Splitting Algorithms for Federated Learning

Saber Malekmohammadi[1,2], Kiarash Shaloudegi[2], Zeou Hu[1,2], and Yaoliang Yu[1]

[1] School of Computer Science, University of Waterloo, Canada
[2] Noah's Ark Lab, Huawei Technologies, Montreal, Canada
{saber.malekmohammadi, zeou.hu, yaoliang.yu}@uwaterloo.ca
kiarash.shaloudegi@huawei.com

Abstract. Over the past few years, the federated learning (FL) community has witnessed a proliferation of new FL algorithms. However, our understating of the theory of FL is still fragmented, and a thorough, formal comparison of these algorithms remains elusive. Motivated by this gap, we show that many of the existing FL algorithms can be understood from an operator splitting point of view. This unification allows us to compare different algorithms with ease, refine previous convergence results and uncover new algorithmic variants. In particular, our analysis reveals the vital role played by the step size in FL algorithms. We perform numerical experiments on both convex and nonconvex models to validate our findings.

1 Introduction

The accompanying (moderate) computational power of modern smart devices such as phones, watches, home appliances, cars, *etc*., and the enormous data accumulated from their interconnected ecosystem have fostered new opportunities and challenges to train/tailor modern deep models. Federated learning (FL), as a result, has recently emerged as a massively distributed framework that enables training a shared or personalized model without infringing user privacy. Tremendous progress has been made since the seminal work of [31], including algorithmic innovations [23, 47, 38, 35], convergence analysis [19, 25, 20, 29], personalization [30, 13, 12, 48], privacy protection [33, 1], model robustness [6, 3], fairness [17, 32, 24], standardization [9, 16], applications [41, 37], just to name a few. We refer to the excellent surveys [18, 22, 44] and the references therein for the current state of affairs in FL.

The main goal of this work is to take a closer examination of some of the most popular FL algorithms, including FedAvg [31], FedProx [23], and FedSplit [35], by connecting them with the well-established theory of operator splitting in optimization. In particular, we show that FedAvg [31] corresponds to forward-backward splitting and we demonstrate a trade-off in its step size and number of local epochs. FedProx [23], on the other hand, belongs to backward-backward splitting, or equivalently, forward-backward splitting on a related but regularized problem, which has been somewhat overlooked in the literature. Interestingly, our results reveal that the recent personalized model in [13] is exactly the problem that FedProx aims to solve. We show that when the step size in FedProx diminishes (sublinearly fast), then it actually solves the original problem, which is contrary to the observation in [35] where the step size is fixed and confirms again the importance of step size in FL. These results largely clarify the pros and cons of FedProx. Furthermore, FedSplit [35] corresponds to Peaceman-Rachford splitting [36, 28] and we show that its convergence (in theory) heavily hinges on the strong convexity of the objective and hence might be less stable for nonconvex problems.

Inspecting FL through the lens of operator splitting allows us to immediately uncover new FL algorithms, by adapting existing splitting algorithms. Indeed, we show that Douglas-Rachford splitting [14, 28] (more precisely the method of partial inverse [43]) yields a (slightly) slower but more stable variant of FedSplit. Our holistic view of FL algorithms suggests a very natural unification, building on which we show that the aforementioned algorithms reduce to different parameter settings in a grand scheme. We believe this is an important step towards standardizing FL from an algorithmic point of view, in addition to standard datasets, models and evaluation protocols as already articulated in [9, 16]. Our unification also allows one to compare and implement different FL algorithms with ease. We perform experiments on both convex and nonconvex models to validate our findings, and compare the above-mentioned FL algorithms on an equal footing.

We proceed in §2 with some background introduction. Our main contributions are presented in §3, where we connect FL with operator splitting, shed new insights on existing algorithms, suggest new algorithmic variants and refine convergence analysis, and in §4, where we present a unification of current algorithms. We conclude in §5 with some future directions.

2 Background

In this section we recall the federated learning (FL) framework of [31]. We consider m users (edge devices), where the i-th user aims at minimizing a function $f_i : \mathbb{R}^d \to \mathbb{R}, i = 1, \dots, m$, defined on a shared model parameter $\mathbf{w} \in \mathbb{R}^d$. Typically, each user function f_i depends on the respective user's local (private) data $\mathcal{D}_i$.

Following [31], many existing FL algorithms fall into the natural formulation that optimizes the (arithmetic) average performance:

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

The weights λ_i here are nonnegative and w.l.o.g. sum to 1. Below, we assume they are specified *beforehand* and fixed throughout.

To facilitate our discussion, we recall the Moreau envelope and proximal map of a function[3] $f : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$, defined respectively as:

$$\mathsf{M}_f^\eta(\mathbf{w}) = \min_{\mathbf{x}} \; \tfrac{1}{2\eta}\|\mathbf{x}-\mathbf{w}\|_2^2 + f(\mathbf{x}), \qquad \mathsf{P}_f^\eta(\mathbf{w}) = \operatorname*{argmin}_{\mathbf{x}} \; \tfrac{1}{2\eta}\|\mathbf{x}-\mathbf{w}\|_2^2 + f(\mathbf{x}), \tag{2}$$

where $\eta > 0$ is a parameter acting similarly as the step size. We also define the reflector

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

Clearly, when $f = \iota_C$ is the indicator function of a (closed) set $C \subseteq \mathbb{R}^d$, $\mathsf{P}_f^\eta(\mathbf{w}) \equiv \mathsf{P}_C(\mathbf{w})$ is the usual (Euclidean) projection onto the set C while $\mathsf{R}_f^\eta(\mathbf{w}) \equiv \mathsf{R}_C(\mathbf{w})$ is the reflection of $\mathbf{w}$ w.r.t. C.

We are now ready to reformulate our main problem of interest (1) in a product space:

$$\min_{\mathsf{w}\in H} \; \mathsf{f}(\mathsf{w}) = \langle \mathbf{1}, \mathbf{f}(\mathsf{w})\rangle, \;\; \text{where} \;\; \mathbf{f}(\mathsf{w}) := \big(f_1(\mathbf{w}_1), \dots, f_m(\mathbf{w}_m)\big), \tag{4}$$

$\mathbf{1}$ is the vector of all 1s, $H := \{\mathsf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_m) \in \mathbb{R}^{dm} : \; \mathbf{w}_1 = \cdots = \mathbf{w}_m\}$, and we equip with the inner product $\langle \mathsf{w}, \mathsf{z}\rangle := \sum_i \lambda_i \mathbf{w}_i^\top \mathbf{z}_i$. Note that the complement $H^\perp = \{\mathsf{w} : \sum_i \lambda_i \mathbf{w}_i = \mathbf{0}\}$,

$$\mathsf{P}_H(\mathsf{w}) = (\bar{\mathbf{w}}, \dots, \bar{\mathbf{w}}), \;\; \text{where} \;\; \bar{\mathbf{w}} := \sum\nolimits_i \lambda_i \mathbf{w}_i, \;\; \text{and} \;\; \mathsf{R}_H(\mathsf{w}) = (2\bar{\mathbf{w}} - \mathbf{w}_1, \dots, 2\bar{\mathbf{w}} - \mathbf{w}_m). \tag{5}$$

We define the (forward) gradient update map w.r.t. to a (sub)differentiable function f:

$$\mathsf{G}_f^\eta := \mathrm{id} - \eta \cdot \partial f, \qquad \mathbf{w} \mapsto \mathbf{w} - \eta \cdot \partial f(\mathbf{w}). \tag{6}$$

When f is differentiable and convex, we note that $\mathsf{R}_f^\eta = \mathsf{G}_f^\eta \circ \mathsf{P}_f^\eta$ and $\nabla \mathsf{M}_f^\eta = \frac{\mathrm{id} - \mathsf{P}_f^\eta}{\eta}$.

3 FL as operator splitting

Following [35], in this section we interpret existing FL algorithms (FedAvg, FedProx, FedSplit) from the operator splitting point of view. We reveal the importance of the step size, obtain new convergence guarantees, and uncover some new and accelerated algorithmic variants.

3.1 FedAvg as forward-backward splitting

The FedAvg algorithm of [31] is essentially a k-step version of the forward-backward splitting of [8]:

$$\mathsf{w}_{t+1} \leftarrow \mathsf{P}_H \mathsf{G}_{\mathsf{f},k}^{\eta_t} \mathsf{w}_t, \quad \text{where} \quad \mathsf{G}_{\mathsf{f},k}^{\eta_t} := \underbrace{\mathsf{G}_{\mathsf{f}}^{\eta_t} \circ \mathsf{G}_{\mathsf{f}}^{\eta_t} \circ \cdots \circ \mathsf{G}_{\mathsf{f}}^{\eta_t}}_{k \text{ times}}, \tag{7}$$

i.e. we perform k steps of (forward) gradient updates w.r.t. f followed by 1 step of (backward) proximal update w.r.t. H. The forward step $\mathsf{G}_{\mathsf{f},k}^{\eta_t}$ is performed in parallel by the users while the backward

[3] We allow the function f to take ∞ when the input is out of our domain of interest.

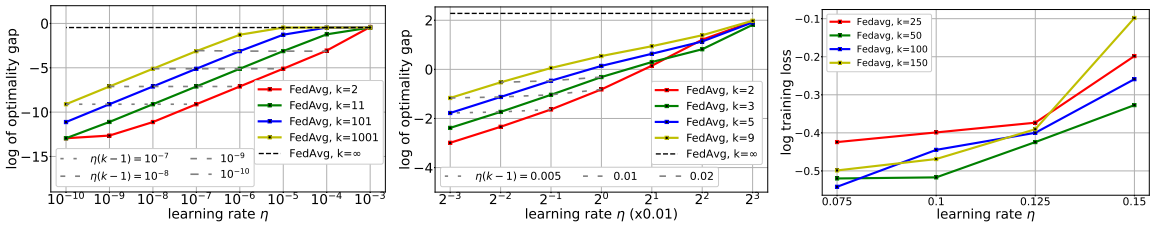

Fig. 1. Optimality gap $\{f(\mathbf{w}^*_{\texttt{FedAvg}}) - f^*\}$ or training loss $\{f(\mathbf{w}^*_{\texttt{FedAvg}})\}$ of (approximate) fixed-point solutions of `FedAvg` for different learning rates η and local epochs k. Different colored lines are for different numbers of local epochs, and dashed lines for different product values $\eta(k-1)$. Left: least squares (closed-form solution); Middle: logistic regression (6000 communication rounds); Right: non-convex CNN on the MNIST dataset (200 communication rounds).

step P_H is performed by the server. Of course, in practice we employ stochastic (i.e. minibatch) approximations of the gradient and sample the users in each communication round.

The number of local steps k turns out to be a key factor: Setting $k = 1$ reduces to the usual (stochastic) forward-backward algorithm and enjoys the well-known convergence guarantees. On the other hand, setting $k = \infty$ (and assuming convergence on each local function f_i) amounts to (repeatedly) averaging the chosen minimizers of f_i's, and eventually converges to the average of minimizers of all m user functions. In general, the performance of the fixed point solution of `FedAvg` appears to depend on the number of local steps k and the step size η. Let us illustrate this with quadratic functions $f_i(\mathbf{w}) = \frac{1}{2}\|A_i\mathbf{w} - \mathbf{b}_i\|_2^2$ and non-adaptive step size $\eta_t \equiv \eta$, where we obtain the fixed-point $\mathbf{w}^*_{\texttt{FedAvg}}(k)$ of `FedAvg` in closed-form [35]:

$$\mathbf{w}^*_{\texttt{FedAvg}}(k) = \big(\textstyle\sum_{i=1}^m A_i^\top A_i \cdot \frac{1}{k}\sum_{j=0}^{k-1}(I-\eta A_i^\top A_i)^j\big)^{-1}\big(\sum_{i=1}^m \frac{1}{k}\sum_{j=0}^{k-1}(I-\eta A_i^\top A_i)^j \cdot A_i^\top \mathbf{b}_i\big). \tag{8}$$

For small η we may apply Taylor expansion and ignore the higher order terms:

$$\tfrac{1}{k}\textstyle\sum_{j=0}^{k-1}(I-\eta A_i^\top A_i)^j \approx \frac{1}{k}\sum_{j=0}^{k-1}(I - j\eta A_i^\top A_i) = I - \frac{\eta(k-1)}{2}A_i^\top A_i. \tag{9}$$

Thus, we observe that the fixed point $\mathbf{w}^*_{\texttt{FedAvg}}(k)$ of `FedAvg` depends on $\eta(k-1)$, up to higher order terms. In particular, when $k = 1$, the fixed point does not depend on η (as long as it is small enough to guarantee convergence), but for $k > 1$, the solution that `FedAvg` converges to does depend on η. Moreover, when η is small, the final performance is almost completely determined by $\eta(k-1)$ in this quadratic setting, as we verify in experiments, see Figure 1 left, where details on generating A_i and $\mathbf{b}_i$ can be found in appendix A.1.

In Figure 1, we run `FedAvg` in (7) on least squares, logistic regression and a non-convex CNN on the MNIST dataset [21]. The experiments[4] are run with pre-determined numbers of communication rounds and different configurations of local epochs k and learning rate η. By examining the dependencies of `FedAvg`'s final performance on k and η we conclude that, in general, smaller local epochs and learning rates gives better final solutions (assuming sufficient communication rounds to ensure convergence). Moreover, for least squares and logistic regression, the product $\eta(k-1)$ we derived in (9) almost completely determines the final performance, when k and η are within a proper range. For the nonconvex CNN (Fig 1, right), especially with limited communications, the approximation in (9) is too crude to be indicative. We note that similar results were also reported in e.g. [29, 10].

3.2 `FedProx` as backward-backward splitting

The recent `FedProx` algorithm [23] replaces the gradient update in `FedAvg` with a proximal update:

$$\mathsf{w}_{t+1} = \mathsf{P}_H\mathsf{P}_{\mathsf{f}}^{\eta_t}\mathsf{w}_t, \tag{10}$$

where as before we may use a minibatch to approximate $\mathsf{P}_{\mathsf{f}}^{\eta}$ or select a subset of users to approximate P_H. Written in this way, it is clear that `FedProx` is an instantiation of the backward-backward splitting algorithm of [27, 34]. In fact, this algorithm traces back to the early works of *e.g.* [11, 26, 2], sometimes

[4] Complete details of our experimental setup are given in Appendix A while all non-convex experimental results are averaged over 4 runs with different random seeds.

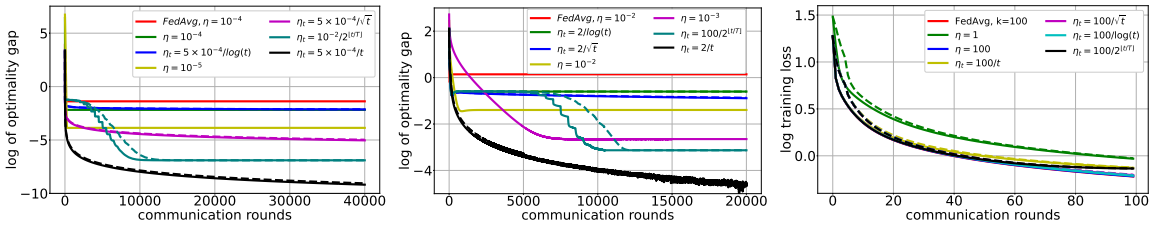

Fig. 2. Effect of step size η and averaging on `FedProx`. Left: least squares; Middle: logistic regression; Right: CNN on MNIST. The dashed and solid lines with the same color show the results obtained with and without the ergodic averaging step in Theorem 1, respectively. For exponentially decaying η_t, we use period T equal to 500 for both least squares and logistic regression experiments, and 10 for CNN experiment.

under the name of the Barycenter method. It was also rediscovered in [45, 46] in the ML community under a somewhat different motivation. The recent work [35] pointed out that `FedProx` does not solve the original problem (1). While technically correct, this conclusion did not take many other subtleties into account, which we explain next.

Following [5], we first note that, with a constant step size $\eta_t \equiv \eta$, `FedProx` is actually equivalent as `FedAvg` but applied to a "regularized" problem:

$$\min_{\mathsf{w}\in H} \ \tilde{\mathsf{f}}(\mathsf{w}), \quad \text{where} \quad \tilde{\mathsf{f}}(\mathsf{w}) := \langle \mathbf{1}, \mathsf{M}^\eta_{\mathbf{f}}(\mathsf{w})\rangle\,, \quad \text{and} \quad \mathsf{M}^\eta_{\mathbf{f}}(\mathsf{w}) = \big(\mathsf{M}^\eta_{f_1}(\mathbf{w}_1), \dots, \mathsf{M}^\eta_{f_m}(\mathbf{w}_m)\big). \tag{11}$$

Interestingly, [13] proposed exactly (11) for the purpose of personalization, which we now realize is automatically achieved if we apply `FedProx` to the original formulation (1). Indeed, $\nabla \mathsf{M}^\eta_{\mathbf{f}}(\mathsf{w}) = [\mathsf{w} - \mathsf{P}^\eta_{\mathbf{f}}(\mathsf{w})]/\eta$ hence $\mathsf{G}^\eta_{\tilde{f}}(\mathsf{w}) = \mathsf{w} - \eta\nabla\tilde{\mathsf{f}}(\mathsf{w}) = \mathsf{P}^\eta_{\mathbf{f}}(\mathsf{w}) = \big(\mathsf{P}^\eta_{f_1}(\mathbf{w}_1), \dots, \mathsf{P}^\eta_{f_m}(\mathbf{w}_m)\big)$. This simple observation turns out to be crucial in understanding `FedProx`.

Another important observation is that[5] as $\eta \to 0$, $\mathsf{M}^\eta_f \to f$ (pointwise or uniformly if f is Lipschitz) while $\mathsf{M}^\eta_f \to \min f$ as $\eta \to \infty$. Therefore as $\eta \to 0$, `FedProx` tends to solve the original problem (1), which can indeed be formalized as follows:

Theorem 1. *Assuming each user participates in every round, the step size η_t is diminishing and non-summable (*i.e. *$\eta_t \to 0$ and $\sum_t \eta_t = \infty$) and the functions $\{f_i\}$ are convex, then the* averaged *iterates $\bar{\mathsf{w}}_t := \frac{\sum_{s=1}^t \eta_s \mathsf{w}_s}{\sum_{s=1}^t \eta_s}$ of `FedProx` converge to a* correct *solution of the* original *problem* (1).

The proof is essentially due to [27, 34], which we merely adapt to our `FL` setting. In Appendix B, we give examples to show that this step size condition is both sufficient and necessary. We emphasize that $\bar{\mathsf{w}}_t$ converges to a solution of the original problem (1), not the regularized problem (11). The subtlety is that we must let the step size η_t approach 0 reasonably slowly, a possibility that was not discussed in [35] where they always fixed the step size η_t to a constant η. Theorem 1 also makes intuitive sense, as $\eta_t \to 0$ slowly, we are effectively tracking the solution of the regularized problem (11) which itself tends to the original problem (1): recall that $\mathsf{M}^\eta_f \to f$ as $\eta \to 0$.

Even the ergodic averaging step in Theorem 1 can be omitted in some cases:

Theorem 2 ([34]). *Under the same assumptions as in Theorem 1, if $\mathbf{f}$ is strongly convex or the solution set of* (1) *has nonempty interior, then the vanilla iterates w_t of `FedProx` also converge.*

We remark that convergence is in fact linear for the second case. Nevertheless, the above two conditions are perhaps not easy to satisfy or verify in most applications. Thus, in practice, we recommend the ergodic averaging in Theorem 1 since it does not create additional overhead (if implemented incrementally) and in our experiments it does not slow down the algorithm noticeably.

In Figure 2, we show the effect of step size η on the convergence of `FedProx`, and compare the results with that of `FedAvg` on both convex and non-convex models. We run `FedProx` with both fixed and diminishing step sizes. In the experiments with diminishing step sizes, we have set the initial value of η (i.e. η_0) to larger values - compared to the constant η values - to ensure that η does not get very small after the first few rounds. From the convex experiments (Fig 2, left and middle), one can see that `FedProx` with a fixed learning rate converges fast (in a few hundred rounds) to a suboptimal solution. In contrast, `FedProx` with diminishing η converges slower, but to better quality solutions.

[5] These results are classic and well-known, see *e.g.* [40].

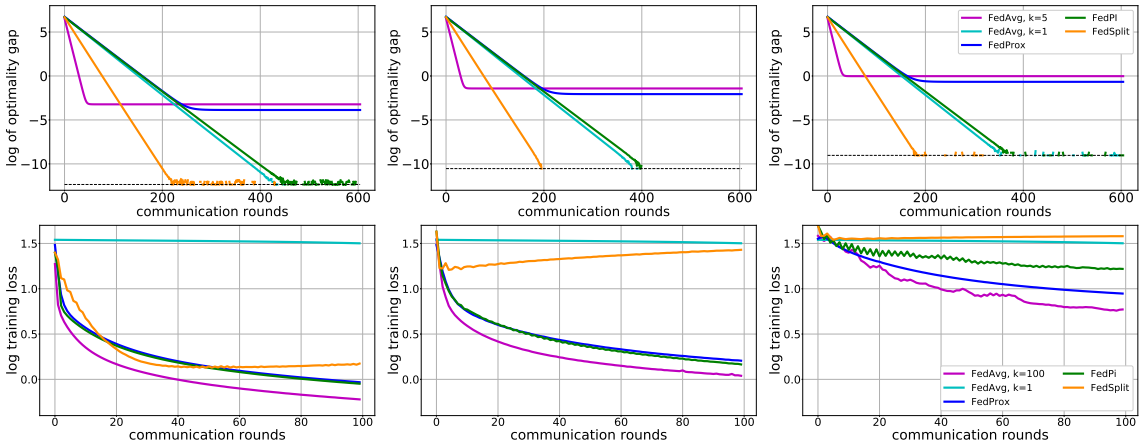

Fig. 3. The effect of data heterogeneity on the performance of different splitting methods; top row: least squares problem. bottom row: nonconvex CNN model. Top-Left: small data heterogeneity with $H \approx 119 \times 10^3$; Top-Middle: moderately data heterogeneity with $H \approx 7.61 \times 10^6$; Top-Right: large data heterogeneity with $H \approx 190.3 \times 10^6$. Bottom-Left: i.i.d. data distribution; Bottom-Middle: non-i.i.d. data distribution with maximum 6 classes per user; Bottom-Right: non-i.i.d. data distribution with maximum 2 classes per user.

It is interesting to note that only when the conditions of Theorem 1 are satisfied (i.e. η diminishes neither too fast nor too slow), `FedProx` converges to a correct solution of the original problem (1), *e.g.* see the results for $\eta_t \propto 1/t$ which satisfies both conditions in Theorem 1. Surprisingly, for the nonconvex setting (Figure 2, right), the best results are achieved with larger learning rates. A similar observation about `FedAvg` in nonconvex settings was reported in [31, Fig. 5 & 6]. [6] Moreover, from the results on both convex and nonconvex models, one can see that ergodic averaging does not affect the convergence rate of `FedProx`.

3.3 `FedSplit` as Peaceman-Rachford splitting

Pathak and Wainwright [35] introduced the `FedSplit` algorithm recently:

$$\mathsf{w}_{t+1} = \mathsf{R}_H \mathsf{R}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t, \tag{12}$$

which is essentially an instantiation of the Peaceman-Rachford splitting algorithm [36, 28]. As shown by [28], `FedSplit` converges to a correct solution of (1) if $\mathbf{f}$ is strictly convex, and the convergence rate is linear if $\mathbf{f}$ is strongly convex and smooth (and η is small). Pathak and Wainwright [35] also studied the convergence behaviour of `FedSplit` when the reflector $\mathsf{R}_{\mathsf{f}}^{\eta}$ is computed approximately. However, we note that convergence behaviour of `FedSplit` is not known or widely studied for nonconvex problems. In particular, we have the following surprising result:

Theorem 3. *If the reflector R_f^{η} is a (strict) contraction, then f must be strongly convex.*

The converse is true if f is also smooth and η is small [28, 15]. Therefore, for non-strongly convex or nonconvex problems, we cannot expect `FedSplit` to converge linearly (if it converges at all).

3.4 `FedPi` as Douglas-Rachford splitting

A popular alternative to the Peaceman-Rachford splitting is the Douglas-Rachford splitting [14, 28], which, to our best knowledge, has not been applied to the `FL` setting. The resulting update, which we call `FedPi`, can be written succinctly as:

$$\mathsf{w}_{t+1} = \frac{\mathsf{w}_t + \mathsf{R}_H \mathsf{R}_{\mathsf{f}}^{\eta_t} \mathsf{w}_t}{2}, \tag{13}$$

i.e. we simply average the current iterate and that of `FedSplit` evenly. Strictly speaking, the above algorithm is a special case of the Douglas-Rachford splitting and was rediscovered by [42] under the name of partial inverse (hence our name `FedPi`). The moderate averaging step in (13) makes `FedPi` much more stable:

[6] Note that the results of `FedAvg` and `FedProx` for $\eta = 100$ and $100/\log(t)$ overlap with each other.

Table 1. A unifying framework (14)-(16) for FL. Note that (a) FedAvg replaces the proximal update $\mathsf{P}_{\mathsf{f}}^{\eta}$ with a gradient update $\mathsf{G}_{\mathsf{f},k}^{\eta}$; (b) ? indicates properties that remain to be studied; (c) "sampling" refers to selecting a subset of users while "stochastic" refers to updating with stochastic gradient.

Algorithm	α	β	γ	$\eta_t \equiv \eta$	$\eta_t \to 0, \sum_t \eta_t = \infty$	nonconvex	sampling	stochastic
FedAvg	1	1	1	eq. (1)	eq. (1)	✓	✓	✓
FedProx	1	1	1	eq. (11)	eq. (1)	✓	✓	✓
FedSplit	2	2	1	eq. (1)	–	?	?	?
FedPi	2	2	$\frac{1}{2}$	eq. (1)	–	✓	?	?

Theorem 4 ([42, 28]). *Assuming each user participates in every round, the step size $\eta_t \equiv \eta$ is constant, and the functions $\{f_i\}$ are convex, then the vanilla iterates w_t of* ***FedPi*** *converge to a* correct *solution of the* original *problem* (1).

Compared to FedSplit, FedPi imposes less stringent condition on f_i. However, when f_i is indeed strongly convex and smooth, as already noted by [28], FedPi will be slower than FedSplit by a factor close to $\sqrt{2}$ (assuming the constant step size is set appropriately for both). More importantly, it may be easier to analyze FedPi on nonconvex functions, as recently demonstrated by [39].

We remark that in FedProx we need the step size η_t to diminish in order to converge to a solution of the original problem (1) whereas FedPi achieves the same with a constant step size η, although at the cost of doubling the memory cost at the server side.

In Figure 3, we compare the performance of different splitting algorithms and how they respond to different degrees of user heterogeneity. We use least squares and a convolutional neural network (CNN) model on MNIST for the convex and nonconvex experiments, respectively. The details of the experimental setup are described in Appendix A.1. For the convex setting, as expected, FedSplit, FedPi, and FedAvg with $k = 1$ achieve the smallest optimality gaps. The performance of all the algorithms deteriorates as users' data become more heterogeneous (see Appendix A.1). For the nonconvex setting, the best results can be achieved by FedProx, FedPi and FedAvg with a big k. It is noteworthy that in the non-convex setting, the performance of FedAvg with $k = 1$ is significantly worse than that with $k = 100$.

4 Unification

The operator splitting view of FL not only allows us to compare and understand the many existing FL algorithms but also opens the door for unifying them. Indeed, we now introduce a grand scheme that unifies all aforementioned FL algorithms:

$$\mathsf{z}_{t+1} = (1-\alpha_t)\mathsf{u}_t + \alpha_t \mathsf{P}_{\mathsf{f}}^{\eta_t}(\mathsf{u}_t) \tag{14}$$

$$\mathsf{w}_{t+1} = (1-\beta_t)\mathsf{z}_{t+1} + \beta_t \mathsf{P}_H(\mathsf{z}_{t+1}) \tag{15}$$

$$\mathsf{u}_{t+1} = (1-\gamma_t)\mathsf{u}_t + \gamma_t \mathsf{w}_{t+1}. \tag{16}$$

Table 1 confirms that the FL algorithms discussed in Section 3 are all special cases of this unifying scheme, which not only provides new (adaptive) variants but also clearly reveals the similarities and differences between seemingly different algorithms. In appendix C, we study the effect of α, β and γ and found that γ mostly affects the convergence speed: the closer γ is to 1, the faster the convergence is, while α and β mostly determine the final optimality gap: the closer they are *both* to 2 (as in FedSplit and FedPi), the considerably smaller the final optimality gap is. However, setting only one of them close to 2 only has a minor effect on optimality gap or convergence speed.

5 Conclusions

We have connected FL with the established theory of operator splitting, revealed new insights on existing algorithms and suggested new algorithmic variants and analysis. Our unified view makes it easy to understand, compare and implement different FL algorithms in a streamlined and standardized fashion. Our experiments demonstrate some interesting differences in the convex and nonconvex settings, and in the early and late communication rounds. In the future we plan to study the effect of stochasticity and extend our analysis to nonconvex functions.

Bibliography

[1] Augenstein, S., McMahan, H.B., Ramage, D., Ramaswamy, S., Kairouz, P., Chen, M., Mathews, R., y Arcas, B.A.: Generative models for effective ml on private, decentralized datasets. In: ICLR (2020), https://openreview.net/forum?id=SJgaRA4FPH

[2] Auslender, A.: Méthodes Numériques pour la Résolution des Problèmes d'Optimisation avec Contraintes. Ph.D. thesis, Faculté des Sciences, Grenoble, France (1969)

[3] Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: AISTATS. Proceedings of Machine Learning Research, vol. 108, pp. 2938–2948 (2020), http://proceedings.mlr.press/v108/bagdasaryan20a.html

[4] Bauschke, H.H., Combettes, P.L.: Convex Analysis and Monotone Operator Theory in Hilbert Spaces. Springer, 2nd edn. (2017), https://link.springer.com/book/10.1007/978-3-319-48311-5

[5] Bauschke, H.H., Combettes, P.L., Reich, S.: The asymptotic behavior of the composition of two resolvents. Nonlinear Analysis: Theory, Methods & Applications **60**(2), 283–301 (2005), https://doi.org/10.1016/j.na.2004.07.054

[6] Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: ICML. vol. 97, pp. 634–643 (2019), http://proceedings.mlr.press/v97/bhagoji19a.html

[7] Brézis, H., Browder, F.E.: Nonlinear ergodic theorems. Bulletin of the American Mathematical Society **82**(6), 959–961 (1976), https://projecteuclid.org:443/euclid.bams/1183538367

[8] Bruck, R.E.: On the weak convergence of an ergodic iteration for the solution of variational inequalities for monotone operators in hilbert space. Journal of Mathematical Analysis and Applications **61**(1), 159–164 (1977), https://doi.org/10.1016/0022-247X(77)90152-4

[9] Caldas, S., Wu, P., Li, T., Konecny, J., McMahan, H.B., Smith, V., Talwalkar, A.: Leaf: A benchmark for federated settings (2018), https://arxiv.org/abs/1812.01097, arXiv:1812.01097

[10] Charles, Z., Konečný, J.: Convergence and accuracy trade-offs in federated learning and meta-learning. In: AISTATS. Proceedings of Machine Learning Research, vol. 108, pp. 4519–4529 (2021), http://proceedings.mlr.press/v130/charles21a/charles21a.pdf

[11] Cimmino, G.: Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari. La Ricerca Scientifica, Series II **9**, 326–333 (1938)

[12] Diao, E., Ding, J., Tarokh, V.: Heterofl: Computation and communication efficient federated learning for heterogeneous clients. In: ICLR (2021), https://openreview.net/forum?id=TNkPBBYFkXg

[13] Dinh, C.T., Tran, N., Nguyen, J.: Personalized federated learning with moreau envelopes. In: NeurIPS. pp. 21394–21405 (2020), https://proceedings.neurips.cc/paper/2020/file/f4f1f13c8289ac1b1ee0ff176b56fc60-Paper.pdf

[14] Douglas, Jr., J., Rachford, Jr., H.H.: On the numerical solution of heat conduction problems in two and three space variables. Transactions of the American Mathematical Society **82**(2), 421–439 (1956), https://doi.org/10.2307/1993056

[15] Gabay, D.: Applications of the method of multipliers to variational inequalities **15**(9), 299–331 (1983), https://doi.org/10.1016/S0168-2024(08)70034-1

[16] He, C., Li, S., So, J., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., Shen, L., et al.: FedML: A research library and benchmark for federated machine learning (2020), https://arxiv.org/abs/2007.13518, arXiv:2007.13518

[17] Hu, Z., Shaloudegi, K., Zhang, G., Yu, Y.: Fedmgda+: Federated learning meets multi-objective optimization (2020), https://arxiv.org/abs/2006.11489, arXiv:2006.11489

[18] Kairouz, P., McMahan, H.B., Avent, B.: Advances and open problems in federated learning (2019), https://arxiv.org/abs/1912.04977, arXiv:1912.04977

[19] Khaled, A., Mishchenko, K., Richtárik, P.: First analysis of local gd on heterogeneous data (2020), https://arxiv.org/abs/1909.04715, arXiv:1909.04715

[20] Khaled, A., Mishchenko, K., Richtarik, P.: Tighter theory for local sgd on identical and heterogeneous data. In: AISTATS. Proceedings of Machine Learning Research, vol. 108, pp. 4519–4529 (2020), http://proceedings.mlr.press/v108/bayoumi20a.html

[21] LeCun, Y., Cortes, C., Burges, C.: Mnist handwritten digit database (2010), http://yann.lecun.com/exdb/mnist, available Under the Terms of the Creative Commons Attribution-Share Alike 3.0 License

[22] Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: Challenges, methods, and future directions (2019), https://arxiv.org/abs/1908.07873, arXiv:1908.07873

[23] Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: Proceedings of Machine Learning and Systems. vol. 2, pp. 429–450 (2020), https://proceedings.mlsys.org/paper/2020/file/38af86134b65d0f10fe33d30dd76442e-Paper.pdf
[24] Li, T., Sanjabi, M., Beirami, A., Smith, V.: Fair resource allocation in federated learning. In: ICLR (2020), https://openreview.net/forum?id=ByexElSYDr
[25] Li, X., Huang, K., Yang, W., Wang, S., Zhang, Z.: On the convergence of fedavg on non-iid data. In: ICLR (2020), https://openreview.net/forum?id=HJxNAnVtDS
[26] Lions, J.L., Temam, R.: 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 (1966), https://gallica.bnf.fr/ark:/12148/bpt6k6426017v/f241
[27] Lions, P.L.: Une methode iterative de resolution d'une inequation variationnelle. Israel Journal of Mathematics **31**(2), 204–208 (1978), https://doi.org/10.1007/BF02760552
[28] Lions, P.L., Mercier, B.: Splitting algorithms for the sum of two nonlinear operators. SIAM Journal on Numerical Analysis **16**(6), 964—-979 (1979), https://doi.org/10.1137/0716071
[29] Malinovskiy, G., Kovalev, D., Gasanov, E., Condat, L., Richtarik, P.: From local sgd to local fixed-point methods for federated learning. In: ICML. vol. 119, pp. 6692–6701 (2020), http://proceedings.mlr.press/v119/malinovskiy20a.html
[30] Mansour, Y., Mohri, M., Ro, J., Suresh, A.T.: Three approaches for personalization with applications to federated learning (2020), https://arxiv.org/abs/2002.10619, arXiv:2002.10619
[31] McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: AISTATS. vol. 54, pp. 1273–1282 (2017), http://proceedings.mlr.press/v54/mcmahan17a/mcmahan17a.pdf
[32] Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. In: ICML. vol. 97, pp. 4615–4625 (2019), http://proceedings.mlr.press/v97/mohri19a.html
[33] Nasr, M., Shokri, R., Houmansadr, A.: 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 (2019), 10.1109/SP.2019.00065
[34] Passty, G.B.: Ergodic convergence to a zero of the sum of monotone operators in hilbert space. Journal of Mathematical Analysis and Applications **72**(2), 383–390 (1979), https://doi.org/10.1016/0022-247X(79)90234-8
[35] Pathak, R., Wainwright, M.J.: Fedsplit: An algorithmic framework for fast federated optimization. In: NeurIPS (2020), https://proceedings.neurips.cc//paper/2020/hash/4ebd440d99504722d80de606ea8507da-Abstract.html
[36] Peaceman, D.W., Rachford, Jr., H.H.: The numerical solution of parabolic and elliptic differential equations. Journal of the Society for Industrial and Applied Mathematics **3**(1), 28—-41 (1955), https://www.jstor.org/stable/2098834
[37] Qiang, Y.: Federated recommendation systems. In: IEEE International Conference on Big Data. pp. 1–1 (2019), 10.1109/BigData47090.2019.9005952
[38] Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečny, J., Kumar, S., McMahan, H.B.: Adaptive Federated Optimization (2020), https://arxiv.org/abs/2003.00295, arXiv:2003.00295
[39] Rockafellar, R.T.: Progressive decoupling of linkages in optimization and variational inequalities with elicitable convexity or monotonicity. Set-Valued and Variational Analysis **27**, 863—893 (2019), https://doi.org/10.1007/s11228-018-0496-1
[40] Rockafellar, R.T., Wets, R.J.B.: Variational Analysis. Springer (1998), https://link.springer.com/book/10.1007/978-3-642-02431-3
[41] Smith, V., Chiang, C.K., Sanjabi, M., Talwalkar, A.S.: Federated multi-task learning. In: NeurIPS (2017), https://papers.nips.cc/paper/2017/hash/6211080fa89981f66b1a0c9d55c61d0f-Abstract.html
[42] Spingarn, J.E.: Partial inverse of a monotone operator. Applied Mathematics and Optimization **10**, 247—-265 (1983), https://doi.org/10.1007/BF01448388
[43] Spingarn, J.E.: Applications of the method of partial inverses to convex programming: Decomposition. Mathematical Programming **32**, 199—-223 (1985), https://doi.org/10.1007/BF01586091
[44] Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology **10**(2) (2019), https://doi.org/10.1145/3298981
[45] Yu, Y.: Better approximation and faster algorithm using the proximal average. In: NeurIPS (2013), https://proceedings.neurips.cc/paper/2013/file/49182f81e6a13cf5eaa496d51fea6406-Paper.pdf
[46] Yu, Y., Zheng, X., Marchetti-Bowick, M., Xing, E.P.: Minimizing nonconvex non-separable functions. In: AISTATS. vol. 38, pp. 1107–1115 (2015), http://proceedings.mlr.press/v38/yu15.html

[47] Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., Khazaeni, Y.: Bayesian nonparametric federated learning of neural networks. In: ICML. vol. 97, pp. 7252–7261 (2019), http://proceedings.mlr.press/v97/yurochkin19a.html
[48] Zhang, M., Sapra, K., Fidler, S., Yeung, S., Alvarez, J.M.: Personalized federated learning with first order model optimization. In: ICLR (2021), https://openreview.net/forum?id=ehJqJQk9cw

Appendix for *Splitting Algorithms for Federated Learning*

A Experimental Setup

In this section we provide more experimental details that are deferred from the main paper.

A.1 Experimental setup: Least Squares and Logistic Regression

For simulating an instance of the aforementioned least squares and logistic regression problems, we follow the experimental setup of [35].

Least squares regression: We consider a set of m devices with local loss functions $f_i(\mathbf{w}) := \frac{1}{2}\|A_i\mathbf{w}-\mathbf{b}_i\|_2^2$, and the main goal is to solve the following minimization problem:

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

where $\mathbf{w}\in\mathbb{R}^d$ is the optimization variable. For each user i, the response vector $\mathbf{b}_i \in \mathbb{R}^{n_i}$ is related to the design matrix $A_i \in \mathbb{R}^{n_i\times d}$ via the linear model

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

where $\boldsymbol{\epsilon}_i \sim N(0,\sigma^2 I_{n_i})$ for some $\sigma > 0$ is the noise vector. The design matrix $A_i \in \mathbb{R}^{n_i\times d}$ is generated by sampling its elements from a standard normal, $A_i^{k,l} \sim N(0,1)$. For the least squares experiments, we instantiated the problem with the following set of parameters:

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

Binary logistic regression: There are m users and the design matrices $A_i \in \mathbb{R}^{n_i\times d}$ for $i = 1,\ldots,m$ are generated as described before. Each user i has a label vector $\mathbf{b}_i \in \{-1,1\}^{n_i}$.The conditional probability of observing $\mathbf{b}_{ij} = 1$ (the j-th label in $\mathbf{b}_i$) is

$$\mathbf{P}\{\mathbf{b}_{ij}=1\} = \frac{e^{\mathbf{a}_{ij}^\top\mathbf{w}_0}}{1+e^{\mathbf{a}_{ij}^\top\mathbf{w}_0}}, \quad j=1,\ldots,n_i,$$

where $\mathbf{a}_{ij}$ is the j-th row of A_i. Also, $\mathbf{w}_0 \in \mathbb{R}^d$ is fixed and sampled from $N(0,1)$. Having generated the design matrices A_i and sampled the labels $\mathbf{b}_i$ for all users, we find the maximum likelihood estimate of $\mathbf{w}_0$ by solving the following convex program, which has a solution $\mathbf{w}_*$:

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

Following [35], we set

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

Data heterogeneity measure: We adopt the data heterogeneity measure of [20] to quantify the amount of heterogeneity in users' data for the least squares and logistic regression problems by

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

where $\mathbf{w}_*$ is a minimizer of the original problem (1). When users' data is homogeneous, all the local functions f_i have the same minimizer of $\mathbf{w}_*$ and $H = 0$. In general, the more heterogeneous the users' data is, the larger H becomes.

Other parameters: Through the experiments, we used local learning rate $\eta = 10^{-5}$ for least squares, and $\eta = 10^{-2}$ for logistic regression, unless otherwise specified. In addition, `FedAvg` is run with $k = 5$ for both least squares and logistic regression, unless otherwise specified.

A.2 Experimental setup: MNIST datasets

We consider a distributed setting with 20 users. In order to create a non-i.i.d. dataset, we follow a similar procedure as in [31]: first we split the data from each class into several shards. Then, each user is randomly assigned a number of shards of data. For example, in Figure 3 to guarantees that no user receives data from more than 6 classes, we split each class of MNIST into 12 shards (i.e., a total of 120 shards for the whole dataset), and each user is randomly assigned 6 shards of data. By considering 20 users, this procedure guarantees that no user receives data from more than 6 classes and the data distribution of each user is different from each other. The local datasets are balanced–all users have the same amount of training samples. The local data is split into train, validation, and test sets with percentage of 80%, 10%, and 10%, respectively. In this way, each user has 2400 data points for training, 300 for test, and 300 for validation. We use a simple 2-layer CNN model with ReLU activation, the detail of which can be found in Table 2. To update the local models at each user using its local data, unless otherwise is stated, we apply gradient descent with $\eta = 0.01$ for `FedAvg`, and gradient descent with $k = 100$ and $\eta = 0.01$ for proximal update in the splitting algorithms.

Table 2. Details of the CNN model on MNIST.

Layer	Output Shape	# of Trainable 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	$(20, 8, 8)$	5020	ReLU	kernel size =5; strides=$(1, 1)$
MaxPool2d	$(20, 4, 4)$	0		pool size=$(2, 2)$
Flatten	320	0		
Dense	20	6420	ReLU	
Dense	10	210	softmax	
Total		11910		

B Proofs

We first recall the following convenient result:

Lemma 1 ([7], [4, Theorem 5.36]). *Let $\{\mathbf{w}_t\}$ and $\{\mathbf{z}_t\}$ be two sequences in $\mathbb{R}^d$, $\emptyset \neq \mathsf{F} \subseteq \mathbb{R}^d$, and $W_k := \operatorname{cl}\operatorname{conv}(\mathbf{w}_t : t \geq k)$. Suppose that*

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

Then, the sequence $\{\mathbf{z}_t\}$ has at most one limit point in F. Therefore, if additionally

3. *all limit points of $\{\mathbf{z}_t\}$ lie in F,*

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

Below, we use well-known properties about firm nonexpansions and Fejér monotone sequences, see the excellent book [4] for background.

Theorem 1. *Assuming each user participates in every round, the step size η_t is diminishing and non-summable (*i.e. *$\eta_t \to 0$ and $\sum_t \eta_t = \infty$) and the functions $\{f_i\}$ are convex, then the* averaged *iterates $\bar{\mathsf{w}}_t := \frac{\sum_{s=1}^t \eta_s \mathsf{w}_s}{\sum_{s=1}^t \eta_s}$ of* **`FedProx`** *converge to a* correct *solution of the* original *problem* (1).

Proof. We simply verify Lemma 1.

Let $\mathsf{w} \in \operatorname{dom}\mathsf{f} \cap H$, $\mathbf{a}^* \in \partial\mathsf{f}(\mathsf{w})$ and $\mathbf{b}^* \in H^\perp$. Applying the firm nonexpansiveness of $\mathsf{P}_\mathsf{f}^{\eta_t}$:

$$\begin{aligned}
\|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 &= \|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{P}_\mathsf{f}^{\eta_t}(\mathsf{w} + \eta_t\mathbf{a}^*)\|_2^2 && (17)\\
&\leq \|\mathsf{w}_t - \mathsf{w} - \eta_t\mathbf{a}^*\|_2^2 - \|\mathsf{w}_t - \mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \eta_t\mathbf{a}^*\|_2^2 && (18)\\
&= \|\mathsf{w}_t - \mathsf{w}\|_2^2 - \|\mathsf{w}_t - \mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t\|_2^2 + 2\eta_t\langle \mathsf{w} - \mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t; \mathbf{a}^*\rangle, && (19)\\
\|\mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 &\leq \|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 - \|\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t\|_2^2 + 2\eta_t\langle \mathsf{w} - \mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t; \mathbf{b}^*\rangle. && (20)
\end{aligned}$$

Summing the above two inequalities and applying the inequality $-\|\mathbf{x}\|_2^2 + 2\langle \mathbf{x}; \mathbf{y}\rangle \leq \|\mathbf{y}\|_2^2$ repeatedly:

$$\|\mathsf{P}_H\mathsf{P}_\mathsf{f}^{\eta_t}\mathsf{w}_t - \mathsf{w}\|_2^2 \leq \|\mathsf{w}_t - \mathsf{w}\|_2^2 + 2\eta_t\langle \mathsf{w} - \mathsf{w}_t; \mathbf{a}^* + \mathbf{b}^*\rangle + \eta_t^2[\|\mathbf{a}^* + \mathbf{b}^*\|_2^2 + \|\mathbf{a}^*\|_2^2]. \tag{21}$$

Summing over t and rearranging we obtain for any $\mathsf{w} \in \operatorname{dom}\mathsf{f} \cap H, \mathsf{w}^* = \mathbf{a}^* + \mathbf{b}^*$:

$$2\langle \mathsf{w} - \bar{\mathsf{w}}_t; \mathsf{w}^*\rangle + [\|\mathbf{a}^*\|_2^2 + \|\mathsf{w}^*\|_2^2] \sum_{k=0}^t \eta_k^2/\Lambda_t \geq (\|\mathsf{w}_{t+1} - \mathsf{w}\|_2^2 - \|\mathsf{w}_1 - \mathsf{w}\|_2^2)/\Lambda_t, \tag{22}$$

where $\Lambda_t := \sum_{k=1}^t \eta_k$. Using the assumptions on η_t we thus know

$$\liminf_{t\to\infty} \ \langle \mathsf{w} - \bar{\mathsf{w}}_t; \mathsf{w}^*\rangle \geq 0. \tag{23}$$

Since w is arbitrary and w^* is chosen to be its subdifferential, it follows that any limit point of $\{\bar{\mathsf{w}}_t\}$ is a solution of the original problem (1), *i.e.* condition 3 of Lemma 1 holds. If $\{\bar{\mathsf{w}}_t\}$ is bounded, then $\mathsf{F} \neq \emptyset$, which we assume now. Let $\mathsf{w} \in \mathsf{F}$ and set $\mathsf{w}^* = \mathbf{0}$ we know from (21) that $\{\mathsf{w}_t\}$ is quasi-Fejér monotone w.r.t. F (*i.e.* condition 1 in Lemma 1 holds). Lastly, let $\bar{\eta}_{t,k} := \eta_k/\Lambda_t$ and we verify condition (II) in Lemma 1:

$$\begin{aligned}
\mathsf{dist}(\bar{\mathsf{w}}_t, W_k) &\leq \left\| \bar{\mathsf{w}}_t - \sum_{s=k}^t \bar{\eta}_{t,s}\mathsf{w}_s / \sum_{\kappa=k}^t \bar{\eta}_{t,\kappa} \right\|_2 && (24)\\
&\leq \sum_{\kappa=0}^{k-1} \bar{\eta}_{t,\kappa} \left[\|\mathsf{w}_\kappa\|_2 + \left\| \sum_{s=k}^t \bar{\eta}_{t,s}\mathsf{w}_s \right\|_2 / \sum_{\kappa=k}^t \bar{\eta}_{t,\kappa} \right] && (25)\\
&\overset{t\to\infty}{\longrightarrow} 0, && (26)
\end{aligned}$$

since w_t is bounded and for any k, $\bar{\eta}_{t,k} \to 0$ as $t \to \infty$. All three conditions in Lemma 1 are now verified.

We remark that the step size condition is tight, as shown by the following simple example:

Example 1. Let $f_\pm(w) = \frac{1}{2}(w \pm 1)^2$. Simple calculation verifies that

$$\mathsf{P}^\eta_{f_\pm}(w) = \tfrac{w \mp \eta}{1+\eta}. \tag{27}$$

Therefore, the iterates of `FedProx` for the two functions f_+ and f_- are:

$$w_{t+1} = \tfrac{w_t}{1+\eta_t} = \prod_{\tau=0}^{t} \tfrac{1}{1+\eta_\tau} w_0, \tag{28}$$

which tends to the true minimizer $w_\star = 0$ for any w_0 iff

$$\prod_{\tau=0}^{t} \tfrac{1}{1+\eta_\tau} \to 0 \iff \sum_t \eta_t \to \infty. \tag{29}$$

If we consider instead f_+ and $2f_-$, then

$$2w_{t+1} = \tfrac{w_t - \eta_t}{1+\eta_t} + \tfrac{w_t + 2\eta_t}{1+2\eta_t}. \tag{30}$$

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

$$2 = \tfrac{1-3\eta}{1+\eta} + \tfrac{1+6\eta}{1+2\eta} \iff 2(1+\eta)(1+2\eta) = (1-3\eta)(1+2\eta) + (1+6\eta)(1+\eta) \tag{31}$$

$$\iff \eta = 0, \tag{32}$$

contradiction. Therefore, it is necessary to have $\eta_t \to 0$ in order for `FedProx` to converge to the true solution $w_\star = \frac{1}{3}$ on this example.

Theorem 3. *If the reflector R^η_f is a (strict) contraction, then f must be strongly convex.*

Proof. Since $\mathsf{R}^\eta_f = \mathsf{R}_{\eta f}$ and f is strongly convex iff ηf is so, w.l.o.g. we may take $\eta = 1$. Suppose R_f is γ-contractive for some $\gamma \in (0, 1)$, *i.e.* for all $\mathbf{w}$ and $\mathbf{z}$:

$$\|\mathsf{R}_f \mathbf{w} - \mathsf{R}_f \mathbf{z}\|_2 \leq \gamma \cdot \|\mathbf{w} - \mathbf{z}\|_2. \tag{33}$$

It then follows that the proximal map $\mathsf{P}_f = \frac{\mathrm{id} + \mathsf{R}_f}{2}$ is $\frac{1+\gamma}{2}$-contractive. Moreover, $\frac{2}{1+\gamma}\mathsf{P}_f$, being nonexpansive, is the gradient of the convex function $\frac{2}{1+\gamma}\mathsf{M}_{f^\star}$. Thus, $\frac{2}{1+\gamma}\mathsf{P}_f$ is actually firmly nonexpansive [4, Corollary 18.17]. But,

$$\tfrac{2}{1+\gamma}\mathsf{P}_f = \tfrac{2}{1+\gamma}(\mathrm{id} + \partial f)^{-1} = [\mathrm{id} + (\partial f - \tfrac{1-\gamma}{1+\gamma}\mathrm{id}) \circ \tfrac{1+\gamma}{2}\mathrm{id}]^{-1}, \tag{34}$$

and hence $(\partial f - \frac{1-\gamma}{1+\gamma}\mathrm{id}) \circ \frac{1+\gamma}{2}\mathrm{id}$ is maximal monotone [4, Proposition 23.8], *i.e.* f is $\frac{1-\gamma}{1+\gamma}$-strongly convex.

Table 3. Explanation for the parameter tuning in Fig. 4. For each adaptive tuning experiment, we change the adaptive parameter values from round 0 to round T. After round T, all parameters are fixed (to the values as in `FedSplit`).

Algorithm	α	β	γ	adaptive parameters setting
α adaptive	adaptive	2	1	$\alpha = \min(1 + \frac{t}{T}, 2)$
β adaptive	2	adaptive	1	$\beta = \min(1 + \frac{t}{T}, 2)$
γ adaptive	2	2	adaptive	$\gamma = \min(\frac{1}{2} + \frac{t}{2T}, 1)$
α, β adaptive	adaptive	adaptive	1	$\alpha = \beta = \min(1 + \frac{t}{T}, 2)$
α, β, γ adaptive	adaptive	adaptive	adaptive	$\alpha = \beta = \min(1 + \frac{t}{T}, 2)$ and $\gamma = \frac{\alpha}{2}$

C Over-relaxation

The parameters α_t, β_t and γ_t are fixed to various constants in existing `FL` algorithms (see Table 1), but of course in practice we may adaptively adjust them. In general, we keep the following principle in mind: choosing these parameters in $[0, 1]$ leads to more stable and potentially slower variants (known as under-relaxation in numerical analysis) while choosing them in $[1, 2]$ leads to potentially faster but less stable variants (known as over-relaxation). To see why over-relaxation may accelerate convergence, let us use (14) as an example. Reversing time, a (backward descending) proximal update becomes a (forward ascending) gradient update:

$$\tilde{\mathsf{u}}_t := \mathsf{P}_{\mathsf{f}}^{\eta_t}(\mathsf{u}_t) \iff \mathsf{u}_t = \tilde{\mathsf{u}}_t + \eta_t \cdot \partial \mathsf{f}(\tilde{\mathsf{u}}_t), \tag{35}$$

whence follows from (14) that

$$\mathsf{z}_{t+1} = \tilde{\mathsf{u}}_t - (\alpha_t - 1)\eta_t \cdot \partial \mathsf{f}(\tilde{\mathsf{u}}_t). \tag{36}$$

Thus, if $\alpha_t \geq 1$, through over-relaxation we obtain a free gradient *descent* update on top of the proximally updated $\tilde{\mathsf{u}}_t$, which could further decrease the function value of f. When f is L-smooth and σ-strongly convex, a suitable choice is to set

$$(\alpha_t - 1)\eta_t = \tfrac{2}{\sigma + L} \implies \alpha_t = 1 + \tfrac{2}{\eta_t(\sigma + L)}. \tag{37}$$

In contrast, when $\alpha_t \in [0, 1]$, through under-relaxation we obtain a free gradient *ascent* step which is likely to increase the function value of f. Nevertheless, averaging does make the update more stable. To summarize, in practice we recommend starting with relaxation parameters (*i.e.* $\alpha_t, \beta_t, \gamma_t$) less than 1 and gradually increasing them to 2 as we approach the solution.

Effect of over-relaxation As can be seen from Fig. 4, setting γ to values close to 1 enhances the speed of convergence to the final solution. However, it does not affect the final optimality gap (final solution quality). On the other hand, α and β mostly determine the final optimality gap in the sense that setting *both* α and β to values close to 2 reduces the final optimality gap considerably (the difference between `FedSplit` and `FedPi` algorithms compared to `FedProx`). However, setting only one of them to values close to 2 has a minor effect on the final optimality gap or the convergence speed.

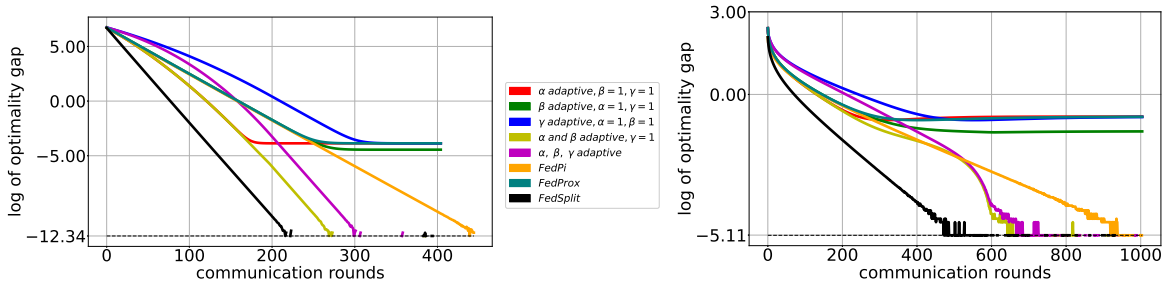

Fig. 4. The effect of parameters α, β and γ on the final optimality gap and the convergence speed of splitting algorithms. Left: least squares; Right: logistic regression. In each experiment, we change the value of adaptive parameters as in Table 3. T is equal to 200 and 600, for least squares and logistic regression, respectively.