From Local SGD to Local Fixed-Point Methods for Federated Learning

Grigory Malinovsky [1] **Dmitry Kovalev** [2] **Elnur Gasanov** [2] **Laurent Condat** [2] **Peter Richtárik** [2]

Abstract

Most algorithms for solving optimization problems or finding saddle points of convex–concave functions are fixed-point algorithms. In this work we consider the generic problem of finding a fixed point of an average of operators, or an approximation thereof, in a distributed setting. Our work is motivated by the needs of federated learning. In this context, each local operator models the computations done locally on a mobile device. We investigate two strategies to achieve such a consensus: one based on a fixed number of local steps, and the other based on randomized computations. In both cases, the goal is to limit communication of the locally-computed variables, which is often the bottleneck in distributed frameworks. We perform convergence analysis of both methods and conduct a number of experiments highlighting the benefits of our approach.

1. Introduction

In the 'big data' era, the explosion in size and complexity of the data arises in parallel to a shift towards distributed computations, as modern hardware increasingly relies on the power of uniting many parallel units into one system. For distributed optimization tasks, specific issues arise, such as decentralized data storage. For instance, the huge amount of mobile phones or smart home devices in the world contain an important volume of data captured and stored on each of them. This data contains a wealth of potentially useful information to their owners, and more so if appropriate machine learning models could be trained on the heterogeneous data stored across the network of such devices. Yet, many users are increasingly sensitive to privacy concerns and prefer their data to never leave their devices. But the only way to share knowledge while not having all data in one place is to communicate, to keep moving towards the solution of the overall problem. Typically, mobile phones communicate back and forth with a distant server, so that a global model is progressively improved and converges to a steady state, which is globally optimal for all users. This is precisely the purpose of the recent and rising paradigm of *federated learning* (Konečný et al., 2016; McMahan et al., 2017) where typically a global supervised model is trained in a massively distributed manner over a network of *heterogeneous* devices. Communication, which can be costly and slow, is the main bottleneck in this framework. So, it is of primary importance to devise novel algorithmic strategies, where the computation and communication loads are balanced.

A strategy increasingly used by practitioners is to make use of *local computations*; that is, more local computations are performed on each device before communication and subsequent model averaging, with the hope that this will reduce the total number of communications needed to obtain a globally meaningful solution. Thus, local gradient descent methods have been investigated (Stich, 2019; Khaled et al., 2019; 2020; Ma et al., 2017; Haddadpour & Mahdavi, 2019). Despite their practical success, local methods are little understood and there is much to be discovered. In this paper, we don't restrict ourselves to gradient descent to minimize an average of smooth functions; we consider the much broader setting of finding a fixed point of an average of a large number M of operators. Indeed, most, if not all, iterative methods are fixed-point methods, which aim at finding a fixed point of some operator (Bauschke et al., 2011; Condat et al., 2019). Fixed-point methods are typically made from compositions and averages of gradient or proximity operators of functions (Combettes & Yamada, 2015; Bauschke & Combettes, 2017); for instance, a sum of proximity operators corresponds to the 'proximal average' of functions (Yu, 2013). Using more involved Lyapunov functions than the distance to the solution or the objective value, convergence of methods with inertia, e.g. Nesterov's acceleration techniques, to a fixed point, can be established (Lessard et al., 2016). (Block-)coordinate or alternating minimization methods are fixed-point methods as well (Richtárik & Takáč, 2014; Pesquet & Repetti, 2015). Let us also mention that by the design of nontrivial fixed-point operators, nonlinear inverse problems can be solved (Combettes &

[1]Moscow Institute of Physics and Technology, Dolgoprudny, Russia [2]King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia. Correspondence to: Laurent Condat <see https://lcondat.github.io/>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Woodstock, 2020). Beyond optimization, fixed-point methods are used to solve monotone inclusions or variational inequalities, with applications in mechanics or stochastic control. They are also used to find saddle points of convex–concave functions, e.g. Nash equilibria in game theory. Yet another example is attaining the steady-state of a control system or a dynamic phenomenon modeled by a PDE.

1.1. Contributions

We model the setting of communication-efficient distributed fixed-point optimization as follows: we have $M \geq 1$ parallel computing nodes. The variables handled by these nodes are modeled as vectors of the Euclidean space $\mathbb{R}^d$, endowed with the classical inner product, for some $d \geq 1$. Let $\mathcal{T}_i$, for $i = 1, \dots, M$ be operators on $\mathbb{R}^d$, which model the set of operations during one iteration. We define the average operator

$$\mathcal{T} : x \in \mathbb{R}^d \mapsto \frac{1}{M} \sum_{i=1}^{M} \mathcal{T}_i(x). \tag{1}$$

Our goal is to find a fixed point of $\mathcal{T}$; that is, a vector $x^\star \in \mathbb{R}^d$ such that $\mathcal{T}(x^\star) = x^\star$. The sought solution $x^\star$ should be obtained by repeatedly applying $\mathcal{T}_i$ at each node, in parallel, with averaging steps to achieve a consensus. Here we consider that, after some number of iterations, each node communicates its variable to a distant server, synchronously. Then the server computes the average of the received vectors and broadcasts it to all nodes.

We investigate two strategies. The first one consists, for each computing node, in iterating several times some sequence of operations; we call this *local steps*. The second strategy consists in reducing the number of communication steps by sharing information only with some low probability, and doing only local computations inbetween. We analyze two algorithms, which instantiate these two ideas, and we prove their convergence. Their good performances are illustrated by experiments.

1.2. Mathematical Background

Let T be an operator on $\mathbb{R}^d$. We denote by $\mathrm{Fix}(T)$ the set of its fixed points. T is said to be χ-Lipschitz continuous, for some $\chi \geq 0$, if, for every x and y in $\mathbb{R}^d$,

$$\|T(x) - T(y)\| \leq \chi \|x - y\|.$$

Moreover, T is said to be nonexpansive if it is 1-Lipschitz continuous, and χ-contractive, if it is χ-Lipschitz continuous, for some $\chi \in [0, 1)$. If T is contractive, its fixed point exists and is unique, see the Banach–Picard Theorem 1.50 in (Bauschke & Combettes, 2017). T is said to be α-averaged, for some $\alpha \in (0, 1]$, if $T = \alpha T' + \mathrm{Id}$ for some nonexpansive operator T', where Id denotes the identity. T is said to be firmly nonexpansive if it is $1/2$-averaged.

2. A Generic Distributed Fixed-Point Method with Local Steps

Let $(t_n)_{n\in\mathbb{N}}$ be the sequence of integers at which communication occurs. We propose Algorithm 1, shown below; it proceeds as follows: at every iteration, the operator $\mathcal{T}_i$ is applied at node i, with further relaxation with parameter λ. After some number of iterations, each of the M computing nodes communicates its vector to a master node, which computes their average and broadcasts it to all nodes. Thus, the later resume computing at the next iteration from the same variable $\hat{x}^k$. The algorithm is a generalization of local gradient descent, a.k.a. Federated Averaging (McMahan et al., 2017).

We call an *epoch* a sequence of local iterations, followed by averaging; that is, the n-th epoch, for $n \geq 1$, is the sequence of iterations of indices $k + 1 = t_{n-1} + 1, \dots, t_n$ (the 0-th epoch is the initialization step $x_i^0 := \hat{x}^0$, for $i = 1, \dots, M$). We assume that the number of iterations in each epoch, between two aggregation steps, is bounded by some integer $H \geq 1$; that is,

Assumption 2.1. *$1 \leq t_n - t_{n-1} \leq H$, for every $n \geq 1$.*

To analyze Algorithm 1, we introduce the following averaged vector:

$$\hat{x}^k = \frac{1}{M} \sum_{i=1}^{M} x_i^k = \frac{1}{M} \sum_{i=1}^{M} h_i^k.$$

Note that this vector is actually computed only when k is one of the t_n. In the uniform case $t_n = nH$, for every $n \in \mathbb{N}$, we introduce the operator

$$\widetilde{\mathcal{T}}_\lambda = \frac{1}{M} \sum_{i=1}^{M} \big(\lambda \mathcal{T}_i + (1 - \lambda)\mathrm{Id}\big)^H,$$

where $\cdot^H$ denotes the composition of an operator with itself H times. Thus, $x_1^{nH} = \cdots = x_M^{nH} = \hat{x}^{nH}$ is the variable shared by every node at the end of the n-th epoch. We have, for every $n \in \mathbb{N}$,

$$\hat{x}^{(n+1)H} = \widetilde{\mathcal{T}}_\lambda(\hat{x}^{nH}).$$

We also assume that the following holds:

Assumption 2.2. *$\mathrm{Fix}(\mathcal{T})$ and $\mathrm{Fix}(\widetilde{\mathcal{T}}_\lambda)$ are nonempty.*

Note that the fixed points of $\widetilde{\mathcal{T}}_\lambda$ depend on λ. The smaller λ, the closer $\mathrm{Fix}(\mathcal{T})$ and $\mathrm{Fix}(\widetilde{\mathcal{T}})$. But the smaller λ, the slower the convergence, so λ controls the tradeoff between accuracy and speed in estimating a fixed point of $\mathcal{T}$.

2.1. General Convergence Analysis

Theorem 2.3 (General convergence). *Suppose that $t_n = nH$, for every $n \in \mathbb{N}$, and suppose that the $\mathcal{T}_i$ are all α-averaged, for some $\alpha \in (0, 1]$. Let $\lambda \in (0, 1/\alpha)$ be the*

Algorithm 1 Local fixed-point method

Input: Initial estimate $\hat{x}^0 \in \mathbb{R}^d$, stepsize $\lambda > 0$, sequence of synchronization times $0 = t_0 < t_1 < \dots$
Initialize: $x_i^0 := \hat{x}^0$, for all $i = 1, \dots, M$
for $k = 0, 1, \dots$ **do**
  for $i = 1, 2, \dots, M$ in parallel **do**
    $h_i^{k+1} := (1-\lambda)x_i^k + \lambda \mathcal{T}_i(x_i^k)$
    if $k+1 = t_n$, for some $n \in \mathbb{N}$, **then**
      Communicate h_i^{k+1} to master node
    else
      $x_i^{k+1} := h_i^{k+1}$
    end if
  end for
  if $k+1 = t_n$, for some $n \in \mathbb{N}$, **then**
    At master node: $\hat{x}^{k+1} := \frac{1}{M}\sum_{i=1}^M h_i^{k+1}$
    Broadcast: $x_i^{k+1} := \hat{x}^{k+1}$, for all i
  end if
end for

Algorithm 2 Randomized fixed-point method

Input: Initial estimate $\hat{x}^0 \in \mathbb{R}^d$, stepsize $\lambda > 0$, communication probability $0 < p \leq 1$
Initialize: $x_i^0 = \hat{x}^0$, for all $i = 1, \dots, M$
for $k = 1, 2, \dots$ **do**
  for $i = 1, 2, \dots, M$ in parallel **do**
    $h_i^{k+1} := (1-\lambda)x_i^k + \lambda \mathcal{T}_i(x_i^k)$
  end for
  Flip a coin and
  with probability p **do**
    Communicate h_i^{k+1} to master, for all i
    At master node: $\hat{x}^{k+1} := \frac{1}{M}\sum_{i=1}^M h_i^{k+1}$
    Broadcast: $x_i^{k+1} := \hat{x}^{k+1}$, for all i
  else, with probability $1-p$, **do**
    $x_i^{k+1} := h_i^{k+1}$, for all $i = 1, \dots, M$
end for

parameter in Algorithm 1. Then the sequence $(\hat{x}^{nH})_{n\in\mathbb{N}}$ converges to a fixed point $x^\dagger$ of $\widetilde{\mathcal{T}}$. In addition, the following hold:

(i) $\widetilde{\mathcal{T}}_\lambda$ is ζ-averaged, with $\zeta = \frac{H\alpha\lambda}{1+(H-1)\alpha\lambda}$.

(ii) The distance between $\hat{x}^{nH}$ and $x^\dagger$ decreases at every epoch: for every $n \in \mathbb{N}$,

$$\|\hat{x}^{(n+1)H} - x^\dagger\|^2 \leq \|\hat{x}^{nH} - x^\dagger\|^2 - \frac{1-\zeta}{\zeta}\|\hat{x}^{(n+1)H} - \hat{x}^{nH}\|^2. \tag{2}$$

(iii) The squared differences between two successive updates are summable:

$$\sum_{n\in\mathbb{N}} \|\hat{x}^{(n+1)H} - \hat{x}^{nH}\|^2 \leq \frac{\zeta}{1-\zeta}\|\hat{x}^0 - x^\dagger\|^2. \tag{3}$$

(iv) For every $n \in \mathbb{N}$,

$$\|\hat{x}^{(n+1)H} - \hat{x}^{nH}\|^2 \leq \frac{1}{\zeta(1-\zeta)(n+1)}\|\hat{x}^0 - x^\dagger\|^2. \tag{4}$$

(v)
$$\|\hat{x}^{(n+1)H} - \hat{x}^{nH}\|^2 = o(1/n). \tag{5}$$

Proof. The convergence property and the property *(iii)* come from the application of the Krasnosel'skii–Mann theorem, see Theorem 5.15 in (Bauschke & Combettes, 2017). The properties *(i)* and *(ii)* are applications of Proposition 4.46, Proposition 4.42, and Proposition 4.35 in (Bauschke & Combettes, 2017). *(iv)* and *(v)* come from Theorem 1 in (Davis & Yin, 2016). □

We can note that in most cases, the fixed-point residual $\|T(x^k) - x^k\|$ is a natural way to measure the convergence speed of a fixed-point algorithm $x^{k+1} = T(x^k)$. For gradient descent, $T(x^k) = x^k - \gamma\nabla F(x^k)$, so we have $\|T(x^k) - x^k\| = \gamma\|\nabla F(x^k)\|$, which indeed measures the discrepancy to $\nabla F(x^\star) = 0$. For the proximal point algorithm to solve a monotone inclusion $0 \in M(x^\star)$, $T(x^k) = (\gamma M + \mathrm{Id})^{-1}(x^k)$, so that $T(x^k) - x^k \in -\gamma M(x^{k+1})$; again, $\|T(x^k) - x^k\|$ characterizes the discrepancy to the solution.

Remark 2.4 (Convergence speed). *For the baseline algorithm (Algorithm 1 with $H = 1$), where averaging occurs after every iteration, we have after H iterations:*

$$\|\hat{x}^{(n+1)H} - x^\dagger\|^2 \leq \|\hat{x}^{nH} - x^\dagger\|^2 - \frac{1-\alpha\lambda}{\alpha\lambda}\sum_{k=nH}^{(n+1)H-1}\|\hat{x}^{k+1} - \hat{x}^k\|^2. \tag{6}$$

We can compare this 'progress', made in decreasing the squared distance to the solution, with the one in Theorem 2.3-(ii), where $\frac{1-\zeta}{\zeta} = \frac{1-\alpha\lambda}{H\alpha\lambda}$. This latter value multiplies $\|\hat{x}^{(n+1)H} - \hat{x}^{nH}\|^2$, which can be up to H^2 larger than $\|\hat{x}^{k+1} - \hat{x}^k\|^2$, for k in $nH, \dots, (n+1)H - 1$. So, in favorable cases, Algorithm 1 progresses as fast as the baseline algorithm. In less favorable cases, the progress in one epoch is H times smaller, corresponding to the progress in 1 iteration. Given that communication occurs only once per epoch, the ratio of convergence speed to communication burden is, roughly speaking, between 1 and H times better than the one of the baseline algorithm. They don't converge to the same elements, however.

A complementary result on the convergence speed is the following. In the rest of the section, the t_n are not restricted

to be uniform; we assume that Assumption (2.1) holds, as well as:

Assumption 2.5. *Each operator $\mathcal{T}_i$ is firmly nonexpansive.*

Then we have the following results on the iterates of Algorithm 1:

Theorem 2.6. *Suppose that $\lambda \leq \frac{1}{8\max(1,H-1)}$. Then $\forall\, T \in \mathbb{N}$,*

$$\frac{1}{T}\sum_{k=0}^{T-1}\left\|\hat{x}^k - \mathcal{T}(\hat{x}^k)\right\|^2 \leq \frac{3\|\hat{x}^0 - x^\star\|^2}{\lambda T} + \frac{36\lambda^2(H-1)^2}{M}\sum_{i=1}^{M}\|x^\star - \mathcal{T}_i(x^\star)\|^2. \tag{7}$$

The next result gives us an explicit complexity, in terms of number of iterations sufficient to achieve ε-accuracy:

Corollary 2.7. *Suppose that $H \geq 2$ and that $\lambda \leq \frac{1}{8}$. Then a sufficient condition on the number T of iterations to reach ε-accuracy, for any $\varepsilon > 0$, is*

$$\frac{T}{H-1} \geq \frac{24\|\hat{x}^0 - x^\star\|^2}{\varepsilon}\max\left\{2, \frac{3\sigma}{\sqrt{\varepsilon}}\right\}. \tag{8}$$

Note that as long as the target accuracy is not too high, in particular if $\varepsilon \geq \frac{9\sigma^2}{8}$, then $\frac{T}{H} = \mathcal{O}\left(\frac{\|\hat{x}^0 - x^\star\|^2}{\varepsilon}\right)$. If $\varepsilon < \frac{9}{8}\sigma^2$, the communication complexity is equal to $\frac{T}{H} = \mathcal{O}\left(\frac{\|\hat{x}^0 - x^\star\|^2\sigma}{\epsilon^{3/2}}\right)$.

Corollary 2.8. *Let $T \in \mathbb{N}$ and let $H \geq 1$ be such that $H \leq \frac{\sqrt{T}}{\sqrt{M}}$; set $\lambda = \frac{1}{8}\frac{\sqrt{M}}{\sqrt{T}}$. Then*

$$\frac{1}{T}\sum_{k=0}^{T-1}\left\|\hat{x}^k - \mathcal{T}(\hat{x}^k)\right\|^2 \leq \frac{24\|\hat{x}^0 - x^\star\|^2}{\sqrt{MT}} + \frac{3M(H-1)^2\sigma^2}{8T}. \tag{9}$$

Hence, to get a convergence rate of $\frac{1}{\sqrt{MT}}$ we can choose the parameter H as $\mathcal{O}\left(T^{1/4}M^{-3/4}\right)$, which implies a total number of $\Omega\left(T^{3/4}M^{3/4}\right)$ synchronization steps. If we need a rate of $1/\sqrt{T}$, we can set a larger value $H = \mathcal{O}\left(T^{1/4}\right)$.

Remark 2.9 (Case $H = 1$). *We remark that if $H = 1$, i.e. communication occurs after every iteration, the last term in Theorem 2.6, which depends on $H - 1$, is zero. This is coherent with the fact that $x^\dagger = x^\star$ in that case, so that the algorithm converges to an exact fixed point of $\mathcal{T}$. In that sense, Theorem 2.6 is tight.*

Remark 2.10 (Local gradient descent (GD) case). *Consider that $\mathcal{T}_i(x_i^k) = x_i^k - \frac{1}{L}\nabla f_i(x_i^k)$, where each convex function f_i is L-smooth; that is, f_i is differentiable with L-Lipschitz continuous gradient. Then the assumptions in Theorem 2.6 are satisfied and our results recover known results about Local GD for heterogeneous data as particular cases (Khaled et al., 2019).*

2.2. Linear Convergence with Contractive Operators

Theorem 2.11 (Linear convergence). *Suppose that $t_n = nH$, for every $n \in \mathbb{N}$, and suppose that the $\mathcal{T}_i$ are all χ-contractive, for some $\chi \in [0, 1)$. Let $\lambda \in (0, \frac{2}{1+\chi})$ be the parameter in Algorithm 1. Then the the fixed point $x^\dagger$ of $\widetilde{\mathcal{T}}_\lambda$ exists and is unique, and the sequence $(\hat{x}^{nH})_{n\in\mathbb{N}}$ converges linearly to $x^\dagger$. More precisely, the following hold:*

(i) $\widetilde{\mathcal{T}}_\lambda$ is ξ^H-contractive, with $\xi = \max\left(\lambda\chi + (1-\lambda), \lambda(1+\chi) - 1\right)$.

(ii) For every $n \in \mathbb{N}$,

$$\|\hat{x}^{(n+1)H} - x^\dagger\| \leq \xi^H\|\hat{x}^{nH} - x^\dagger\|. \tag{10}$$

(iii) We have linear convergence with rate ξ: for every $n \in \mathbb{N}$,

$$\|\hat{x}^{nH} - x^\dagger\| \leq \xi^{nH}\|\hat{x}^0 - x^\dagger\|. \tag{11}$$

Proof. For every $i = 1, \ldots, M$, the operator $\lambda\mathcal{T}_i + (1-\lambda)\mathrm{Id}$ is ξ-contractive, with $\xi = \{\lambda\chi + (1-\lambda)$ if $\lambda \leq 1$, $\lambda(1+\chi) - 1$ else$\}$. Thus, $(\lambda\mathcal{T}_i + (1-\lambda)\mathrm{Id})^H$ is ξ^H contractive. Furthermore, the average of ξ^H-contractive operators is ξ^H-contractive. The claimed properties are applications of the Banach–Picard theorem (Theorem 1.50 in (Bauschke & Combettes, 2017)). □

Remark 2.12 (Convergence speed). *In the conditions of Theorem 2.11, the convergence rate ξ with respect to the number of iterations is the same, whatever H: the distance to a fixed point is contracted by a factor of ξ after every iteration, in average. The fixed point depends on H, however.*

Remark 2.13 (Choice of λ). *In the conditions of Theorem 2.11, without further knowledge on the operators $\mathcal{T}_i$, we should set $\lambda = 1$, so that $\xi = \chi$, since every other choice may slow down the convergence.*

Since Algorithm 1 converges linearly to $x^\dagger$, it remains to characterize the distance between $x^\dagger$ and $x^\star$.

Theorem 2.14 (Neighborhood of the solution). *In the conditions of Theorem 2.11, suppose that $\lambda = 1$. So, $\xi = \chi$. Then*

$$\|x^\dagger - x^\star\| \leq S, \tag{12}$$

where

$$S = \frac{\xi}{1-\xi}\frac{1-\xi^{H-1}}{1-\xi^H}\frac{1}{M}\sum_{i=1}^{M}\|\mathcal{T}_i(x^\star) - x^\star\|. \tag{13}$$

Remark 2.15 (Comments on Theorem 2.14).

(1) *If $M = 1$, $\mathcal{T}_1 = \mathcal{T}$, so that $\|\mathcal{T}_1(x^\star) - x^\star\| = 0$ and $S = 0$, so that we recover that $x^\dagger = x^\star$, whatever H. In that case, the unique node and the master do not need to communicate, and the variable at the node will converge to $x^\star$. In other words, communication is irrelevant in that case.*

(2) *If $H = 1$, $1 - \xi^{H-1} = 0$ and $S = 0$, so that we recover that $x^\dagger = x^\star$.*

(3) *If $H \to +\infty$, S is finite and we have*

$$S = \frac{\xi}{1-\xi}\frac{1}{M}\sum_{i=1}^{M}\|\mathcal{T}_i(x^\star) - x^\star\|. \tag{14}$$

This corresponds to $x^\dagger = \frac{1}{M}\sum_{i=1}^{M} x_i^\star$, where $x_i^\star$ is the fixed point of T_i.

(4) *If we let H vary from 1 to $+\infty$, S increases monotonically from 0 to the value in (14).*

(5) *In 'one-shot minimization', applying $\mathcal{T}_i$ consists in going to its fixed point: $\mathcal{T}_i(x) = x_i^\star$, for every x. Then $\xi = 0$. Hence, $S = 0$, because $x^\dagger = \frac{1}{M}\sum_{i=1}^{M} x_i^\star = x^\star$.*

(6) *In the homogeneous case $\mathcal{T}_i = \mathcal{T}$ for every i,*

$$S = \frac{\xi}{1-\xi}\frac{1}{M}\sum_{i=1}^{M}\|\mathcal{T}(x^\star) - x^\star\| = 0,$$

since $\mathcal{T}(x^\star) = x^\star$. In this case, the M nodes do the same computations, so this is the same as having only one node, like in (1).

(7) *As a direct corollary of Theorem 2.11 (iii) and Theorem 2.14, we have, for every $n \in \mathbb{N}$,*

$$\|\hat{x}^{nH} - x^\star\| \le \xi^{nH}\|\hat{x}^0 - x^\dagger\| + S \tag{15}$$
$$\le \xi^{nH}(\|\hat{x}^0 - x^\star\| + S) + S. \tag{16}$$

Remark 2.16 (Local gradient descent). *Let us consider that each $\mathcal{T}_i : x \mapsto x - \gamma\nabla F_i(x)$, for some L-smooth and μ-strongly convex function F_i, with $L \ge \mu > 0$ and $0 < \gamma \le 2/(L+\mu)$. Set $\lambda = 1$. Then $\xi = \chi = 1 - \gamma\mu$ and $\|\mathcal{T}_i(x^\star) - x^\star\| = \gamma\|\nabla F_i(x^\star)\|$. To our knowledge, our characterization of the convergence behavior is new and improves upon state-of-the-art results (Khaled et al., 2019), even in this case.*

To summarize, in presence of contractive operators, Algorithm 1 converges at the same rate as the baseline algorithm ($H = 1$), up to a neighborhood of size S, for which we give a tight bound. So, if the desired accuracy $\epsilon = \|\hat{x}^k - x^\star\|$ is not lower than S, using local steps is the way to go, since the communication load is divided by H, chosen as the largest value such that $S \le \epsilon$ in (13).

3. A Randomized Communication-Efficient Distributed Fixed-Point Method

Now, we propose a second loopless algorithm, where the local steps in Algorithm 1, which can be viewed as an inner loop between two communication steps, is replaced by a probabilistic aggregation. This yields Algorithm 2, shown above. It is communication-efficient in the following sense: while in Algorithm 1 the number of communication rounds is divided by H (or by the average of $t_n - t_{n-1}$ in the nonuniform case), in Algorithm 2 it is multiplied by the probability $p \le 1$. Thus, p plays the same role as $1/H$.

To analyze Algorithm 2, we suppose that the operators are contractive:

Assumption 3.1. *Each operator $\mathcal{T}_i$ is $(1+\rho/2)$-cocoercive (Bauschke & Combettes, 2017), with $\rho > 0$; that is, there exists $\rho > 0$ such that, for every $i = 1, \dots, M$ and every $x, y \in \mathbb{R}^d$,*

$$(1+\rho)\|\mathcal{T}_i(x) - \mathcal{T}_i(y)\|^2 \le \|x-y\|^2 - \|x - \mathcal{T}_i(x) - y + \mathcal{T}_i(y)\|^2.$$

In the particular case of gradient descent (GD) as the operator, this assumption is satisfied with $\rho > 0$ for strongly convex smooth functions, see Theorem 2.1.11 in (Nesterov, 2004).

Almost sure linear convergence of Algorithm 2 up to a neighborhood is established in the next theorem:

Theorem 3.2. *Let us define the Lyapunov function: for every $k \in \mathbb{N}$,*

$$\Psi^k := \|\hat{x}^k - x^\star\|^2 + \frac{5\lambda}{p}\frac{1}{M}\sum_{i=1}^{M}\left\|x_i^k - \hat{x}^k\right\|^2 \tag{17}$$

Then, under Assumption 3.1 and if $\lambda < \frac{p}{15}$, we have, for every $k \in \mathbb{N}$,

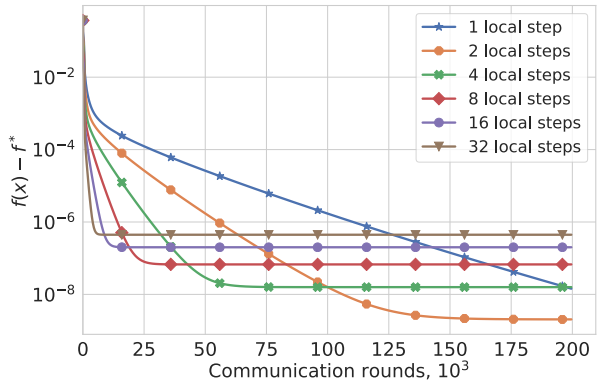
$$\mathbb{E}\Psi^k \le \left(1 - \min\left(\frac{\lambda\rho}{1+\rho}, \frac{p}{5}\right)\right)^k \Psi^0 + \frac{150}{\min\left(\frac{\lambda\rho}{1+\rho}, \frac{p}{5}\right)p^2}\lambda^3\sigma^2, \tag{18}$$

where $\sigma^2 := \frac{1}{M}\sum_{i=1}^{M}\|x^\star - \mathcal{T}_i(x^\star)\|^2$ and $\mathbb{E}$ denotes the expectation.

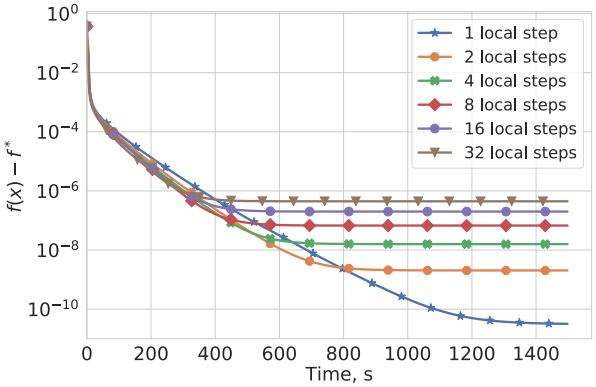
Since the previous theorem may be difficult to analyze, the next results gives a bound to reach ε-accuracy in in Algorithm 2:

Corollary 3.3. *Under Assumption 3.1 and if $\lambda < \frac{p}{15}$, for any $\varepsilon > 0$, ε-accuracy is reached after T iterations, with*

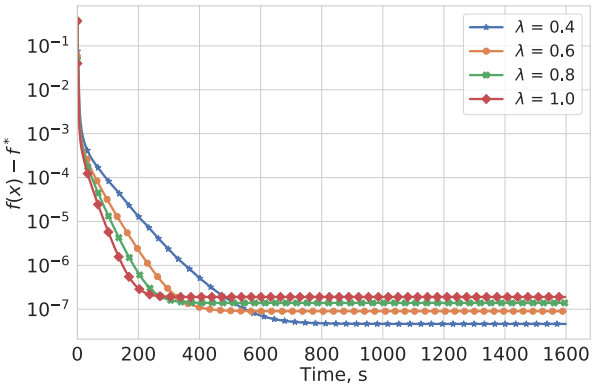
$$T \ge \max\left\{\frac{15(1+\rho)}{\rho p}, \frac{18\sigma(1+\rho)^{\frac{1}{3}}}{p\rho^{\frac{3}{2}}\varepsilon^{\frac{1}{2}}}, \frac{40\sigma^{\frac{2}{3}}(1+\rho)}{p\rho\varepsilon^{\frac{1}{3}}}\right\} \times \log\frac{2\Psi_0}{\varepsilon}. \tag{19}$$



(a)



(b)



(c)

Figure 1. We analyze the convergence of Algorithm 1 with gradient descent steps, with uniform communication times $t_n = nH$; in (a) w.r.t. number of communication rounds, for different values of H, with $\lambda = 0.5$; in (b) w.r.t. computation time, for different values of H, with $\lambda = 0.5$; in (c) w.r.t. computation time, for different values of λ, with $H = 4$.

4. Experiments

Model Although our approach can be applied more broadly, we focus on logistic regression, since this is one of the most important models for classification. The corresponding objective function takes the following form:

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} \log\left(1 + \exp\left(-b_i a_i^\top x\right)\right) + \frac{\kappa}{2}\|x\|^2,$$

where $a_i \in \mathbb{R}^d$ and $b_i \in \{-1, +1\}$ are the data samples.

Datasets We use the 'a9a' and 'a4a' datasets from the LIBSVM library and we set κ to be $\frac{L}{n}$, where n is the size of the dataset and L is a Lipschitz constant of the first part of ∇f, without regularization.

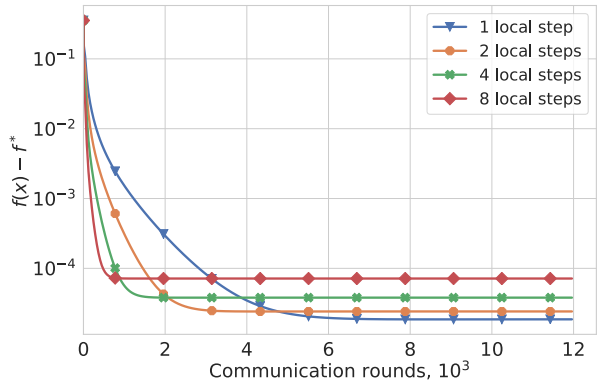
Hardware and software We implemented all algorithms in Python using the package MPI4PY, in order to run the code on a truly parallel architecture. All methods were evaluated on a computer with an Intel(R) Xeon(R) Gold 6146 CPU at 3.20GHz, having 24 cores. The cores are connected to 2 sockets, with 12 cores for each of them.

4.1. Local Gradient Descent

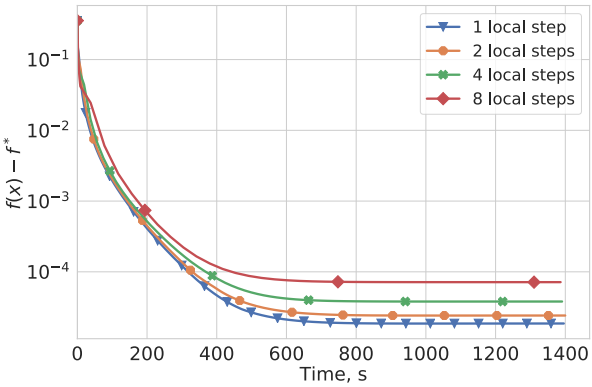
We consider gradient descent (GD) steps as the operators. That is, we consider the problem of minimizing the finite sum:

$$f(x) = \frac{1}{M}\sum_{i=1}^{M} f_i(x), \tag{20}$$

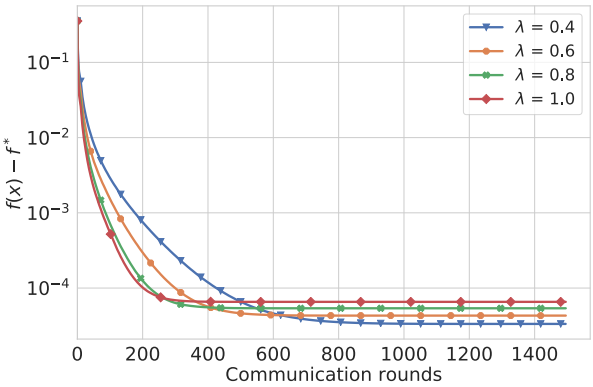
where each function f_i is convex and L-smooth. We set $\mathcal{T}_i(x_i^k) := x_i^k - \frac{1}{L}\nabla f_i(x_i^k)$. We use $\frac{1}{L}$ as the stepsize, so that each $\mathcal{T}_i$ is firmly nonexpansive. The results of Algorithms 1 and 2 are illustrated in Figures 1 and 3, respectively.



(a)



(b)



(c)

Figure 2. We analyze the convergence of Algorithm 1 with cyclic gradient descent steps, with uniform communication times $t_n = nH$; in (a) w.r.t. number of communication rounds, for different values of H, with $\lambda = 0.5$; in (b) w.r.t. computation time, for different values of H, with $\lambda = 0.5$; in (c) w.r.t. computation time, for different values of λ, with $H = 4$.

4.2. Local Cycling Gradient Descent

In this section, we consider another operator, which is cycling GD. So, we consider minimizing the same function as in (20), but this time each function f_i is also a finite sum: $f_i = \frac{1}{N}\sum_{j=1}^{N} f_{ij}$. Instead of applying full gradient steps, we apply N element-wise gradient steps, in the sequential order of the data points. Thus,

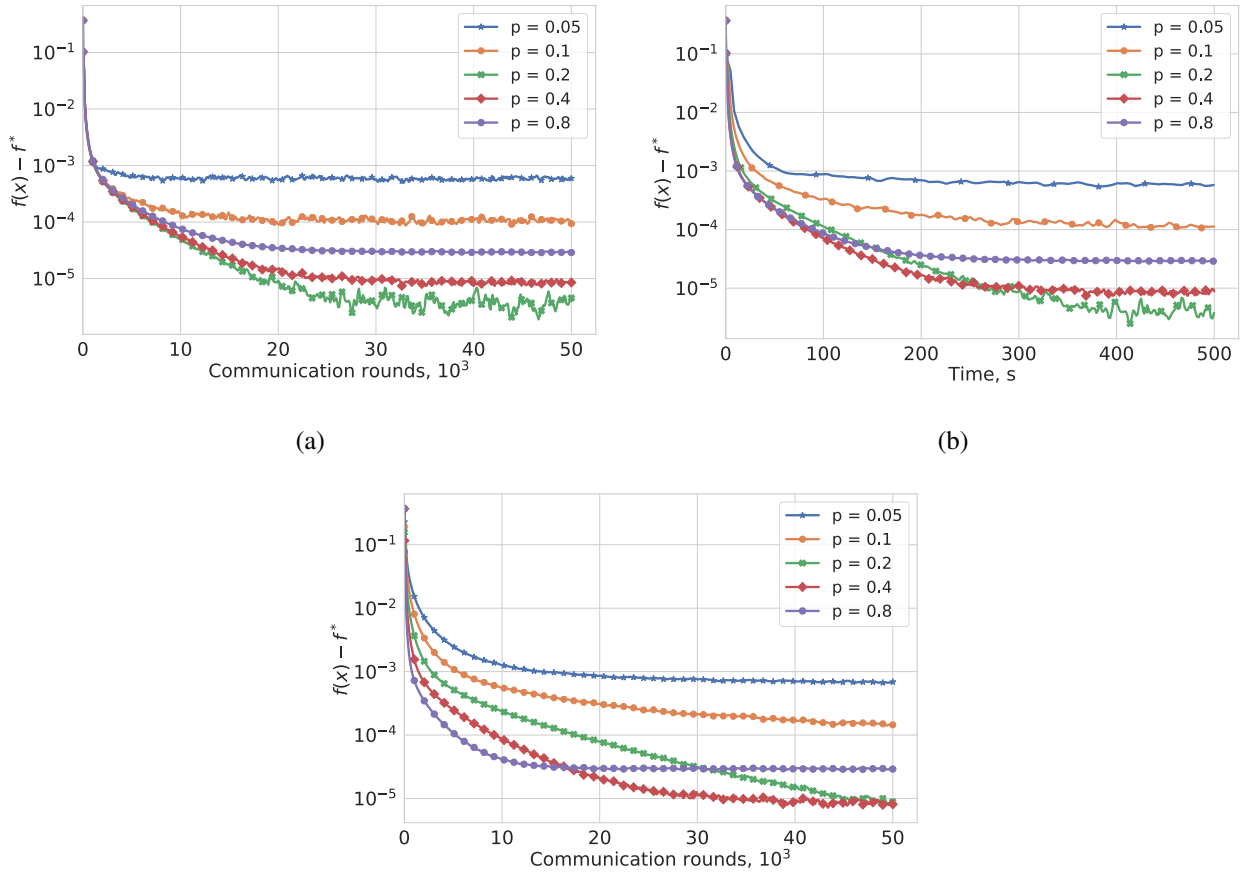
$$\mathcal{T}_i(x_i^k) \coloneqq S_{i1}(S_{i2}(\dots S_{in}(x_i^k))),$$

where $S_{ij} : y \mapsto y - \frac{1}{NL}\nabla f_{ij}$. If, for each i, all functions f_{ij} have the same minimizer $x_i^\star$, then this joint minimizer is a fixed point of $\mathcal{T}_i$. Also, these operators can be shown to be firmly nonexpansive. The results of Algorithms 1 and 2 are illustrated in Figures 2 and 4, respectively.

4.3. Results

We observe a very tight match between our theory and the numerical results. As can be seen, the larger the value of the parameters H and λ, the faster the convergence at the beginning, but the larger the radius of the neighborhood. In terms of computational time, there is no big advantage, since the experiments were run on a single machine and the communication time was negligible. But in a distributed setting where communication is slow, our approach has a clear advantage. We can also observe the absence of oscillations. Hence, there is a clear advantage of local methods when only limited accuracy is required.

In the experiment with cyclic GD, the algorithm converges only to a neighbourhood of the ideal solution, even when 1 local step is used. This happens because the assumption of a joint minimizer for all i is not satisfied here. However, since the operators are firmly nonexpansive, we have con-



(a) (b)

(c)

Figure 3. We analyze the convergence of Algorithm 2 with gradient descent steps, with $\lambda = 0.5$; in (a) with the same gradient stepsizes, w.r.t. number of communication rounds, for different values of p; in (b) with the same gradient stepsizes, w.r.t. computation time, for different values of p; in (c) with gradient stepsizes proportional to p, w.r.t. number of communication rounds, for different values of p.

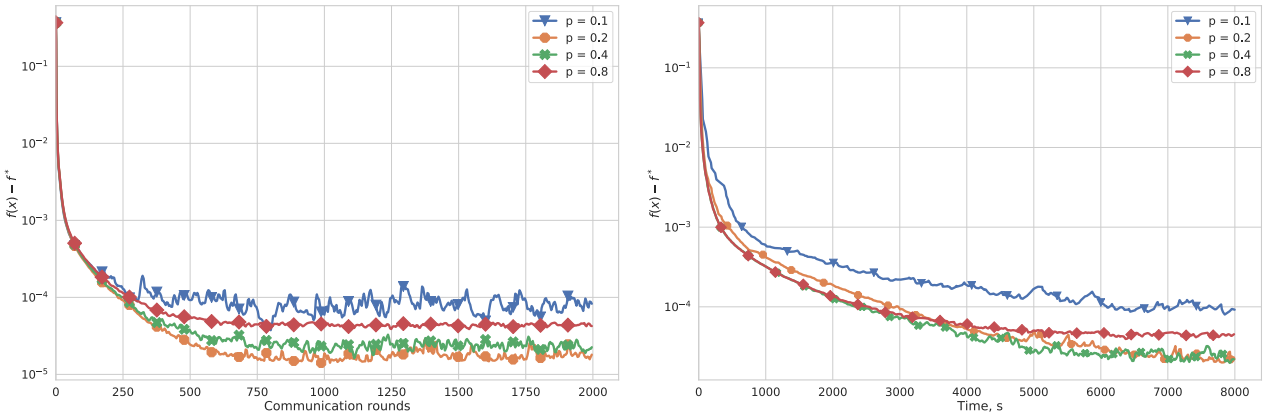
vergence to a fixed point. The convergence of Algorithm 1 is illustrated with respect to the relaxation parameter λ. If λ is small, convergence is slower, but the algorithm converges to a point closer to the true solution $x^\star$. In Figure 4, we further illustrate the behavior of Algorithm 2 with respect to the probability p, for cyclic gradient descent. We can see that the fastest and most accurate convergence is obtained for an intermediate value of p, here $p = 0.2$.

The experiments with Algorithm 2 show that, with a low probability p of update, the neighborhood is substantially larger; however, with p increasing, the convergence in terms of communication rounds becomes worse. Therefore, with careful selection of the probability parameter, a significant advantage can be obtained.

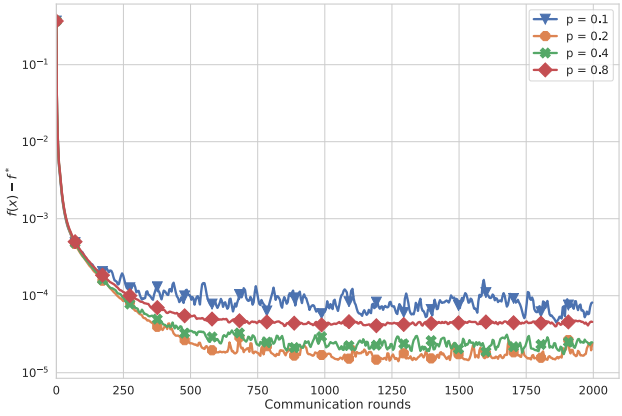
5. Conclusion

We have proposed two strategies to reduce the communication burden in a generic distributed setting, where a fixed point of an average of operators is sought. We have shown that they improve the convergence speed, while achieving the goal of reducing the communication load. At convergence, only an approximation of the ideal fixed point is attained, but if medium accuracy is sufficient, the proposed algorithms are particularly adequate.

In future work, we will generalize the setting to randomized fixed-point operators, to generalize stochastic gradient descent approaches. We will also investigate compression (Khaled & Richtárik, 2019; Chraibi et al., 2019) of the communicated variables, with or without variance reduction, in combination with locality.



(a) (b)



(c)

Figure 4. We analyze the convergence of Algorithm 2 with cyclic gradient descent steps, with $\lambda = 0.5$; in (a) with the same gradient stepsizes, w.r.t. number of communication rounds, for different values of the p; in (b) same as in (a), but w.r.t. computation time; in (c) with gradient stepsizes proportional to p, w.r.t. number of communication rounds, for different values of p.

Acknowledgements

Part of this work was done while the first author was an intern at KAUST.

References

Bauschke, H. H. and Combettes, P. L. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, New York, 2nd edition, 2017.

Bauschke, H. H., Burachik, R. S., Combettes, P. L., Elser, V., Luke, D. R., and Wolkowicz, H. (eds.). *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer, 2011.

Chraibi, S., Khaled, A., Kovalev, D., Richtárik, P., Salim, A., and Takáč, M. Distributed fixed point methods with compressed iterates. *preprint ArXiv:1912.09925*, 2019.

Combettes, P. L. and Woodstock, Z. C. A fixed point framework for recovering signals from nonlinear transformations. preprint arXiv:2003.01260, 2020.

Combettes, P. L. and Yamada, I. Compositions and convex combinations of averaged nonexpansive operators. *Journal of Mathematical Analysis and Applications*, 425(1): 55–70, 2015.

Condat, L., Kitahara, D., Contreras, A., and Hirabayashi, A. Proximal splitting algorithms: Relax them all! preprint arXiv:1912.00137, 2019.

Davis, D. and Yin, W. Convergence rate analysis of several splitting schemes. In Glowinski, R., Osher, S. J., and Yin, W. (eds.), *Splitting Methods in Communication, Imaging, Science, and Engineering*, pp. 115–163, Cham, 2016. Springer International Publishing.

Haddadpour, F. and Mahdavi, M. On the convergence of

local descent methods in federated learning. *preprint arXiv:1910.14425*, 2019.

Khaled, A. and Richtárik, P. Gradient descent with compressed iterates. In *NeurIPS Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.

Khaled, A., Mishchenko, K., and Richtárik, P. First analysis of local GD on heterogeneous data. In *NeurIPS Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local SGD on identical and heterogeneous data. In *The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS 2020)*, 2020.

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

Lessard, L., Recht, B., and Packards, A. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM J. Optim.*, 26(1):57–95, 2016.

Ma, C., Konečný, J., Jaggi, M., Smith, V., Jordan, M. I., Richtárik, P., and Takáč, M. Distributed optimization with arbitrary local solvers. *Optimization Methods and Software*, 32(4):813–848, 2017.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Agüera y Arcas, B. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

Nesterov, Y. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Publishers, 2004.

Pesquet, J.-C. and Repetti, A. A class of randomized primal-dual algorithms for distributed optimization. *J. Nonlinear Convex Anal.*, 12(16), December 2015.

Richtárik, P. and Takáč, M. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Math. Program.*, 144(1–2):1–38, April 2014.

Stich, S. U. Local SGD Converges Fast and Communicates Little. In *International Conference on Learning Representations*, 2019.

Yu, Y.-L. On decomposing the proximal map. In *Proc. of 26th Int. Conf. Neural Information Processing Systems (NIPS)*, pp. 91–99, 2013.