigma^2)$ ($\sigma > 0$). Also, we sample the elements of $A_i \in \mathbb{R}^{n_i\times d}$ independently: $A_i^{k,l} \sim N(0, 1)$. We used the following parameters to instantiate the problem:

$$d = 100, \quad m = 25, \quad n_i = 5000, \quad \sigma^2 = 0.25.$$

In our experiments, $\eta = 10^{-5}$ is used as the learning rate for local updates. Also, FedAvg is tun with the number of local epochs $k = 5$.

Experimental setup for binary logistic regression

Similar to the least-squares problem, we consider m users with user i having the matrix $A_i \in \mathbb{R}^{n_i\times d}$. We generate the matrices A_i similar to what was done in the least-squares experiment. The vector $\mathbf{b}_i \in \{-1, 1\}^{n_i}$ is the label vector corresponding to the user i. In logistic regression, given a parameter vector $\mathbf{w}_\star$, the probability of $b_{ij} = 1$ is

$$\mathbf{P}\{b_{ij} = 1\} = \frac{e^{\mathbf{a}_{ij}^\top\mathbf{w}_\star}}{1 + e^{\mathbf{a}_{ij}^\top\mathbf{w}_\star}}, \quad j = 1, \ldots, n_i.$$

The j-th row of A_i is denoted by $\mathbf{a}_{ij}$. We also sample the elements of $\mathbf{w}_\star \in \mathbb{R}^d$ from $N(0, 1)$. When A_i and $\mathbf{b}_i$ are generated for all users, we need to find the unique solution of the following convex program to get the maximum likelihood estimate of $\mathbf{w}_\star$:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) := \sum_{i=1}^{m} f_i(\mathbf{w}) = \sum_{i=1}^{m}\sum_{j=1}^{n_i} \log(1 + e^{-b_{ij}\mathbf{a}_{ij}^\top \mathbf{w}}) + \frac{\|\mathbf{w}\|_2^2}{2mn_i}.$$

Following [5], we set

$$m = 10, \quad d = 100, \quad n_i = 1000.$$

We use local learning rate $\eta = 10^{-2}$, and run `FedAvg` with $k = 5$.

Data heterogeneity measure

We use the measure used in [11] to measure the data heterogeneity in the convex problems of logistic regression and least squares. More specifically, we use the following measure:

$$\mathfrak{H} := \frac{1}{m}\sum_{i=1}^{m} \| \nabla f_i(\mathbf{w}_*) \|_2^2 .$$

The gradients of the local objective functions are computed at $\mathbf{w}_*$, which is a solution of (1). The more homogeneous the users' data is, the smaller the measure becomes. When all users' data are completely homogeneous, the minimizers of the local objective functions will be the same and $\mathfrak{H} = 0$.

Experimental setup for image classification on MNIST dataset

For this experiment, we consider 20 users in a distributed setting. We follow [1] to create a non-IID dataset: As the first step, the data of each class gets divided into some shards. Then, we assign a number of shards to each user randomly. For instance, in Fig. 6 to prevent users from receiving more than 6 classes of the MNIST dataset, we split each class into 12 shards (120 total shards), and then we randomly assign 6 shards of data to each user. With 20 users, this makes the data distribution of users different and produces balanced datasets–all having the same number of data points. Finally, we divide each user's allocated data to train (80%), validation (10%), and

Table 1 Details of the convolutional neural network model for MNIST dataset

Layer	Shape of output	# of parameters	Activation	Hyper-parameters
Input	(1, 28, 28)	0		
Conv2d	(10, 24, 24)	260	ReLU	kernel size = 5; strides = (1, 1)
MaxPool2d	(10, 12, 12)	0		pool size = (2, 2)
Conv2d	(20, 8, 8)	5,020	ReLU	kernel size = 5; strides = (1, 1)
MaxPool2d	(20, 4, 4)	0		pool size = (2, 2)
Flatten	320	0		
Dense	20	6,420	ReLU	
Dense	10	210	softmax	
Total		11,910		

test (10%) sets. This results in each user having 2400, 300, and 300 data points for training, validation and testing, respectively. A CNN model with 2 layers and ReLU activation function is used as the model. The details of the used model is reported in Table 1. In `FedAvg`, each user uses gradient descent (default value $\eta = 0.01$) for updating its local model. Also, for proximal updates in the splitting algorithms, we use gradient descent and $k = 100$ (the number of epochs) and $\eta = 0.01$ (learning rate) for solving the related optimization problem.

Experimental setup for image classification on CIFAR-10 dataset

In this experiment, we consider 10 users in a distributed setting. Also, we use a dataset which is subsampled from the original CIFAR-10: we subsample 20% of the original training set. For creating a non-IID dataset, we follow [1]: We first use labels to sort all data points. Then, they are split into 100 shards, and 10 shards are assigned to each user randomly. We split each user local data into train (80%), validation (10%), and test (10%) sets. This splitting results in 800, 100 and 100 data points for train set, validation set, and test set of each user, respectively. We use a CNN model with 2 layers. The details of the model is reported in Table 2. Each user uses its own data to update its local model by applying stochastic gradient descent (SGD) with $\eta = 0.1$ (local learning rate) and $B = 20$ (local batch size). We also assume full participation of all users in each communication round.

Table 2 Details for the convolutional neural network model on CIFAR-10

Layer	Shape of output	# of parameters	Activation	Hyper-parameters
Input	(3, 32, 32)	0		
Conv2d	(64, 28, 28)	4,864	ReLU	kernel = 5; strides = (1, 1)
MaxPool2d	(64, 14, 14)	0		pool size = (2, 2)
LocalResponseNorm	(64, 14, 14)	0		size = 2
Conv2d	(64, 10, 10)	102,464	ReLU	kernel = 5; strides = (1, 1)
LocalResponseNorm	(64, 10, 10)	0		size = 2
MaxPool2d	(64, 5, 5)	0		pool size = (2, 2)
Flatten	1,600	0		
Dense	384	614,784	ReLU	
Dense	192	73,920	ReLU	
Dense	10	1,930	softmax	
Total		797,962		

3.2 `FedAvg` as Forward-Backward Splitting

The `FedAvg` algorithm was the seminal work in `FL` and got proposed in [1]:

$$\mathsf{w}_{t+1} \leftarrow \mathsf{P}_H \mathsf{G}_{\mathsf{f},k}^{\eta_t} \mathsf{w}_t, \quad \mathsf{G}_{\mathsf{f},k}^{\eta_t} := \underbrace{\mathsf{G}_{\mathsf{f}}^{\eta_t} \circ \mathsf{G}_{\mathsf{f}}^{\eta_t} \circ \cdots \circ \mathsf{G}_{\mathsf{f}}^{\eta_t}}_{k \text{ times}}. \tag{13}$$

In each communication round, `FedAvg` performs k gradient updates (forward), and then performs 1 proximal update w.r.t. H (backward). Hence, `FedAvg` belongs to forward-backward splitting [46] and is a k-step version of it. To improve efficiency, [1] also replaced $\mathsf{G}_{\mathsf{f}}^{\eta_t}$ with $\hat{\mathsf{G}}_{\mathsf{f}}^{\eta_t}$, to approximate (sub)gradient on a minibatch of training data. Reference [1] also replaced P_H with $\hat{\mathsf{P}}_H$: the averaging is done only over a chosen subset of users. Note that users perform $\mathsf{G}_{\mathsf{f},k}^{\eta_t}$ (the forward step) in parallel, and the server performs P_H (the backward step) afterwards.

Now, we use the technical tool proximal average to reveal a deeper insight of `FedAvg`. Recall that [45] proved that

$$\mathsf{A}_{\mathbf{f}}^{0}(\mathbf{w}) := \lim_{\eta \to 0^+} \mathsf{A}_{\mathbf{f}}^{\eta}(\mathbf{w}) = \langle \mathbf{1}, \mathbf{f}(\mathbf{w}) \rangle, \tag{14}$$

$$\mathsf{A}_{\mathbf{f}}^{\infty}(\mathbf{w}) := \lim_{\eta \to \infty} \mathsf{A}_{\mathbf{f}}^{\eta}(\mathbf{w}) = \min_{\mathbf{w}_1, \ldots, \mathbf{w}_m} \langle \mathbf{1}, \mathbf{f}(\mathbf{w}) \rangle, \text{ s.t. } \sum_i \lambda_i \mathbf{w}_i = \mathbf{w}, \tag{15}$$

and that $\mathsf{A}_{\mathbf{f}}^{0}(\mathbf{w}) \geq \mathsf{A}_{\mathbf{f}}^{\infty}(\mathbf{w})$. We then have the following theorem about `FedAvg`:

Theorem 2 *`FedAvg` with $k = 1$ amounts to minimizing $\mathsf{A}_{\mathbf{f}}^{0}(\mathbf{w})$ while `FedAvg` with $k = \infty$ amounts to minimizing $\mathsf{A}_{\mathbf{f}}^{\infty}(\mathbf{w})$.*

Proof To see the latter, let $f_i^* = f_i(\mathbf{w}_i^*) = \min_{\mathbf{w}_i} f_i(\mathbf{w}_i)$. Therefore,

$$\sum_i \lambda_i f_i(\mathbf{w}_i) \geq \sum_i \lambda_i f_i(\mathbf{w}_i^*) \geq \mathsf{A}_{\mathbf{f},\boldsymbol{\lambda}}^{\infty}(\mathbf{w}^*),$$

where $\mathbf{w}^* := \sum_i \lambda_i \mathbf{w}_i^*$. Taking minimum over $\{\mathbf{w}_i\}$ (such that $\mathbf{w} = \sum_i \lambda_i \mathbf{w}_i$) on both sides we obtain

$$\mathsf{A}_{\mathbf{f},\boldsymbol{\lambda}}^{\infty}(\mathbf{w}) \geq \mathsf{A}_{\mathbf{f},\boldsymbol{\lambda}}^{\infty}(\mathbf{w}^*),$$

i.e., the minimizer of $\mathsf{A}_{\mathbf{f},\boldsymbol{\lambda}}^{\infty}(\mathbf{w})$ is the average of minimizers $\{\mathbf{w}_i^*\}$ of f_i's. □

Least-squares example for the proximal average

Let us illustrate the proximal average with the least-squares problem, where $f_i(\mathbf{w}) = \frac{1}{2}\|A_i\mathbf{w} - \mathbf{b}_i\|_2^2$. We first recall that the *resolvent average* [47] of a set of positive definite matrices $\mathbf{Q} = (Q_1, \ldots, Q_m)$ is given by:

$$\mathsf{R}_{\mathbf{Q}}^{\eta} := \left\langle \mathbf{1}, \mathbf{Q}^{-1} + \eta I \right\rangle^{-1} - \eta^{-1} I, \tag{16}$$

where $\mathbf{Q}^{-1} := (Q_1^{-1}, Q_2^{-1}, \ldots, Q_m^{-1})$. In addition, the following results are proven in [47]:

$$\lim_{\eta \to 0^+} \mathsf{R}_{\mathbf{Q}}^{\eta} = \langle \mathbf{1}, \mathbf{Q} \rangle, \tag{17}$$

$$\lim_{\eta \to \infty} \mathsf{R}_{\mathbf{Q}}^{\eta} = \left\langle \mathbf{1}, \mathbf{Q}^{-1} \right\rangle^{-1}. \tag{18}$$

Then, for general convex-quadratic functions of the form $f_i(\mathbf{w}) := \frac{1}{2}\mathbf{w}^\top Q_i \mathbf{w} + \mathbf{c}_i^\top \mathbf{w} + r_i$, their proximal average $\mathsf{A}_{\mathbf{f}}^{\eta}$ can be derived in a closed-form expression.

Proposition 1 (proximal average of quadratic functions, [47]) *Let $Q_i \in \mathbb{S}_+^d$ (i.e., symmetric positive definite), $\mathbf{c}_i \in \mathbb{R}^d$, and $r_i \in \mathbb{R}$, and $f_i(\mathbf{w}) := \frac{1}{2}\mathbf{w}^\top Q_i \mathbf{w} + \mathbf{c}_i^\top \mathbf{w} + r_i$ for $i \in \{1, \ldots, m\}$, then the proximal average $\mathsf{A}_{\mathbf{f}}^{\eta}$ is given by*

$$\mathsf{A}_{\mathbf{f}}^{\eta}(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathsf{R}_{\mathbf{Q}}^{\eta} \mathbf{w} + \mathbf{w}^\top \left(\sum_{i=1}^{m} \lambda_i (Q_i + \eta^{-1} I)^{-1} \right)^{-1} \sum_{i=1}^{m} \lambda_i (Q_i + \eta^{-1} I)^{-1} \mathbf{c}_i + C,$$

where C is some constant that does not depend on $\mathbf{w}$.

Now returning to the least squares problem, we have $Q_i = A_i^\top A_i$, $\mathbf{c}_i = -A_i^\top \mathbf{b}_i$, and $r_i = \mathbf{b}_i^\top \mathbf{b}_i$. Given these, we apply Proposition 1, together with the two limits in (17) and (18), to immediately deduce:

$$\left[\operatorname*{argmin}_{\mathbf{w}} \mathsf{A}_{\mathbf{f}}^0(\mathbf{w})\right] = \left(\sum_{i=1}^m \lambda_i A_i^\top A_i\right)^{-1} \sum_{i=1}^m \lambda_i A_i^\top \mathbf{b}_i, \tag{19}$$

$$\left[\operatorname*{argmin}_{\mathbf{w}} \mathsf{A}_{\mathbf{f}}^\infty(\mathbf{w})\right] = \sum_{i=1}^m \lambda_i (A_i^\top A_i)^{-1} A_i^\top \mathbf{b}_i, \tag{20}$$

which is consistent with Theorem 2, as well as the closed form obtained by [5] for the fixed-point solution of `FedAvg` for $k = 1$ and $k = \infty$, respectively. Theorem 2 is a motivation for us to state that `FedAvg` with an intermediate value of k (local epochs) minimizes the proximal average $\mathsf{A}_{\mathbf{f}}^\eta$ with an intermediate η. Interestingly, it can be shown that the difference between the arithmetic average $\mathsf{A}_{\mathbf{f}}^0$ and the proximal average $\mathsf{A}_{\mathbf{f}}^\eta$ is bounded *uniformly*, using the Lipschitz constants of the functions f_i [48]. Therefore, if the step size η is small, `FedAvg` (with any finite k) can be interpreted as minimizing some *approximation* of (1).

Another important observation about `FedAvg` with an intermediate k is that in this case, the quality of `FedAvg` final solution appears to have a dependency on k and η. Let us illustrate this again on the least-squares problem, with $f_i(\mathbf{w}) = \frac{1}{2}\|A_i\mathbf{w} - \mathbf{b}_i\|_2^2$. When η is fixed and the weights λ_i are uniform ($\lambda_i \equiv \frac{1}{m}$), the final solution of `FedAvg` can be found in closed-form [5]:

$$\mathbf{w}^*_{\texttt{FedAvg}}(k) = \left(\sum_{i=1}^m \frac{1}{k}\sum_{j=0}^{k-1} A_i^\top A_i (I - \eta A_i^\top A_i)^j\right)^{-1} \left(\sum_{i=1}^m \frac{1}{k}\sum_{j=0}^{k-1} (I - \eta A_i^\top A_i)^j A_i^\top \mathbf{b}_i\right). \tag{21}$$

When η is small, we can use Taylor expansion and drop the terms of higher order:

$$\frac{1}{k}\sum_{j=0}^{k-1}(I - \eta A_i^\top A_i)^j \approx \frac{1}{k}\sum_{j=0}^{k-1}(I - j\eta A_i^\top A_i) = I - \frac{\eta(k-1)}{2} A_i^\top A_i. \tag{22}$$

From (22), we observe that, when k has an intermediate value, the final solution of `FedAvg` depends on $\eta(k-1)$. However, when $k = 1$, the final solution (fixed point) of `FedAvg` is independent of η (as long as convergence of `FedAvg` is guaranteed with small enough η) and `FedAvg` converges to the correct solution. However, when $k > 1$, final solution of `FedAvg` does depend on η, and its quality is determined by $\eta(k-1)$ completely (when η is small). This is verified in our experiments.

Figure 1 shows the experimental results where we use `FedAvg` for the convex problems of logistic regression and least squares and a nonconvex CNN model [49]. We have evaluated `FedAvg` based on the optimality gap ($f(\mathbf{w}^*_{\texttt{FedAvg}}) - f^*$) of its

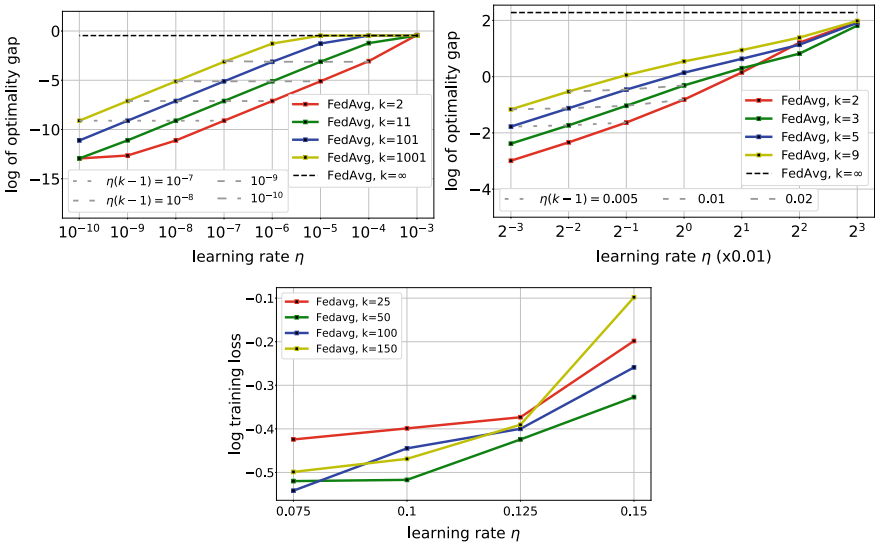

Fig. 1 The optimality gap or training loss of final solutions of `FedAvg`. Different numbers of local epochs (k) are shown with different colored lines, and different product values $\eta(k-1)$ are shown with dashed lines. Top left: `FedAvg` closed-form solution for least squares; Top right: `FedAvg` solution after 6000 communication rounds for logistic regression; Bottom: `FedAvg` solution after 200 communication rounds for nonconvex CNN on MNIST

solution in the convex settings and the training loss of its solution in the nonconvex settings. Please refer to Sect. 3.1 for detailed experimental setups. The experiments[3] are run with different values of k and η and a pre-determined number of communication rounds. We investigate the effect of k and η on the quality of `FedAvg` final solution. As can be understood from Fig. 1, in general, better final solutions are found with smaller values of k and η (assuming convergence is achieved with sufficient communication rounds). Moreover, when k and η have moderate values, the product $\eta(k-1)$, as shown in (22), determines the quality of the `FedAvg` final solution for logistic regression and least squares. However, for the nonconvex CNN (Fig. 1, right), the Taylor approximation that we used before is too crude to be useful, especially when k is relatively small (i.e. limited communications).

Figure 2 further illustrates this trade-off between η and k. As can be observed, larger values of learning rate η and local epochs k result in faster convergence (less communication rounds) of `FedAvg` in the early communication rounds, but a worse quality of the final solution. It is noteworthy that similar observations were reported in [12, 34]. However, in the nonconvex setting, we observe a different behavior: in contrast to what observed in Fig. 2 for the previous convex settings, where fixing learning rate and reducing the number of local epochs improves the final solution quality at the cost of slower convergence, a similar trade-off does not exist in the

[3] We report the average result of 4 runs with different random seeds for all nonconvex experiments.

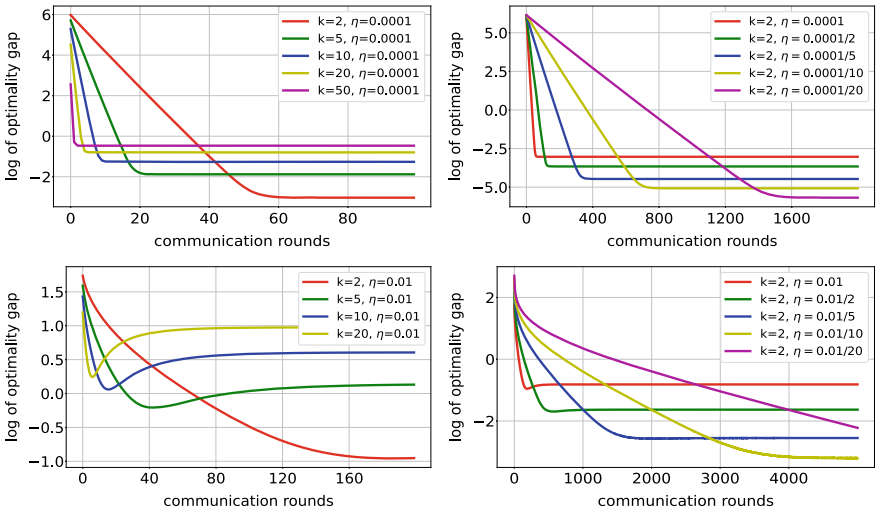

Fig. 2 `FedAvg` convergence for different values of η or k. Top: least squares; Bottom: logistic regression

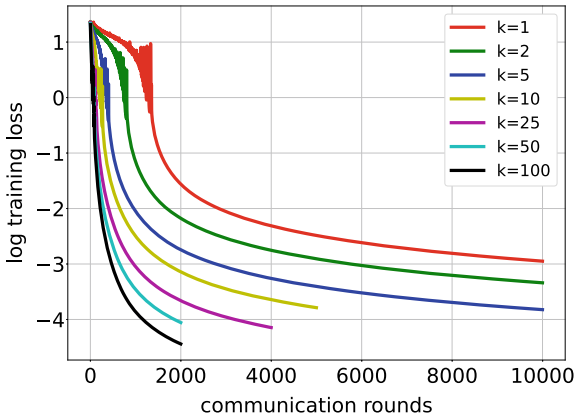

Fig. 3 `FedAvg` convergence on CIFAR-10 dataset for different values of k (local epochs). There are 10 users, and $\eta = 0.1$. The distribution of data among users is non-uniform

nonconvex setting (Fig. 3), (at least) given the computation we can afford. We guess that this is due to the combined complexity of the CIFAR-10 dataset and CNN model, which puts the solution we get still in an early stage. This is in comparison with the previous relatively simple convex problems, where the complexity of the models and the data were much less.

3.3 `FedProx` as Backward-Backward Splitting

Unlike `FedAvg`, which uses the gradient update $\mathsf{G}_{\mathsf{f},k}^{\eta_t}$, `FedProx` algorithm [2] uses proximal update $\mathsf{P}_{\mathsf{f}}^{\eta_t}$ on the users side:

$$\mathsf{w}_{t+1} = \mathsf{P}_H \mathsf{P}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t. \tag{23}$$

In order to approximate $\mathsf{P}_{\mathsf{f}}^{\eta}$, we may use a minibatch. Similarly, we may subsample a set of users to approximate P_H. From (23), we can understand that `FedProx` belongs to backward-backward splitting algorithms, which was proposed in [50, 51]. In fact, `FedProx` traces back to earlier works like the works of e.g. [52–54]. Also, in the ML community, `FedProx` was rediscovered by [48, 55] under a different motivation. As stated in [5], `FedProx` does not find the correct solution of the problem (1). While this statement seems correct, it does not consider some subtle points. In the following, we show that under appropriate conditions, `FedProx` could solve the original problem (1) exactly. First, we note that for a fixed η (step size), applying `FedProx` to the main problem (1) is equivalent as applying `FedAvg` to the following "regularized" problem [56]:

$$\min_{\mathsf{w}\in H} \tilde{\mathsf{f}}(\mathsf{w}), \quad \text{where} \quad \tilde{\mathsf{f}}(\mathsf{w}) := \left\langle \mathbf{1}, \mathsf{M}_{\mathbf{f}}^{\eta}(\mathsf{w}) \right\rangle. \tag{24}$$

In the above problem, $\mathsf{M}_{\mathbf{f}}^{\eta}(\mathsf{w}) = \left(\mathsf{M}_{f_1}^{\eta}(\mathbf{w}_1), \ldots, \mathsf{M}_{f_m}^{\eta}(\mathbf{w}_m)\right)$ is the vector of users Moreau envelopes. Note that $\mathsf{M}_f^{\eta} \to f$ (uniformly or pointwise for Lipschitz function f) as $\eta \to 0$, while as $\eta \to \infty, \mathsf{M}_f^{\eta} \to \min f$.[4] In other words, using a larger value for η in (24) "smoothens" data heterogeneity: $\mathsf{M}_{f_i}^{\eta}$ tend to have similar minimizers (note that their minimum values may be still different). This motivates us to understand the effect of small η (corresponding to (1)) and large η (corresponding to (24)) on the convergence behaviour of `FedProx`. This leads us to thinking about even adjusting η dynamically. In the following theorem, we show that this is indeed the key point in the convergence of `FedProx`: if η is adjusted properly and dynamically, `FedProx` finds the correct solution of the problem (1). We have assumed full gradient (i.e. large batch size) for simplicity, although we can extend the results to the case of stochastic gradient.

Theorem 3 *With assuming full participation of users in all rounds, and the step size* η_t *satisfying* $\sum_t \eta_t^2 < \infty$ *and* $\sum_t \eta_t = \infty$, *and convexity of* $\{f_i\}_{i=1}^m$, *the* `FedProx` averaged *iterates* $\bar{\mathsf{w}}_t := \frac{\sum_{s=1}^t \eta_s \mathsf{w}_s}{\sum_{s=1}^t \eta_s}$ *converge to the* exact *solution of problem* (1).

We use the following Lemma to prove Theorem 3.

Lemma 1 ([57], [58, Theorem 5.36]) *Assume* $\{\mathbf{z}_t\}$ *and* $\{\mathbf{w}_t\}$ *are two sequences with their elements in* $\mathbb{R}^d$, $W_k := \operatorname{cl}\operatorname{conv}(\mathbf{w}_t : t \geq k)$, *and* $\emptyset \neq \mathsf{F} \subseteq \mathbb{R}^d$. *Suppose that*

[4] These are some well-known and classic results, see e.g. [42].

1. $\|\mathbf{w}_t - \mathbf{w}\|_2^2 \to p(\mathbf{w}) < \infty$, *for all* $\mathbf{w} \in \mathsf{F}$;
2. $\mathsf{dist}(\mathbf{z}_t, W_k) \to 0$ *as* $t \to \infty$, *for all* k.

Then, one limit point of $\{\mathbf{z}_t\}$ *is in* F *(at most). Hence, if also*

3. *the sequence* $\{\mathbf{z}_t\}$ *has all its limit points in* F,

then $\{\mathbf{z}_t\}$ *(the whole sequence) converges to a point in* F.

There are some popular properties of Fejér monotone sequences and firm nonexpansions, which we use below. The excellent book [58] provides a useful background.

Proof The proof of this Lemma is mainly attributed to [50, 51]. We adapt the proof to our FL problem setting. Simply, we verify Lemma 1 in the following. Let $\mathsf{w} \in \mathrm{dom}\,\mathsf{f} \cap H$, $\mathbf{a}^* \in \partial \mathsf{f}(\mathsf{w})$ and $\mathbf{b}^* \in H^\perp$. Using $\mathsf{P}_\mathsf{f}^{\eta_t}$ being firmly nonexpansiveness, we have:

$$\begin{aligned}
\|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 &= \|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{P}_\mathsf{f}^{\eta_t}(\mathsf{w} + \eta_t \mathbf{a}^*)\|_2^2 \\
&\le \|\mathsf{w}_t - \mathsf{w} - \eta_t \mathbf{a}^*\|_2^2 - \|\mathsf{w}_t - \mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \eta_t \mathbf{a}^*\|_2^2 \\
&= \|\mathsf{w}_t - \mathsf{w}\|_2^2 + 2\eta_t \langle \mathsf{w} - \mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t; \mathbf{a}^* \rangle - \|\mathsf{w}_t - \mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t\|_2^2, \\
\|\mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 &\le \|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 + 2\eta_t \langle \mathsf{w} - \mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t; \mathbf{b}^* \rangle - \|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t\|_2^2.
\end{aligned}$$

Now, we sum the inequalities above and repeatedly apply the inequality $-\|\mathbf{x}\|_2^2 + 2\langle \mathbf{x}; \mathbf{y} \rangle \le \|\mathbf{y}\|_2^2$ to get:

$$\|\mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 \le \|\mathsf{w}_t - \mathsf{w}\|_2^2 + \eta_t^2[\|\mathbf{a}^* + \mathbf{b}^*\|_2^2 + \|\mathbf{a}^*\|_2^2] + 2\eta_t \langle \mathsf{w} - \mathsf{w}_t; \mathbf{a}^* + \mathbf{b}^* \rangle. \quad (25)$$

We now sum over t and then rearrange to obtain the following inequality for $\mathsf{w}^* = \mathbf{a}^* + \mathbf{b}^*$ and any $\mathsf{w} \in \mathrm{dom}\,\mathsf{f} \cap H$:

$$2\langle \mathsf{w} - \bar{\mathsf{w}}_t; \mathsf{w}^* \rangle + [\|\mathbf{a}^*\|_2^2 + \|\mathsf{w}^*\|_2^2] \sum_{k=0}^{t} \eta_k^2 / \Lambda_t \ge (\|\mathsf{w}_{t+1} - \mathsf{w}\|_2^2 - \|\mathsf{w}_1 - \mathsf{w}\|_2^2)/\Lambda_t,$$

where $\Lambda_t := \sum_{k=1}^{t} \eta_k$. Now, we use the assumptions on η_t and

$$\liminf_{t \to \infty} \ \langle \mathsf{w} - \bar{\mathsf{w}}_t; \mathsf{w}^* \rangle \ge 0.$$

Note that w is an arbitrary point with the subdifferential w^*. Therefore, any limit point of $\{\bar{\mathsf{w}}_t\}$ solves (1), i.e. condition 3 of Lemma 1 holds. Now, if we assume $\{\bar{\mathsf{w}}_t\}$ is bounded, then $\mathsf{F} \neq \emptyset$. Let $\mathsf{w} \in \mathsf{F}$ and set $\mathsf{w}^* = \mathbf{0}$ we know from (25) that $\{\mathsf{w}_t\}$ is quasi-Fejér monotone w.r.t. F (i.e. condition 1 in Lemma 1 holds). As the last step, let $\bar{\eta}_{t,k} := \eta_k / \Lambda_t$ and we verify the condition (II) mentioned in Lemma 1:

$$\begin{aligned}\mathsf{dist}(\bar{\mathsf{w}}_t, W_k) &\le \left\| \bar{\mathsf{w}}_t - \sum_{s=k}^{t} \bar{\eta}_{t,s} \mathsf{w}_s / \sum_{\kappa=k}^{t} \bar{\eta}_{t,\kappa} \right\|_2 \\ &\le \sum_{\kappa=0}^{k-1} \bar{\eta}_{t,\kappa} \left[\|\mathsf{w}_\kappa\|_2 + \left\| \sum_{s=k}^{t} \bar{\eta}_{t,s} \mathsf{w}_s \right\|_2 / \sum_{\kappa=k}^{t} \bar{\eta}_{t,\kappa} \right] \\ &\xrightarrow{t\to\infty} 0,\end{aligned}$$

since for any k, $\bar{\eta}_{t,k} \to 0$ as $t \to \infty$ and also w_t is bounded. Therefore, we have successfully verified all the conditions of Lemma 1. □

The following simple example shows that for `FedProx` to converge correctly, it is in general necessary to have $\sum_t \eta_t = \infty$ and $\lim_t \eta_t = 0$.

Necessary conditions on the step size of `FedProx`

Let $f_+(w) = \frac{1}{2}(w+1)^2$ and $f_-(w) = \frac{1}{2}(w-1)^2$. By simple calculations, we can verify that

$$\mathsf{P}^{\eta}_{f_\pm}(w) = \tfrac{w \mp \eta}{1+\eta}.$$

We can find the `FedProx` iterates for these two functions as follows:

$$w_{t+1} = \tfrac{w_t}{1+\eta_t} = \prod_{\tau=0}^{t} \tfrac{1}{1+\eta_\tau} w_0.$$

It is clear that for any w_0, the iterates w_t converges the true minimizer $w_\star = 0$ iff

$$\prod_{\tau=0}^{t} \tfrac{1}{1+\eta_\tau} \to 0 \iff \sum_t \eta_t \to \infty.$$

Now, if we instead consider the two functions f_+ and $2f_-$, then

$$2w_{t+1} = \tfrac{w_t - \eta_t}{1+\eta_t} + \tfrac{w_t + 2\eta_t}{1+2\eta_t}.$$

Passing to a subsequence if necessary, suppose $w_t \to w_\star = \frac{1}{3}$ and let $\eta_t \to \eta \neq 0$, then passing to the limit we get

$$\begin{aligned} 2 = \tfrac{1-3\eta}{1+\eta} + \tfrac{1+6\eta}{1+2\eta} &\iff 2(1+\eta)(1+2\eta) = (1-3\eta)(1+2\eta) + (1+6\eta)(1+\eta) \\ &\iff \eta = 0, \end{aligned}$$

which is a contradiction. Hence, the condition $\eta_t \to 0$ is necessary for `FedProx` to converge to $w_\star = \frac{1}{3}$.

The above example shows that our step size condition in Theorem 3 is sufficient and close to necessary. The subtle point is that η_t *must approach to 0 at a reasonably slow rate*. This point was not considered in [5]: it was assumed that the step size η_t is always fixed. This subtle point and Theorem 3 also make sense: we track the solution of the problem (24) as $\eta_t \to 0$ slowly. This regularized problem itself tends to the original problem (1), as $\eta_t \to 0$ slowly: we know that $\mathsf{M}_f^{\eta} \to f$ as $\eta \to 0$. Interestingly, what [20] proposed for the purpose of personalization is exactly the `FedProx` algorithm. The personalization is automatically achieved by applying `FedProx` to the problem (1).

We now note that, in some cases, which are explained in the following theorem, even the averaging step in Theorem 3 is not necessary.

Theorem 4 ([51]) *With the same assumptions of Theorem 3, if* (1) *has a solution set with nonempty interior or* $\mathbf{f}$ *is strongly convex, then the* `FedProx` *vanilla iterates* w_t *also converge.*

When there is a nonempty interior for the solution set of (1), the convergence is linear. Nevertheless, it is usually not easy to verify or satisfy the two conditions above in most applications. Hence, using the ergodic averaging in Theorem 3 in practice is recommended. It can be shown that, even in the absence of convexity, the regularized problem (24) and the original problem (1) are quantitatively related.

Theorem 5 ([56]) *Suppose each* f_i *is* M_i*-Lipschitz continuous (and possibly non-convex), then*

$$\forall \mathsf{w}, \quad f(\mathsf{w}) - \tilde{f}(\mathsf{w}) \leq \textstyle\sum_i \lambda_i \frac{\eta M_i^2}{2}. \tag{26}$$

Therefore, we conclude that, as long as the step size η is small, `FedAvg` that aims at minimizing f and `FedProx`, which aims at minimizing the regularized function $\tilde{\mathsf{f}}$, are quantitatively close. This point again shows the vital role of step size η. We remark that, if the functions f_i are "definable", we may remove the need of ergodic averaging for `FedProx` (following the ideas in [56]). As the potential gain is not significant, we omit the technical details.

In Fig. 4, the effect of η on `FedProx` convergence is shown. Also, the results of `FedAvg` are compared with them. We have tried both fixed and diminishing step sizes. When η_t diminishes by time, we set its initial value (i.e. η_0) to larger values to make sure that η does not diminish too fast after the few initial communication rounds. It can be observed from the convex experiments (Fig. 4, top left and top right) that when η is fixed, `FedProx` converges fast but to a suboptimal solution. When η diminishes by time, the convergence of `FedProx` is slower, but the quality of its final solution is better.

Interestingly, when both of the conditions of Theorem 3 are satisfied, the fixed point of `FedProx` (with ergodic averaging) is a correct solution of (1). This shows that the step size η should diminish by time but neither too fast nor too slow. This can be observed in the results obtained for $\eta_t \propto 1/t$. In this case, η_t meets both of the conditions mentioned in Theorem 3. As can be observed in this case, the

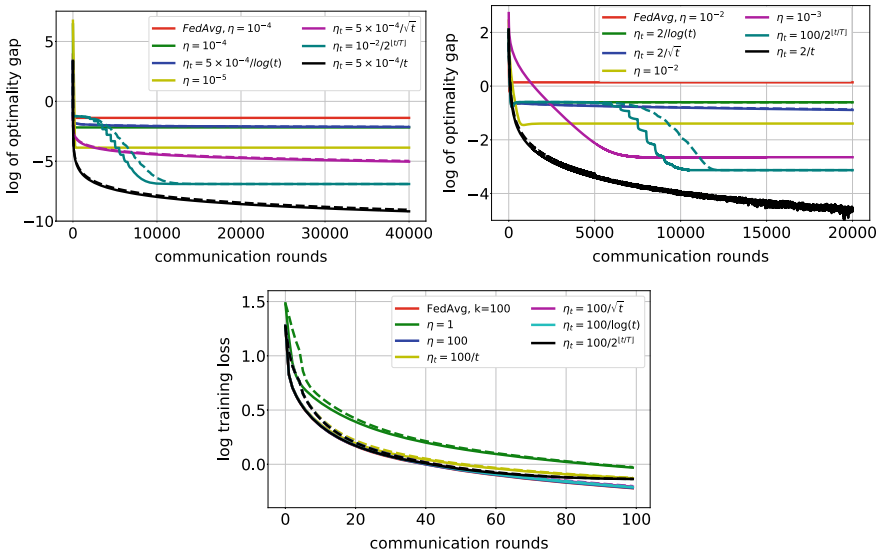

Fig. 4 Convergence of `FedProx`. Top left: least squares; Top right: logistic regression; Bottom: CNN on MNIST. The results without and with the averaging step mentioned in Theorem 3 are shown with the solid and dashed lines with the same color, respectively. For decaying η_t exponentially, the period T is equal to 500 for both of the convex experiments, and is equal to 10 for CNN experiment

`FedProx` iterates converge to a solution of (1) correctly. Another important and surprising observation about the nonconvex problem of the CNN model in Fig. 4 is that larger learning rates result in not only faster convergence of `FedProx` but also smaller training loss values. It is noteworthy that in nonconvex problems, a similar observation was mentioned in [1, Figs. 5 and 6] for `FedAvg`. Furthermore, it can be seen in Fig. 4 that the convergence rate of `FedProx` is not affected by the ergodic averaging noticeably.

An interpretation of `FedProx` in terms of personalization

When the distribution of data among users is non-IID, “multi-model" techniques become important. These techniques learn different models for different users: each user has its own “personalized" model at inference time. In non-IID settings, these local personalized models may perform better than the global model which is shared among all users. Various approaches for personalization has been proposed in `FL`. We briefly discuss some of these techniques in the following.

Multi-task learning [19, 59, 60] is one of the personalization approaches, in which each user problem is treated as a different task, and one model is learned for each task/user. Local fine-tuning is another family of techniques for personalization: given a global trained model, each user personalizes it based on its local data before using it at inference time. The precise technique used for this type of personalization is a key factor: fine-tuning, transfer learning, domain adaptation are among the main

approaches used for transferring knowledge between different but related learning tasks in non-Federated learning settings. However, these approaches often consider a pair of source and target domains. Hence, their application to FL setting is limited. Liu et al. [61] applied transfer learning techniques to FL to build a secure federated transfer-learning technique to transfer the learned knowledge from one user's learning task to another user. Application of [61] in federated settings is limited to FL structures with only two clients, as mentioned earlier. *model agnostic meta learning*, which is another class of algorithms, have been used for meta-learning a global model, which can be personalized by users with further training it locally based on their local data [62–64].

A personalization approach which is close to our analysis of FedProx is proposed in [20]. Interestingly, exactly (24) was proposed for the purpose of personalized FL. Now, we know that (24) is automatically achieved by applying FedProx to (1). More specifically, $\nabla \mathsf{M}_{\mathbf{f}}^{\eta}(\mathsf{w}) = [\mathsf{w} - \mathsf{P}_{\mathbf{f}}^{\eta}(\mathsf{w})]/\eta$. Therefore, $\mathsf{G}_{\tilde{f}}^{\eta}(\mathsf{w}) = \mathsf{w} - \eta \nabla \tilde{\mathsf{f}}(\mathsf{w}) = \mathsf{P}_{\mathbf{f}}^{\eta}(\mathsf{w}) = \left(\mathsf{P}_{f_1}^{\eta}(\mathbf{w}_1), \ldots, \mathsf{P}_{f_m}^{\eta}(\mathbf{w}_m)\right)$. While [20] proved that their proposed algorithm converges to a solution when the value of η is fixed, they did not investigate the effect of varying η on the quality of the solution w.r.t to minimizing the original problem (1). Reference [20] argued that smaller fixed value of η is equivalent to more personalization by users and less contribution by them in data aggregation. Therefore, an interpretation for Theorem 3 is that reasonably slow reduction of users personalization by time in favor of data aggregation leads to gradual improvement of the global model quality. Further, a surprising result that they have reported is that, smaller values of η results in faster convergence of their algorithm to its fixed point solution; however, as we have shown smaller values of η lead to slower convergence of FedProx to a better suboptimal solution of (1).

3.4 FedSplit as Peaceman-Rachford Splitting

The recent work of [5] introduced the FedSplit algorithm:

$$\mathsf{w}_{t+1} = \mathsf{R}_H \mathsf{R}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t. \tag{27}$$

This algorithm is based on the Peaceman-Rachford splitting algorithm [35, 36]. It was shown by [36] that if $\mathbf{f}$ is strictly convex, it converges to the exact solution of (1). Also, they showed that when $\mathbf{f}$ is smooth and strongly convex, the convergence rate of FedSplit is linear. When the $\mathsf{R}_{\mathsf{f}}^{\eta}$ is not computed accurately, the convergence of FedSplit was studied in [5]. However, its convergence for nonconvex problems is not studied widely. We have the following surprising theorem about FedSplit.

Theorem 6 *In order for R_f^{η} to be a (strict) contraction, the function f needs to be strongly convex.*

Proof Note that f is strongly convex iff ηf is strongly convex. Also, $\mathsf{R}_f^{\eta} = \mathsf{R}_{\eta f}$. Hence, w.l.o.g. we can take $\eta = 1$. Assume, for some $\gamma \in (0, 1)$, R_f is γ-contractive, i.e. for all $\mathbf{z}$ and $\mathbf{w}$:

$$\|\mathsf{R}_f \mathbf{z} - \mathsf{R}_f \mathbf{w}\|_2 \leq \gamma \cdot \|\mathbf{z} - \mathbf{w}\|_2. \tag{28}$$

Then, we can conclude that $\mathsf{P}_f = \frac{\mathrm{id}+\mathsf{R}_f}{2}$ is $\frac{1+\gamma}{2}$-contractive. Moreover, $\frac{2}{1+\gamma}\mathsf{P}_f$, being nonexpansive, is the gradient of the convex function $\frac{2}{1+\gamma}\mathsf{M}_{f^*}$. Thus, we can conclude firm nonexpansiveness of $\frac{2}{1+\gamma}\mathsf{P}_f$ [58, Corollary 18.17]. But,

$$\tfrac{2}{1+\gamma}\mathsf{P}_f = \tfrac{2}{1+\gamma}(\mathrm{id} + \partial f)^{-1} = [\mathrm{id} + (\partial f - \tfrac{1-\gamma}{1+\gamma}\mathrm{id}) \circ \tfrac{1+\gamma}{2}\mathrm{id}]^{-1}, \tag{29}$$

and hence $(\partial f - \frac{1-\gamma}{1+\gamma}\mathrm{id}) \circ \frac{1+\gamma}{2}\mathrm{id}$ is maximal monotone [58, Proposition 23.8], i.e. f is $\frac{1-\gamma}{1+\gamma}$-strongly convex. □

If η is small and f is also smooth, the converse is also true [36, 65]. Hence, we conclude that for nonconvex and also non-strongly convex problems, linear convergence for `FedSplit` cannot be expected (if `FedSplit` converges at all for these problems).

3.5 *`FedPi` as Douglas-Rachford Splitting*

Douglas-Rachford splitting [36, 37] is a popular splitting algorithm that can be used instead of the Peaceman-Rachford splitting. This splitting algorithm has not been used in `FL`. We can write the update from Douglas-Rachford splitting as:

$$\mathsf{w}_{t+1} = \frac{\mathsf{w}_t + \mathsf{R}_H \mathsf{R}_{\mathsf{f}}^{\eta_i} \mathsf{w}_t}{2}, \tag{30}$$

i.e. at each round the iterate of `FedSplit` and the current iterate are averaged evenly. This averaging step makes `FedPi` more stable compared to `FedSplit`. The above algorithm is based on partial inverse method, which was rediscovered by [66]. Douglas-Rachford splitting is more general and partial inverse is a special case of it. We have the following theorem about `FedPi`.

Theorem 7 ([36, 66]) *With assuming full participation of users in all communication rounds, a fixed step size η_t, and convex functions $\{f_i\}$, `FedPi` iterates w_t converge to the exact solution of* (1).

In comparison with `FedSplit`, `FedPi` is more stable, and imposes less conditions on f_i. As already noted by [36], when f_i is indeed smooth and strongly convex and the fixed η is set appropriately, the convergence speed of `FedPi` will be less. Also,

as it was recently shown by [67], analyzing FedPi on nonconvex functions may be easier, compared to FedSplit.

Interestingly, FedPi also has close ties to FedProx. Indeed, this is best seen by expanding the concise formula in (30) and introducing a "dual variable" u on the server side[5]:

$$\mathsf{z}_{t+1} \leftarrow \mathsf{P}_{\mathsf{f}}^{\eta}(\mathsf{w}_t + \mathsf{u}_t) \tag{31}$$

$$\mathsf{w}_{t+1} \leftarrow \mathsf{P}_H(\mathsf{z}_{t+1} - \mathsf{u}_t) \tag{32}$$

$$\mathsf{u}_{t+1} \leftarrow \mathsf{u}_t + \mathsf{w}_{t+1} - \mathsf{z}_{t+1}. \tag{33}$$

From the last two updates (32) and (33) it is clear that u_{t+1} is always in $H^{\perp}$. Thus, after performing a change of variable $\mathsf{v}_t := \mathsf{w}_t + \mathsf{u}_t$ and exploiting the linearity of P_H, we obtain exactly FedPi:

$$\begin{aligned} \mathsf{v}_{t+1} = \mathsf{u}_t + 2\mathsf{w}_{t+1} - \mathsf{z}_{t+1} &= \mathsf{v}_t - \mathsf{P}_H \mathsf{v}_t + 2\mathsf{P}_H \mathsf{P}_{\mathsf{f}}^{\eta} \mathsf{v}_t - \mathsf{P}_{\mathsf{f}}^{\eta} \mathsf{v}_t \\ &= \tfrac{\mathsf{v}_t + \mathsf{R}_H \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{v}_t}{2}. \end{aligned} \tag{34}$$

Comparing (23) and (31)–(32) it is clear that FedProx corresponds to fixing the dual variable u to the constant $\mathbf{0}$ in FedPi. We remark that step (31) is done at the users' side while steps (32) and (33) are implemented at the server side. There is no communication overhead either, as the server need only communicate the sum $\mathsf{w}_t + \mathsf{u}_t$ to the respective users while the users need only communicate their $\mathbf{z}_{t,i}$ to the server. The dual variable u is kept entirely at the server's expense.

Let us point out a subtle difference that may prove useful in FL: FedAvg and FedProx are inherently "synchronous" algorithms, in the sense that all participating users start from a common, averaged model at the beginning of each communication round. In contrast, the local models z_t in FedPi may be different from each other, where we "correct" the common, average model w_t with user-specific dual variables u_t. This opens the possibility to personalization by designating the dual variable u in user-specific ways.

Lastly, we remark that FedProx needs a step size η_t which diminishes reasonably slowly to converge to a correct minimizer in (1) whereas FedPi can achieve it with a fixed step size, and only doubles the memory needed at the server.

3.6 *FedRP as Reflection-Projection Splitting*

Examining the updates in (13), (23) and (27), we are naturally led to the following further variants (that have not been tried in FL to the best of our knowledge):

[5] The acute readers may have recognized here the alternating direction method of multipliers (ADMM). Indeed, the equivalence of ADMM, Douglas-Rachford and partial inverse (under our FL setting) has long been known [e.g. 65, 68].

$$\mathsf{w}_{t+1} = \mathsf{R}_H \mathsf{G}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t \tag{35}$$

$$\mathsf{w}_{t+1} = \mathsf{R}_H \mathsf{P}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t \tag{36}$$

$$\texttt{FedRP}: \quad \mathsf{w}_{t+1} = \mathsf{P}_H \mathsf{R}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t. \tag{37}$$

Interestingly, the last variant in (37), which we call `FedRP`, has been studied by [39] under the assumption that $\mathsf{f} = \iota_{\mathsf{K}}$ is an indicator function of an *obtuse*[6] convex cone K. We propose `FedRP` as a new `FL` algorithm and we prove the following result about it.

Theorem 8 *Let each user participate in every round and the functions* $\{f_i\}$ *be convex. If the step size* $\eta_t \equiv \eta$ *is constant, any fixed point of* `FedRP` *solves the* regularized *problem* (24)*. If the reflector* $\mathsf{R}_{\mathsf{f}}^{\eta}$ *is also idempotent* ($\mathsf{R}_{\mathsf{f}}^{\eta}\mathsf{R}_{\mathsf{f}}^{\eta} = \mathsf{R}_{\mathsf{f}}^{\eta}$)*, then the* `FedRP` *vanilla iterates* w_t *converge.*

Proof To see the first claim, let w be a fixed point of `FedRP`, i.e.

$$\mathsf{w} = \mathsf{P}_H \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w} = \mathsf{P}_H (2\mathsf{P}_{\mathsf{f}}^{\eta} \mathsf{w} - \mathsf{w}) = 2\mathsf{P}_H \mathsf{P}_{\mathsf{f}}^{\eta} \mathsf{w} - \mathsf{P}_H \mathsf{w} = 2\mathsf{P}_H \mathsf{P}_{\mathsf{f}}^{\eta} \mathsf{w} - \mathsf{w}, \tag{38}$$

since P_H is linear and $\mathsf{w} \in H$ due to the projection. Thus, $\mathsf{w} = \mathsf{P}_H \mathsf{P}_{\mathsf{f}}^{\eta} \mathsf{w}$. In other word, the fixed points of `FedRP` are exactly those of `FedProx`, and the first claim then follows from [56], see the discussions in Sect. 3.3 and also [39, Lemma 7.1].

To prove the second claim, we first observe that $\{\mathsf{w}_t\}$ is Fejér monotone w.r.t. F, the solution set of the regularized problem (24). Indeed, for any $\mathsf{w} \in \mathsf{F}$, using the firm nonexpansiveness of P_H we have

$$\|\mathsf{w}_{t+1} - \mathsf{w}\|_2^2 + \|\mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}_t - \mathsf{w}_{t+1} - \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w} + \mathsf{w}\|_2^2 \leq \|\mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}_t - \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}\|_2^2 \leq \|\mathsf{w}_t - \mathsf{w}\|_2^2. \tag{39}$$

Summing and telescoping the above we know

$$\mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}_t - \mathsf{w}_{t+1} \to \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w} - \mathsf{w}. \tag{40}$$

Let w_∞ be a limit point of $\{\mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}_t\}$ (which exists since w_t is bounded). Since $\mathsf{R}_{\mathsf{f}}^{\eta}$ is idempotent, we have the range $\mathrm{rge}\, \mathsf{R}_{\mathsf{f}}^{\eta}$ equal to its fixed point $\mathrm{Fix}\mathsf{R}_{\mathsf{f}}^{\eta}$. Therefore, $\mathsf{w}_\infty \in \mathrm{rge}\, \mathsf{R}_{\mathsf{f}} = \mathrm{Fix}\mathsf{R}_{\mathsf{f}}$, When the users are homogeneous, $\mathsf{w} = \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}$ and hence $\mathsf{w}_\infty = \lim_{k\to\infty} \mathsf{R}_{\mathsf{f}}^{\eta} \mathsf{w}_{t_k} = \lim_{k\to\infty} \mathsf{w}_{t_k+1} \in H$. Therefore, $\mathbf{w}_\infty \in \mathsf{F}$ and applying Lemma 1 we conclude that the entire Fejér sequence $\{\mathbf{w}_t\}$ converges to $\mathbf{w}_\infty \in \mathsf{F}$. □

As shown in [39], a (closed) convex cone is obtuse iff its reflector is idempotent. To give another example: the univariate, idempotent reflector $\mathsf{R}_f(w) = (w)_+ := \max\{w, 0\}$ leads to

$$\mathsf{P}_f(w) = \tfrac{w + \mathsf{R}_f w}{2} = \tfrac{w + (w)_+}{2} \implies f(w) = \tfrac{1}{2}(-w)_+^2. \tag{41}$$

[6] Recall that a convex cone K is obtuse iff its dual cone $\mathsf{K}^* := \{\mathbf{w}^* : \langle \mathbf{w}, \mathbf{w}^* \rangle \geq 0\}$ is contained in itself.

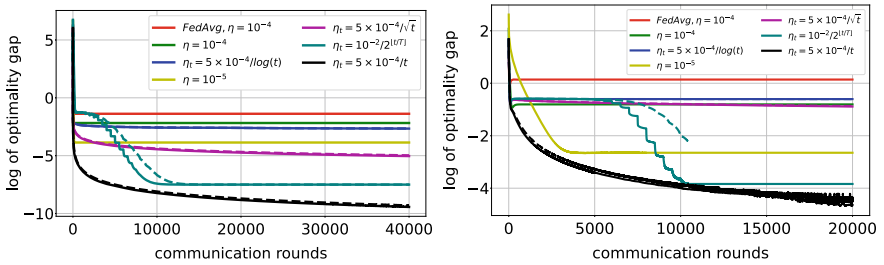

Fig. 5 Convergence of `FedRP`. Left: least squares problem; Right: logistic regression problem. The obtained results without and with the ergodic averaging step are shown with the solid and dashed lines with the same color, respectively. For decaying η_t exponentially, the period T is equal to 500 for both of the experiments

Interestingly, the epigraph of the above f is a convex set but not a cone and yet its reflector is idempotent. Therefore, the idempotent condition introduced here is a slight generalization of [39].

We remark that Theorem 8 does not apply to the variants (35) and (36) since the reflector R_H is not idempotent (recall H is defined in (4)). Of course, we can prove (linear) convergence of both variants (35) and (36) if f is strongly convex. We omit the formal statement and proof since strong convexity does not appear to be easily satisfiable in practice.

Figure 5 shows the effect of diminishing η on `FedRP` convergence. We have considered the least squares and logistic regression problems. These results supplement the results in Fig. 4. We use both fixed and diminishing step sizes to run `FedRP`. When using diminishing step size η_t, we initially set η to larger values in the first rounds to make sure that η does not diminish too fast after the few initial communication rounds. It can be observed from the figure that `FedRP` with a fixed learning rate converges faster but to a sub-optimal solution. On the other hand, with diminishing η, `FedRP` converges slower but the quality of its final solution is better. Interestingly, the convergence behaviour of `FedRP` is similar to that of `FedProx`. More specifically, when all the conditions of Theorem 3 are satisfied, the iterates of `FedRP` (with ergodic averaging) converge to a correct solution of (1). This can be observed in the results for $\eta_t \propto 1/t$. Moreover, similar to `FedProx`, it can be observed that the ergodic averaging hardly affects the convergence speed of `FedRP`.

3.7 Comparison

We compare the effect of data heterogeneity on different splitting algorithms in Fig. 6. We use least squares as a convex problem. We also use training a CNN model on MNIST as a nonconvex problem (see Sect. 3.1). As observed, the lowest optimality gaps for the least squares convex problem are achieved with `FedAvg`

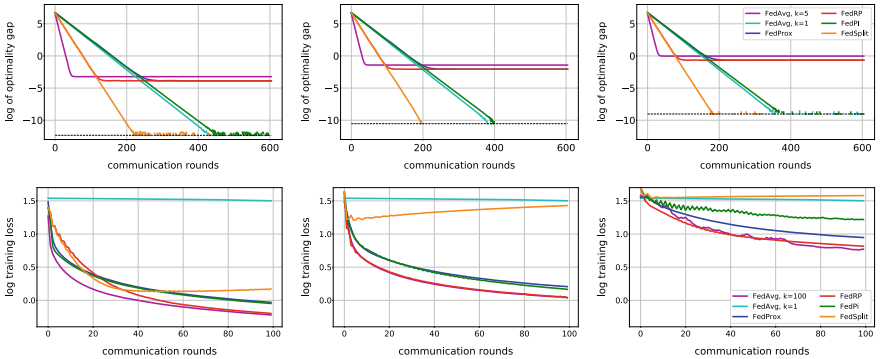

Fig. 6 The variation in the performance of different splitting algorithms with the change in users' data heterogeneity. Top: convex least squares problem. Heterogeneity increases from left to right. Left: $H \approx 118 \times 10^3$; Middle: $H \approx 7.62 \times 10^6$; Right:$H \approx 190.31 \times 10^6$. Bottom: nonconvex CNN model problem. Left: IID. distribution; Middle: non-IID distribution, each user has maximum 6 classes; Right: non-IID distribution, each user has maximum 2 classes

($k = 1$), `FedSplit` and `FedPi`. Also, as data heterogeneity increases, the quality of the final solutions found by the algorithms deteriorate. On the other hand, the best results for the nonconvex setting are obtained by `FedAvg` with large k and `FedRP`. Also, as can be observed, `FedAvg` with with $k = 100$ is considerably better than `FedAvg` with $k = 1$ in the nonconvex setting. Again, this shows the differences between convex and nonconvex settings.

4 Unification and Acceleration

The drawn connection between operator splitting theory and `FL` also lets us to propose a framework for unification and acceleration of many existing `FL` algorithms. In this section, we unify all the aforementioned `FL` algorithms into a grand scheme, and also explain how to practically adapt Anderson acceleration [40] to the unifying framework to accelerate `FL` algorithms.

4.1 Unification

Consider the following grand scheme:

$$\mathsf{z}_{t+1} = (1-\alpha_t)\mathsf{u}_t + \alpha_t \mathsf{P}_{\mathsf{f}}^{\eta_t}(\mathsf{u}_t) \tag{42}$$

$$\mathsf{w}_{t+1} = (1-\beta_t)\mathsf{z}_{t+1} + \beta_t \mathsf{P}_H(\mathsf{z}_{t+1}) \tag{43}$$

$$\mathsf{u}_{t+1} = (1-\gamma_t)\mathsf{u}_t + \gamma_t \mathsf{w}_{t+1}. \tag{44}$$

Table 3 Unification of FL algorithms. (a) FedAvg uses gradient update $\mathsf{G}_{\mathsf{f},k}^{\eta}$ instead of the proximal update $\mathsf{P}_{\mathsf{f}}^{\eta}$; (b) ? shows the properties that are not studied; (c) "stochastic" means updating models with stochastic gradient; (d) "sampling" means selecting a set of users at each round

Algorithm	α	β	γ	$\eta_t \equiv \eta$	$\sum_t \eta_t^2 < \infty$, $\sum_t \eta_t = \infty$	Nonconvex	Stochastic	Sampling
FedAvg	1	1	1	Eq. (1)	Eq. (1)	✓	✓	✓
FedProx	1	1	1	Eq. (24)	Eq. (1)	✓	✓	✓
FedSplit	2	2	1	Eq. (1)	–	?	?	?
FedPi	2	2	$\frac{1}{2}$	Eq. (1)	–	✓	?	?
FedRP	2	1	1	Eq. (24)	Eq. (1)	?	?	?

From Table 3, we can clearly observe that the aforementioned FL algorithms are special cases of this unifying grand scheme. This unification reveals the differences and similarities between different algorithms. Therefore, it shows the possibility to transfer the progress from one algorithm to other algorithms and vice versa. This unifying scheme also enables us to provide new algorithmic variants with different configurations of the grand scheme parameters. For example, the proposed FedPi and FedRP algorithms have a different parameter configuration than the previously existing algorithms, as observed in Table 3.

4.2 Acceleration

The proposed unification also enables us to accelerate different algorithms. Let us abstract the grand scheme (42)–(44) as the map $\mathsf{u}_{t+1} = \mathsf{T}\mathsf{u}_t$, where T is nonexpansive if f is convex and $\alpha_t, \beta_t \in [0, 2]$, $\gamma_t \in [0, 1]$. Following [41], we may then apply the Anderson type-II acceleration to further improve convergence. Let $U = [\mathsf{u}_{t-\tau}, \ldots, \mathsf{u}_t]$ be given along with $T = [\mathsf{T}\mathsf{u}_{t-\tau}, \ldots, \mathsf{T}\mathsf{u}_t]$. We solve the following simple least-squares problem:

$$\boldsymbol{\pi}^* = \underset{\boldsymbol{\pi}^\top \mathbf{1}=1}{\operatorname{argmin}} \{\|(U - T)\boldsymbol{\pi}\|_2^2\} = \frac{G^\dagger \mathbf{1}}{\mathbf{1}^\top G^\dagger \mathbf{1}}, \tag{45}$$

where $G = (U - T)^\top (U - T)$ and note that we do not require $\boldsymbol{\pi}$ to be nonnegative. Then, we update $\mathsf{u}_{t+1} = T\boldsymbol{\pi}^*$. Clearly, when $\tau = 0$, $\pi^* = 1$ and we reduce to $\mathsf{u}_{t+1} = \mathsf{T}\mathsf{u}_t$. With a larger memory size τ, we may significantly improve convergence. Importantly, all heavy lifting (computation and storage) is done at the server side and we do not increase communication at all. We note that the same acceleration can also be applied on each user for computing $\mathsf{P}_{\mathsf{f}}^{\eta_t}$, if the users can afford the memory cost.

We have already seen how different variants (FedAvg, FedProx, FedSplit, FedPi, FedRP) compare to each other in Sect. 3. We performed further experi-

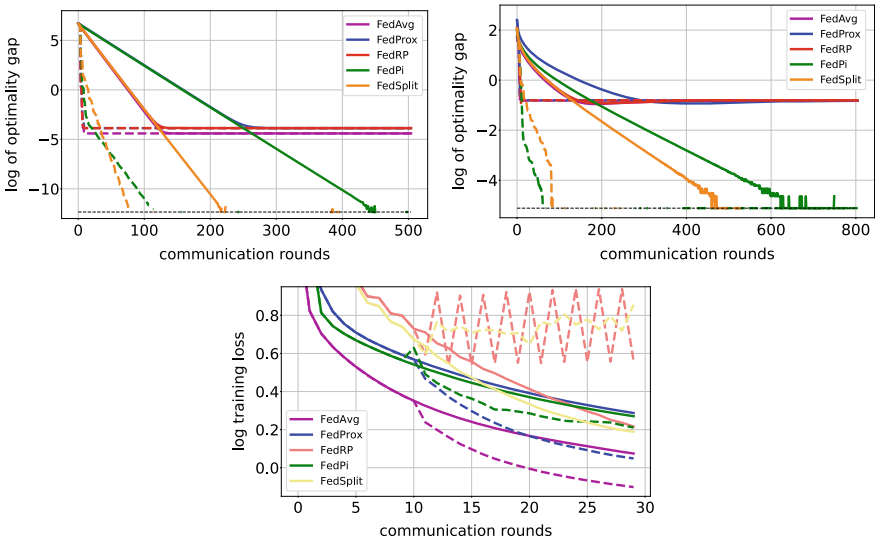

Fig. 7 Effect of Anderson acceleration. Top left: least squares with $\tau = 2$; Top right: logistic regression $\tau = 2$; Bottom: nonconvex CNN with $\tau = 10$. Dashed lines are the accelerated results

ments to illustrate their behaviour under Anderson acceleration. As can be observed in Fig. 7, Anderson acceleration helps `FL` algorithms converge considerably faster, *all without incurring any overhead*. For the convex models (least squares and logistic regression), our implementation of Anderson-acceleration speeds up all of the algorithms, especially `FedAvg`, `FedProx` and `FedRP`. However, for the nonconvex CNN model, it is beneficial only for `FedAvg`, `FedProx` and `FedPi`, while applying it to `FedRP` and `FedSplit` makes them unstable. It is noteworthy that we already know `FedProx` and `FedPi` are more stable than `FedRP` and `FedSplit`, respectively, and hence it makes intuitive sense that acceleration improves the two more stable algorithms. Another important point is that our acceleration method does not affect the quality of the algorithms' final solutions, but rather it just accelerates their convergence.

5 Conclusions

By drawing a connection between `FL` and the operator splitting theory in optimization, we obtained new understandings about existing `FL` algorithms. We showed the important role of step size in different `FL` algorithms and observed some differences between convex and nonconvex settings. We also refined the convergence analysis of the existing algorithms and proposed new `FL` algorithms. We showed that all the algorithms can be unified into a grand scheme. With this unified framework, it is easier to understand and implement different `FL` algorithms. Furthermore, it allows

us to accelerate different FL algorithms in a standardized and efficient way without incurring any additional communication overhead. Studying the effect of stochasticity on FL algorithms and extending our convergence analysis to nonconvex settings are two important directions, which we plan to study in future. While our initial experiments confirmed the potential gain of Anderson-acceleration for FL, further work is required to formalize its effect in theory and to understand its relation with other momentum methods in FL.

References

1. McMahan B, Moore E, Ramage D, Hampson S, Agüera y Arcas B (2017) Communication-efficient learning of deep networks from decentralized data. AISTATS 54:1273–1282
2. Li T, Kumar Sahu A, Zaheer M, Sanjabi M, Talwalkar A, Smith V (2020) Federated optimization in heterogeneous networks. Proc Mach Learn Syst 2:29–450
3. Yurochkin M, Agarwal M, Ghosh S, Greenewald K, Hoang N, Khazaeni Y (2019) Bayesian nonparametric federated learning of neural networks. ICML 97:7252–7261
4. Reddi S, Charles Z, Zaheer M, Garrett Z, Rush K, Konečny J, Kumar S, Brendan McMahan H (2020) Adaptive federated optimization. arXiv:2003.00295
5. Pathak R, Wainwright MJ (2020) FedSplit: an algorithmic framework for fast federated optimization. In: NeurIPS
6. Huo Z, Yang Q, Gu B, Carin L, Huang H (2020) Faster on-device training using new federated momentum algorithm. arXiv:2002.02090
7. Wang H, Yurochkin M, Sun Y, Papailiopoulos D, Khazaeni Y (2020) Federated learning with matched averaging. In: ICLR
8. Li Z, Kovalev D, Qian X, Richtárik P (2020) Acceleration for compressed gradient descent in distributed and federated optimization. In: ICML, pp 5895–5904
9. Khaled A, Mishchenko K, Richtárik P (2020) First analysis of local GD on heterogeneous data. arXiv:1909.04715
10. Li X, Huang K, Yang W, Wang S, Zhang Z (2020) On the convergence of FedAvg on non-IID data. In: ICLR
11. Khaled A, Mishchenko K, Richtarik P (2020) Tighter theory for local SGD on identical and heterogeneous data. In: AISTATS. Proceedings of machine learning research, vol 108, pp 4519–4529
12. Malinovskiy G, Kovalev D, Gasanov E, Condat L, Richtarik P (2020) From local SGD to local fixed-point methods for federated learning. ICML 119:6692–6701
13. Gorbunov E, Hanzely F, Richtárik P (2021) Local SGD: unified theory and new efficient methods. AISTATS 130:3556–3564
14. Mohri M, Sivek G, Theertha Suresh A (2019) Agnostic federated learning. ICML 97:4615–4625
15. Li T, Sanjabi M, Beirami A, Smith V (2020) Fair resource allocation in federated learning. In: ICLR
16. Hu Z, Shaloudegi K, Zhang G, Yu Y (2020) FedMGDA+: federated learning meets multi-objective optimization
17. Nasr M, Shokri R, Houmansadr A (2019) Comprehensive privacy analysis of deep learning: passive and active white-box inference attacks against centralized and federated learning. In: IEEE symposium on security and privacy (SP), pp 739–753
18. Augenstein S, Brendan McMahan H, Ramage D, Ramaswamy S, Kairouz P, Chen M, Mathews R, Aguera y Arcas B (2020) Generative models for effective ML on private, decentralized datasets. In: ICLR

19. Mansour Y, Mohri M, Ro J, Theertha Suresh A (2020) Three approaches for personalization with applications to federated learning. arXiv:2002.10619
20. Dinh CT, Tran N, Nguyen J (2020) Personalized federated learning with moreau envelopes. In: NeurIPS, pp 21394–21405
21. Diao E, Ding J, Tarokh V (2021) HeteroFL: computation and communication efficient federated learning for heterogeneous clients. In: ICLR
22. Zhang M, Sapra K, Fidler S, Yeung S, Alvarez JM (2021) Personalized federated learning with first order model optimization. In: ICLR
23. Deng Y, Mahdi Kamani M, Mahdavi M (2020) Adaptive personalized federated learning. arXiv:2003.13461
24. Nitin Bhagoji A, Chakraborty S, Mittal P, Calo S (2019) Analyzing federated learning through an adversarial lens. ICML 97:634–643
25. Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V (2020) How to backdoor federated learning. In: AISTATS. Proceedings of machine learning research, vol 108, pp 2938–2948
26. Sun Z, Kairouz P, Theertha Suresh A, Brendan McMahan H (2019) Can you really backdoor federated learning? arXiv:1911.07963
27. Reisizadeh A, Farnia F, Pedarsani R, Jadbabaie A (2020) Robust federated learning: the case of affine distribution shifts. In: NeurIPS
28. Caldas S, Wu P, Li T, Konecny J, Brendan McMahan H, Smith V, Talwalkar A (2018) Leaf: a benchmark for federated settings. arXiv:1812.01097
29. He C et al (2020) FedML: a research library and benchmark for federated machine learning. arXiv:2007.13518
30. Smith V, Chiang C-K, Sanjabi M, Talwalkar AS (2017) Federated multi-task learning. In: NeurIPS
31. Kairouz P et al (2019) Advances and open problems in federated learning. arXiv:1912.04977
32. Li T, Kumar Sahu A, Talwalkar A, Smith V (2019) Federated learning: challenges, methods, and future directions. arXiv:1908.07873
33. Yang Q, Liu Y, Chen T, Tong Y (2019) Federated machine learning: concept and applications. In: ACM transactions on intelligent systems and technology 10.2
34. Charles Z, Konečný J (2021) Convergence and accuracy trade- offs in federated learning and meta-learning. In: AISTATS. Proceedings of machine learning research, vol 108, pp 4519–4529
35. Peaceman DW, Rachford Jr HH (1955) The numerical solution of parabolic and elliptic differential equations. J Soc Ind Appl Math 3.1:28–41
36. Lions P-L, Mercier B (1979) Splitting algorithms for the sum of two nonlinear operators. SIAM J Numer Anal 16.6:964–979
37. Douglas Jr J, Rachford Jr HH (1956) On the numerical solution of heat conduction problems in two and three space variables. Trans Am Math Soc 82.2:421–439
38. Spingarn Jonathan E (1985) Applications of the method of partial inverses to convex programming: decomposition. Math Program 32:199–223
39. Bauschke HH, Kruk SG (2004) Reflection-projection method for convex feasibility problems with an obtuse cone. J Optim Theory Appl 120.3:503–531
40. Anderson DG (1965) Iterative procedures for nonlinear integral equations. J ACM 12.4:547–560
41. Fu A, Zhang J, Boyd S (2020) Anderson accelerated Douglas–Rachford splitting. SIAM J Sci Comput 42.6:A3560–A3583
42. Tyrrell Rockafellar R, Wets RJ-B (1998) Variational analysis. Springer
43. Beck A, Teboulle M (2012) Smoothing and first order methods: a unified framework. SIAM J Optim 22:557–580
44. Moreau J (1965) Proximité and dualité in a Hilbertian space. Bulletin de la Société Mathématique de France 93:273–299
45. Bauschke HH, Goebel R, Lucet Y, Wang X (2008) The proximal average: basic theory. SIAM J Optim
46. Bruck RE (1977) On the weak convergence of an ergodic iteration for the solution of variational inequalities for monotone operators in Hilbert space. J Math Anal Appl 61.1:159–164

47. Bauschke HH, Moffat SM, Wang X (2010) The resolvent average for positive semidefinite matrices. Linear Algebra Appl 432.7:1757–1771
48. Yu Y (2013) Better approximation and faster algorithm using the proximal average. In: NeurIPS
49. LeCun Y, Cortes C, Burges CJ (2010) MNIST handwritten digit database. Available under the terms of the creative commons attribution- share alike 3.0 license
50. Lions P-L (1978) Une methode iterative de resolution d'une inequation variationnelle. Isr J Math 31.2:204–208
51. Passty GB (1979) Ergodic convergence to a zero of the sum of monotone operators in Hilbert space. J Math Anal Appl 72(2):383–390
52. Cimmino G (1938) Calcolo Approssimato Per le Soluzioni dei Sistemi di Equazioni Lineari. La Ricerca Scientifica, Series II 9:326–333
53. Louis Lions J, Temam R (1966) Une méthode d'éclatement pes opérateurs et des contraintes en calcul des variations. Comptes rendus mathématiques de l'Académie des Sciences, Paris 263:563–565
54. Auslender A (1969) Méthodes Numériques pour la Résolution des Probl'emes d'Optimisation avec Contraintes. PhD thesis. Faculté des Sciences, Grenoble, France
55. Yu Y, Zheng X, Marchetti-Bowick M, Xing EP (2015) Minimizing nonconvex non-separable functions. AISTATS 38:1107–1115
56. Bauschke HH, Combettes PL, Reich S (2005) The asymptotic behavior of the composition of two resolvents. Nonlinear Anal Theory Methods Appl 60(2):283–301
57. Brézis H, Browder FE (1976) Nonlinear ergodic theorems. Bull Am Math Soc 82(6):959–961
58. Bauschke HH, Combettes PL (2017) Convex analysis and monotone operator theory in Hilbert spaces, 2nd edn. Springer
59. Smith V, Chiang C-K, Sanjabi M, Talwalkar AS (2017) Federated multi-task learning. In: NIPS
60. Zantedeschi V, Bellet A, Tommasi M (2020) Fully decentralized joint learning of personalized models and collaboration graphs. In: AISTATS
61. Liu Yang, Kang Yan, Xing Chaoping, Chen Tianjian, Yang Qiang (2020) A secure federated transfer learning framework. IEEE Intell Syst 35:70–82
62. Jiang Y, Konecný J, Rush K, Kannan S (2019) Improving federated learning personalization via model agnostic meta learning. arXiv:1909.12488
63. Khodak M, Balcan M-F, Talwalkar AS (2019) Adaptive gradient-based meta-learning methods. In: NeurIPS
64. Fallah A, Mokhtari A, Ozdaglar AE (2020) Personalized federated learning with theoretical guarantees: a model-agnostic meta- learning approach. In: NeurIPS
65. Gabay D (1983) Applications of the method of multipliers to variational inequalities. In: Augmented Lagrangian methods: applications to the numerical solution of boundary-value problems, vol 15, pp 299–331
66. Spingarn Jonathan E (1983) Partial inverse of a monotone operator. Appl Math Optim 10:247–265
67. Tyrrell Rockafellar R (2019) Progressive decoupling of linkages in optimization and variational inequalities with elicitable convexity or monotonicity. Set-Valued Var Anal 27:863–893
68. Eckstein Jonathan, Bertsekas Dimitri P (1992) On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math Program 55:293–318