Multistage stepsize schedule in Federated Learning: Bridging Theory and Practice

Charlie Hou [1] **Kiran K. Thekumparampil** [2] **Giulia Fanti** [1] **Sewoong Oh** [3]

Abstract

Stepsize scheduling schemes can improve the convergence speed of Federated Learning (FL) algorithms in practice (Reddi et al., 2020). Similar schemes have also been used in traditional non-federated optimization to obtain algorithms which are oblivious to problem parameters and to improve their convergence speeds (Aybat et al., 2019). However their benefits in the federated setting are not fully characterized. To resolve this, we study a common but general multistage stepsize scheduling scheme which can augment any FL algorithm. We show that for convex problems, this scheme allows us to achieve, for the first-time, a nearly-optimal communication complexity when clients have moderately heterogeneous data, as is observed often in practice. Our results match algorithm-independent lower bounds (Woodworth et al., 2020a), and thus we resolve a recently posed open question (Woodworth et al., 2020a). Additionally, we show that scheduling allows these algorithms to be oblivious to problem parameters. This allows easier tuning of standard FL algorithms, which are often challenging to tune. Finally, we verify our results empirically.

[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA [2]Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana-Champaign, Illinois, USA [3]Allen School of Computer Science and Engineering, University of Washington, Seattle, Washington, USA. Correspondence to: Charlie Hou <charlieh@andrew.cmu.edu>, Kiran K. Thekumparampil <thekump2@illinois.edu>, Giulia Fanti <gfanti@andrew.cmu.edu>, Sewoong Oh <sewoong@cs.washington.edu>.

This work was presented at the International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML 2021 (FL-ICML'21). This workshop does not have official proceedings and this paper is non-archival.

1. Introduction

In federated learning (FL) (McMahan et al., 2017; Kairouz et al., 2019; Li et al., 2020), distributed clients interact with a central server to learn a model without directly sharing their data with the server. In federated minimization, clients are trying to solve the following optimization:

$$F(x) = \min_x \frac{1}{N} \sum_{i=1}^{N} F_i(x) \tag{1}$$

where i indexes the devices, N is the number of devices (or clients), and $F_i(x)$ is a (strongly convex) objective. Typical FL deployments have two properties that make optimizing Eq. (1) challenging: (i) *Data heterogeneity.* Different clients may have different data distributions, so the F_i's are of the form $F_i(x) = \mathbb{E}_{z_i}[f(x; z_i)]$ for some function f, where z_i is the random variable representing client i's data. More formally, we define client heterogeneity ζ as

$$\zeta^2 := \sup_x \frac{1}{N} \sum_{i=1}^{N} \|\nabla F(x) - \nabla F_i(x)\|^2, \tag{2}$$

which captures how different the local client gradients are from the overall global gradient (in the worst-case) (Woodworth et al., 2020a). As in (Koloskova et al., 2020; Karimireddy et al., 2020b; Woodworth et al., 2020a), we define $\zeta_*^2 := \frac{1}{N} \sum_{i=1}^{N} \|\nabla F_i(x^*)\|^2$ as the heterogeneity at the optimum x^*. (ii) *Communication cost.* In many deployments of FL, clients may have limited bandwidth (e.g., mobile devices). Due to these two challenges, most federated optimization algorithms alternate between local rounds of computation, where clients process only their own data to save communication, and global rounds, where clients synchronize with the central server to resolve heterogeneity.

Many federated optimization algorithms navigate this trade-off. A simple baseline is called Minibatch SGD (Woodworth et al., 2020a). In each global round, the central server sends each device the current model, on which each device calculates K stochastic gradients and sends their average to the server. The server averages these to make a minibatch gradient of size NK, and performs a model update. Further, this approach can be accelerated in the Nesterov sense (Ghadimi & Lan, 2012). Woodworth et al. (2020a) recently

showed that Accelerated Minibatch SGD is optimal in the highly-heterogeneous setting where $\zeta_* > \beta D$, where D is the initial distance to the optimum, and the objective functions F_i are β-smooth.

A popular approach to this problem is called Federated Averaging (FedAvg) (McMahan et al., 2017)[1]. Here each client i independently runs SGD on its own objective F_i for K iterations (local updates), and periodically communicates the most recent iterate to the server, which averages the updates. Compared to Accelerated Minibatch SGD, this approach allows clients to learn from local data in between communication rounds. Several algorithms have built on the intuition of FedAvg, including SCAFFOLD (Karimireddy et al., 2020b), MIME (Karimireddy et al., 2020a), FedAdam (Reddi et al., 2020), FedAdagrad (Reddi et al., 2020), FedYogi (Reddi et al., 2020), and S-Local-SGD (Gorbunov et al., 2020). Empirically, these local update algorithms outperform baselines like (Accelerated) Minibatch SGD.

However, there is a gap between these algorithms' empirical gains and their theoretical worst-case guarantees. In theory, none of these methods are known to outperform Minibatch SGD in communication cost in any regime, and it was proven that FedAvg can possibly outperform accelerated Minibatch SGD on convex objectives only in the low-heterogeneity setting: $\zeta_*^2 < \frac{\beta^2 D^2}{R}$, where R is the number of communication rounds (Woodworth et al., 2020a). Additionally, in all but the highly-heterogeneous setting, there is a gap between known convergence rates (including Accelerated Minibatch SGD) and the lower bound of (Woodworth et al., 2020a) for first-order methods (Tab. 1).

We hypothesize that the main reason for this discrepancy is that, in practice (but not in theory) *multistage hyperparameter schedules* are used with federated optimization algorithms. That is, learning rates decay during training, whereas theoretical analysis does not account for this multistage optimization. For instance, the experiments of FedAdam, FedAdagrad, and FedYogi use an exponential learning rate decay schedule (Reddi et al., 2020) to achieve their best performance, which is reminiscent of multistage algorithms of (Aybat et al., 2019; Fallah et al., 2020). Later work of (Charles & Konečnỳ, 2020) in FL also relies on adaptive learning rate decay schedules to outperform previous work on a variety of FL tasks. Unfortunately, analysis of multistaged federated algorithms is still immature.

Today, most analysis of federated optimization algorithms considers constant stepsizes and only in limited stepsize regimes (details in § 2). To date, these analyses have not yielded theoretical communication gains over the simple baselines of both (Accelerated) Minibatch SGD and FedAvg in any regime. Put another way, local update algorithms have not yet been theoretically shown to perform better than minibatch algorithms unless heterogeneity is very low.

Contributions. In this paper, we first show that multistage optimization, when applied to baseline federated minimization algorithms, achieves comparable or better worst-case convergence rates than known baselines (including minibatch SGD and FedAvg) in all heterogeneity regimes. In doing so, multistage algorithms resolve an open problem from (Woodworth et al., 2020a), which asked if one can design an optimal algorithm that combines the advantages of both local SGD and minibatch SGD and enjoys guarantees that dominate both. We propose multistage algorithms that match the lower bound of (Woodworth et al., 2020a) on convergence rates for first-order federated algorithms up to logarithmic or condition numbers factors, explaining why multistaging has been so successful in practice.

We show these theoretical gains by first analyzing an algorithm that (nearly) achieves the lower bound: a simple two-stage combination of Accelerated Minibatch SGD and FedAvg. Loosely, we first run FedAvg up to an error floor that depends on client heterogeneity; then, we use Accelerated Minibatch SGD to complete the optimization. Further, we show that similar gains can be obtained by a more general multistage algorithm which gradually moves from the FedAvg update to Minibatch SGD udpate. This multistage has the added benefit of not requiring the knowledge of the noise bound σ^2, ζ^2, D, or initial suboptimality Δ. This intuition is not specific to the combination of Accelerated Minibatch SGD and FedAvg. We demonstrate the generality of multistaging by applying it to other pairs of FL optimization algorithms to obtain the same nearly-optimal worst-case convergence rate (but possibly much better average-case convergence rate as shown in § 4).

Finally, we demonstrate the practical value of these theoretical insights through experiments involving logistic regression on the MNIST dataset (LeCun et al., 2010).

2. Model and Related Work

As stated in § 1, N is the number of clients, heterogeneity ζ (resp. heterogeneity at optimum ζ_*) is defined in (resp. below) Eq. (2). We assume that each F_i is β-smooth and μ-strongly convex; we let $\kappa = \beta/\mu$ denote the condition number. We assume bounded variance of stochastic gradients on each machine:

$$\mathbb{E}_{z_i} \|\nabla f(x; z_i) - \nabla F_i(x)\|^2 \leq \sigma^2. \tag{3}$$

We bound the initial suboptimality gap as $\Delta \geq \mathbb{E}F(x_0) - F(x^*)$, where x_0 is the initial point, and x^* is the optimal

[1]This approach and its variants has also been referred to as Local Stochastic Gradient Descent (Stich, 2018; Khaled et al., 2020; Woodworth et al., 2020b) and LocalUpdate (Charles & Konečnỳ, 2020) in literature, and it has its origin in Parallelized Stochastic Gradient Descent (Zinkevich et al., 2010).

point. Optimization proceeds in rounds; K is the number of client iterations per round, and R is the number of rounds. In some theorems, clients take minibatches of size B in their local steps; unless specified otherwise, B is 1. Our goal is to minimize the optimization error after R rounds (or *round complexity*, the number of rounds R to a given error ϵ). We use the notation $\tilde{\mathcal{O}}$ to hide polylogarithmic factors.

2.1. Related Work

To date, the best worst-case convergence rates are achieved (in different heterogeneity regimes) by FedAvg (McMahan et al., 2017) and Minibatch SGD.

FedAvg. Convergence properties of Federated Averaging for convex minimization was first studied in the homogeneous client setting (Stich, 2018; Wang & Joshi, 2018; Woodworth et al., 2020b). These rates were later extended to the heterogeneous client setting (Khaled et al., 2020; Karimireddy et al., 2020b; Woodworth et al., 2020a), including an accompanying lower bound (Woodworth et al., 2020a).

The fastest known rate for the worst-case optimization error of Local SGD is due to (Gorbunov et al., 2020) (Table 1):

$$\tilde{\mathcal{O}}(\kappa\Delta\exp(-\frac{KR}{\kappa})+\frac{\sigma^2}{\mu NKR}+\frac{\kappa\sigma^2}{\mu KR^2}+\frac{\kappa\zeta^2}{\mu R^2}). \quad (4)$$

If we take enough local steps K, this error is dominated by the last term, $\frac{\beta\zeta^2}{\mu^2R^2}$. If heterogeneity is low (e.g., $\zeta^2=0$), then for K large enough, only one round of communication is needed. This suggests why FedAvg outperforms Accelerated Minibatch SGD in the low-heterogeneity regime.

Accelerated Minibatch SGD. In the other extreme of highly heterogeneous clients Accelerated Minibatch SGD is optimal. It has a convergence rate (Ghadimi & Lan, 2012; Woodworth et al., 2020a) of

$$\mathcal{O}(\Delta\exp(-\frac{R}{\sqrt{\kappa}})+\frac{\sigma^2}{\mu NKR}) \quad (5)$$

Therefore, FedAvg's communication R complexity cannot be less than that of the basic minibatch algorithms unless $\zeta_*^2\leq\mu\epsilon$, i.e., heterogeneity must scale with ϵ.

Lower Bound. The lower bound for first-order distributed algorithms in the heterogeneous setting, from (Woodworth et al., 2020a), is (also shown in Table 1):

$$\Omega(\min\{\frac{\Delta}{\sqrt{\kappa}},\frac{\mu\zeta_*^2}{\beta^2}\}\exp(-\frac{R}{\sqrt{\kappa}})+\frac{\sigma^2}{\mu NKR}) \quad (6)$$

Note the exponential difference between the lower bound in Eq. (6) and the tightest-known ζ-dependent upper bound (i.e., LSGD, Eq. (4)) with respect to their dependency on heterogeneity and rounds R. In the upper bound for LSGD (Eq. (4)), ζ^2 is scaled by $1/R^2$, whereas in the lower bound (Eq. (6)), it is scaled by a term decaying exponentially fast in R. A natural question is whether this gap can be closed.

Closing the gap. Today, no algorithm is known to close this gap, or even achieve comparable or faster rates than *both* Accelerated Minibatch SGD and Local SGD across all heterogeneity regimes. This may be partially because prior theoretical analysis for FL algorithms focused mainly on three hyperparameter regimes: (i) local update stepsizes scale as $1/K$ (e.g., S-Local-SVRG or FedAdam) (ii) local update stepsizes are small or zero (e.g., SCAFFOLD) (iii) local update stepsizes do not depend on K (e.g., Local SGD). Intuitively, regimes (i) and (ii) do not take sufficient advantage of local steps to excel in the moderate- to low-heterogeneity regime, whereas regime (iii) may converge too quickly to local optima, leading to a severe dependency on heterogeneity. The only work outside these regimes (to our knowledge) applies only to quadratics (Charles & Konečnỳ, 2020) and does not achieve the FedAvg rate, possibly due to insufficiently aggressive stepsize decay.

In this paper, we show that this theoretical gap can be closed by carefully combining existing methods (in stepsize regimes where analysis is known) in *stages*. In the next section, we explain how to achieve these gains through a simple multistage federated minimization framework.

Multistage Algorithms. Multistage algorithms have been employed in the classical convex optimization setting to obtain optimal rates (Aybat et al., 2019; Fallah et al., 2020; Ghadimi & Lan, 2012; Woodworth et al., 2020a). The general idea is to exponentially decrease the learning rate of some algorithm (for example, Accelerated SGD) in stages to optimally balance the optimizer's error due to bias and variance. Our federated setting differs from the prior multistage optimization in the following ways: (1) There is the optimization error due to client drift (Reddi et al., 2020; Karimireddy et al., 2020b)in addition to the error due to bias and variance; this additional type of error cannot be accounted for by existing multistage algorithms (2) Our objective is to minimize communication rounds rather than iteration complexity (3) We "multistage" or schedule more hyperparameters than just a learning rate.

3. Multistage Federated Minimization

In this section, we introduce a framework for multistage federated minimization algorithms in the strongly convex setting. The framework (specified in Algorithm 1) proceeds over S stages; between stages, we can change both the optimization algorithm and the associated hyperparameters (e.g., stepsizes). Each stage $s\leq S$ requires us to choose a local (client) optimization algorithm (e.g., FedAvg), a set of inner (client) and outer (server) stepsizes, a duration in number of rounds R, a number of local client steps K,

Table 1. Rates under the strongly convex conditions. Here we hide constants and log terms. ASG denotes the constant stepsize variant of Accelerated Minibatch SGD that we specify in Theorem 4. M-LSGD is the multistage variant of Local SGD we introduce in Theorem 5. M-ASG is a previously known multistage accelerated stochastic gradient method (Aybat et al., 2019). When applicable, K and B are taken large enough to match the dominating terms in R, and B is folded into σ^2 everywhere to simplify notation. "X $\to$ Y" denotes first running "X" and using the output as input to "Y". All upper bounds require knowledge of at least two of $\sigma^2, \Delta, \zeta^2$, except for Minibatch SGD and the LSGD $\to$ M-ASG rate. All rates achieved by LSGD$\to$X for some algorithm X are also achived by SCAFFOLD$\to$X. The two SCAFFOLD rates are achieved by different stepsize choices. min in the LSGD$\to$SGD and LSGD$\to$ASG rates comes from using a meta-algorithm that runs SGD and ASG repsectively if the first part of the min is smaller. min in the M-LSGD$\to$M-ASG rate comes from running M-ASG and M-LSGD$\to$M-ASG in parallel and then evaluating objective values of both solutions and choosing the best one (requires one extra round of communication). M-LSGD$\to$M-ASG is also $R > \Omega(\sqrt{\kappa})$.

Method/Analysis	$\mathbb{E}F(\hat{x}) - F(z^*) \le \tilde{\mathcal{O}}(\cdot)$
Minibatch Methods	
Minibatch SGD (SGD) (Woodworth et al., 2020a)	$\frac{\beta\Delta}{\mu}\exp(-\frac{R}{\kappa}) + \frac{\sigma^2}{\mu NKR}$
Minibatch AC-SA (AC-SA) (Woodworth et al., 2020a)	$\Delta\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu NKR} \quad [R \ge \Omega(\sqrt{\kappa})]$
Federated Algorithms	
SCAFFOLD (Karimireddy et al., 2020b)	$(\beta\Delta + \frac{\mu\zeta_*^2}{\beta^2})\exp(-\frac{R}{\kappa}) + \frac{\sigma^2}{\mu NKR}$
Local SGD (LSGD) (Gorbunov et al., 2020)	$\kappa\Delta\exp(-\frac{KR}{\kappa}) + \frac{\sigma^2}{\mu NKR} + \frac{\beta\sigma^2}{\mu^2KR^2} + \frac{\beta\zeta^2}{\mu^2R^2}$
SCAFFOLD (Theorem 7)	$\kappa\Delta\exp(-\frac{KR}{\kappa}) + \frac{\sigma^2}{\mu NKR} + \frac{\beta\sigma^2}{\mu^2KR^2} + \frac{\beta\zeta^2}{\mu^2R^2}$
LSGD $\to$ SGD (Theorem 1)	$\min\{\kappa\Delta, \frac{\beta^2\zeta^2}{\mu^3R^2}\}\exp(-\frac{R}{\kappa}) + \frac{\sigma^2}{\mu NKR}$
LSGD $\to$ ASG (Theorem 2)	$\min\{\Delta, \frac{\beta\zeta^2}{\mu^2R^2}\}\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu NKR} + \frac{\beta\sigma^2}{\mu^2NKR^3}$
M-LSGD $\to$ M-ASG (Theorem 3)	$\min\{\Delta, \frac{\beta^2\zeta^2}{\mu^3R^2} + \kappa\Delta\exp(-R)\}\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu NKR}$
Lower bound (Woodworth et al., 2020a)	$\min\{\frac{\Delta}{\sqrt{\kappa}}, \frac{\mu\zeta_*^2}{\beta^2}\}\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu NKR}$

and other hyperparameters (e.g., momentum, weights for averaging client updates).

Step 1: Local updates: For a given stage $s \le S$, we iterate over rounds $r \in \{1, \dots, R\}$. In a round r, each client first runs K local steps of its local optimizer (e.g., FedAvg):

$$x_{i,k}^{r,s} \leftarrow x_{i,k-1}^{r,s} - \eta_l^{(s)}\tilde{\nabla}F_i(x_{i,k-1}^{r,s})$$

where $\eta_l^{(s)}$ denotes the local stepsize and $\tilde{\nabla}F_i(x_{i,k-1}^{r,s})$ is the possibly random (pseudo-)gradient evaluated on the previous local iterate. The nature of this pseudo-gradient depends on the local optimizer.

Step 2: Global round update: Clients then perform a server update, through which the server averages the output of the N clients' local optimizations through a function $\mathsf{ClientOPT}(\eta_l^{(r,s)}, x^{r,s})$, which is stored in a variable $G^{r,s}$. Again, the nature of $\mathsf{ClientOPT}$ depends on the local optimizer being used. For example, for FedAvg, we have

$$\mathsf{ClientOPT}(\eta_l^{(r,s)}, x^{r,s}) = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\tilde{\nabla}F_i(x_{i,k-1}^{r,s}).$$

Other ways to instantiate $\mathsf{ClientOPT}$ include collecting the gradients from client variance-reduced methods like SS-Local-SGD (Gorbunov et al., 2020) (which is a stateless variant of SCAFFOLD), or a simple minibatch update of size NK where $\mathsf{ClientOPT}(\eta_l^{(r,s)}, x^{r,s}) = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\tilde{\nabla}F_i(x^{r,s})$ from the clients, which is a simple minibatch gradient taken at $x^{r,s}$ of size NK, scaled up by K.

Finally, the server takes a step of (accelerated) stochastic gradient descent based on its global iterate history, and the computed $G^{r,s}$ with global stepsize $\eta_g^{(r,s)}$.

Step 3: Stage update: Once R rounds are complete, the server completes the stage by taking a weighted average of its global iterates with weights $\{\{\{w_{i,k}^{r,s}\}_{i\le N}\}_{k\le K}\}_{r\le R}$:

$$\mathsf{StageWgtAverage}(\{\{w_k^{r,s}\}_{k\le K}\}_{r\le R}) = \frac{1}{W^{(s)}}\sum_{r=1}^{R}\sum_{k=1}^{K}w_k^{r,s}\Big(\frac{1}{N}\sum_{i=1}^{N}x_{i,k}^{r,s}\Big),$$

where $W^{(s)} = \sum_{r=1}^{R}\sum_{k=1}^{K}w_k^{r,s}$. This is commonly done to facilitate theoretical analysis (Karimireddy et al., 2020b), (Gorbunov et al., 2020), though it is often unnecessary in practice (i.e., we can just use the last iterate). The weights are determined by strong convexity μ and stepsizes, and do not require further tuning.

Algorithm 1 Multistage federated minimization framework

Input: inner stepsizes $\{\{\eta_l^{(r,s)}\}_{r\le R}\}_{s\le S}$, outer stepsizes $\{\{\eta_g^{(r,s)}\}_{r\le R}\}_{s\le S}$
momentums $\{\gamma^{(s)}\}_{s\le S}$, round lengths $\{R^{(s)}\}_{s\le S}$, initial point $\hat{x}^0$
weights for calculating weighted average solution from iterates $\{\{\{w_k^{r,s}\}_{k\le K}\}\}_{r\le R}\}_{s\le S}$
for $s = 1, \dots, S$ **do**
  $x^{0,s}, x^{1,s} \leftarrow \hat{x}^{s-1}$
  for $r = 1, \dots, R^{(s)}$ **do**
    ▷ **Obtain pseudo-gradient from clients**
    $G^{r,s} \leftarrow \mathsf{ClientOPT}(\eta_l^{(r,s)}, x^{r,s})$
    ▷ **Perform server update**
    $x^{r+1/2,s} \leftarrow (1+\gamma^{(s)})x^{r,s} - \gamma^{(s)}x^{r-1,s}$
    $x^{r+1,s} \leftarrow x^{r+1/2,s} - \eta_g^{(r,s)}G^{r,s}$
  end for
  $\hat{x}^s \leftarrow \mathsf{StageWgtAverage}(\{\{w_k^{r,s}\}_{k\le K}\}_{r\le R})$
end for
return $\hat{x}^{R_S,S}$

3.1. Two stage algorithms

We begin with a two-stage instantiation of Algorithm 1. The main idea is to first run Local SGD for a constant fraction of the round budget R, which reduces error to the "heterogeneity floor". Then we use a minibatch algorithm to complete convergence. This technique parallels multistage algorithms for obtaining optimal dependence on noise in convex optimization (Aybat et al., 2019). There, the idea is to start with a large stepsize to reduce the bias term (i.e. the exponential decay term) in the convergence rate, then decay the stepsize to reduce the noise term until convergence.

In this subsection, we find that two stages suffice to achieve low round complexities if $\Delta, \zeta^2, \sigma^2$ are known. In the next subsection, we show how to achieve low round complexities when $\Delta, \zeta^2, \sigma^2$ are unknown, by using more than two stages. We first show how to combine FedAvg with Minibatch SGD to obtain exponential gains in round complexity (as a function of heterogeneity ζ) compared to either baseline.

Theorem 1. *Run Algorithm 1 over $S = 2$ stages. In the first stage, use Local SGD as for $R/2$ rounds under the following parameter settings: $\eta = \eta_g^{(r,1)} = \eta_l^{(r,1)} = \mathcal{O}(\min\{\frac{1}{\beta}, \frac{1}{\mu KR}\})$, $S = 1$, $\gamma^{(1)} = 0$, and $w_k^{r,1} = (1-\mu\eta)^{rK+k+1}$. In the second stage, run Minibatch SGD (so $\gamma^{(2)} = 0$) for $R/2$ rounds under the following settings:*

1. *If $R > \frac{4\beta}{\mu}$ and $r < \lceil \frac{R}{2} \rceil$:*
   $\eta_l^{(r,2)} = 0$, $\eta_g^{(r,2)} = \eta_g = \frac{1}{4\beta}$, $w_k^{r,2} = 0$
2. *If $R > \frac{4\beta}{\mu}$ and $r \ge \lceil R/2 \rceil$:*
   $\eta_g^{(r,2)} = \eta_g = \frac{2}{\mu(\kappa - R - \lceil \frac{R}{2} \rceil)}$,
   $w_k^{r,2} = (1-\mu\eta_g)^{-(rK+k+1)} = (\kappa + r - \lceil \frac{R}{2} \rceil)^2$
3. *If $R \le \frac{4\beta}{\mu}$:*
   $\eta_g^{(r,2)} = \frac{1}{4\beta}$, $w_k^{r,2} = (1-\mu\eta_g)^{-(rK+k+1)}$.

Denote the output of this second stage by $\hat{x}$. For large enough number of local rounds K, the convergence rate of this two-stage algorithm is

$$\mathbb{E}F(\hat{x}) - F(x^*) \le \tilde{O}((\frac{\beta^2\zeta^2}{\mu^3R^2})\exp(-\frac{R}{\kappa}) + \frac{\sigma^2}{\mu NKR}) \tag{7}$$

Proof: See Section A

This algorithm already exponentially improves the dependence of the round complexity on heterogeneity from the previous state of the art of $\frac{\beta\zeta^2}{\mu^2R^2}$ to $(\frac{\beta^2\zeta^2}{\mu^3R^2})\exp(-\frac{R}{\kappa})$.

Next, we obtain our main result, which tightens the dependency on κ to match the lower bound in Eq. (6) up to logarithmic factors by replacing Minibatch SGD with Accelerated Minibatch SGD.

Theorem 2. *Run Algorithm 1 over $S = 2$ stages. In the first stage, use Local SGD for for $R/2$ rounds under the parameter settings of Stage 1 from Theorem 1.*

In the second stage, run Accelerated Minibatch SGD for $R/2$ rounds with the following parameter settings:
$\eta_l^{(r,2)} = 0, \qquad \eta_g^{(r,2)} = \eta_g = \mathcal{O}(\min\{\frac{1}{K\beta}, \frac{1}{\mu KR^2}\})$
$\gamma^{(2)} = \frac{1-\sqrt{\mu\eta_g K}}{1+\sqrt{\mu\eta_g K}}$
$w_k^{r,2} = 0$ *for all k, r except for* $w_1^{R/2,2} = 1$ *(i.e. return the last iterate).*

Denote the output of this second stage by $\hat{x}$. For large enough number of local rounds K, the expected convergence rate of this two-stage algorithm is

$$\tilde{\mathcal{O}}(\frac{\beta\zeta^2}{\mu^2R^2}\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu NKR} + \frac{\beta\sigma^2}{\mu^2NKR^3}) \tag{8}$$

Proof: See Section B

Observe that this rate is strictly better than the Local SGD rate (4) and is also strictly better than the Minibatch Accelerated SGD rate (5) when $\frac{\beta\zeta^2}{\mu^2R^2} \le \Delta$ (since the Minibatch Accelerated SGD rate requires that $R \ge \sqrt{\kappa}$). In practice, the initialization can be far from the optimum, leading to large Δ. The same rates can be achieved when using one round of SS-Local-SGD as the local optimization algorithm for both of the above two-stage algorithms.

3.2. Multistage algorithms

The previous algorithms require knowledge of several data- and problem-dependent parameters, such as β, μ, σ^2, ζ^2,

and Δ. Although the FL literature commonly assumes knowledge of such parameters (Woodworth et al., 2020a; Karimireddy et al., 2020b; Gorbunov et al., 2020), this may be unrealistic in practice. In this section, we explore how to use multistage algorithms to remove dependencies on all such parameters except β, which can be estimated from line search techniques (Beck & Teboulle, 2009; Schmidt et al., 2015), and μ, which can be estimated from regularization (which is often used in machine learning problems). The key idea is to divide our total stages S into two *superstages*: the first superstage uses one optimization algorithm (e.g., FedAvg) with one hyperparameter schedule, and the second super-stage uses another (e.g., Minibatch SGD).

Theorem 3. *Run Algorithm 1 over two superstages for a total of R rounds, with $R \geq 4\sqrt{\kappa}$. In the first superstage, run FedAvg in the setting of Theorem 5 for $R/2$ rounds.*

In the second superstage, run Multistage Accelerated Stochastic Gradient (M-ASG) (Aybat et al., 2019) with a minibatch of size BNK for some constant $B > 0$, using the following parameters (for legibility, we reset $s = 1$ for the second superstage):
$\eta_l^{(r,s)} = 0, \qquad \eta_g^{(r,1)} = \frac{1}{K\beta}$ *with* $R^{(1)} \geq 1$
$\eta_g^{(r,s)} = \frac{1}{K2^{2k}\beta}$ *with* $R^{(s)} = 2^s\lceil\sqrt{\kappa}\log(2^{p+2})\rceil$ *for any* $p \geq 1$ *and* $s \geq 2$
$\gamma^{(s)} = \frac{1-\sqrt{\mu\eta_g^{(r,s)}K}}{1+\sqrt{\mu\eta_g^{(r,s)}K}}$
$K = \lceil\kappa\rceil$ $R^{(1)} = \lceil(p+1)\sqrt{\kappa}\log(12(p+1)\kappa)\rceil$

For B large enough, this achieves expected convergence rate

$$\mathcal{O}((\kappa\Delta\exp(-R) + \frac{\beta^2\zeta^2}{\mu^3R^2})\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu BNKR}).$$

Proof: See Section D

When $\Delta \geq \frac{\beta\zeta^2}{\mu^2}$ this multistage algorithm has a faster rate than Accelerated Minibatch SGD (note that now the variance is $\frac{\sigma^2}{B}$). This is done without knowledge of ζ^2, σ^2, or Δ. This algorithm also has better round complexity than both Minibatch Accelerated Stochastic Approximation (AC-SA) (Woodworth et al., 2020a) and Local SGD when $\mu\epsilon < \zeta^2 < \frac{\mu^2\Delta}{\beta}$, thus resolving the open question from (Woodworth et al., 2020a).

4. Experiments

We empirically evaluate multistaging on federated regularized logistic regression. Let $(x_{i,j}, y_{i,j})$ be the jth datapoint of the ith client. We minimize Eq. (1) where

$$F_i(w) = \frac{1}{n_i}(\sum_{j=1}^{n_i} -y_{i,j}\log(w^\top x_{i,j}) - (1-y_{i,j})\log(1-w^\top x_{i,j}))) + \frac{\mu}{2}\|w\|^2.$$

Baselines. We compare the communication round complexities of four single-stage baselines: FedAvg, Minibatch SGD (SGD), Accelerated Minibatch SGD (ASG), and SCAFFOLD (precisely, SS-Local-SGD, which is a stateless variant of SCAFFOLD). We compare these baselines to our multistage combinations.

Dataset. We use the MNIST dataset of handwritten digits from 0-9 (LeCun et al., 2010). We model a federated setting with five clients by partitioning the data into groups. First, we take 500 images from each class (digit) for all experiments, totaling 5,000 images. Each client's local data is a mixture of data drawn exclusively from two digit classes (leading to heterogeneity), and data sampled uniformly from all classes. We call a federated dataset $X\%$ homogeneous if first $X\%$ of each class's 500 images is shuffled and evenly partitioned to each client. The remaining $(100-X)\%$ of the dataset is partitioned as follows: client $i \in \{1,\ldots,5\}$ receives the remaining non-shuffled data from classes $2i-2$ and $2i-1$. For example, in a 50% homogeneous setup, client 3 has 250 samples from digit 4, 250 samples from digit 5, and 500 samples drawn uniformly from all classes. Note that 100% homogeneity is *not* the same thing as setting heterogeneity $\zeta = 0$ due to sampling randomness; we use this technique for lack of a better control over ζ. In all experiments we are concerned with variants of binary classification, so we let all the even classes represent 0's and the odd classes represent 1's. We set $K = 20$.

Hyperparameters. All experiments are initialized at 0 with regularization $\mu = 0.1$. We fix the total number of rounds R (differs across experiments). For algorithms using Nesterov momentum, we calculate the momentum parameter from strong convexity μ and the chosen stepsizes. For the two-stage methods, we tune the stage 1 stepsize $\eta = \eta_l^{(r,1)} = \eta_g^{(r,1)}$ in the range below:

$$\{10^{-3}, 10^{-2.5}, 10^{-2}, 10^{-1.5}, 10^{-1}\} \tag{9}$$

For the second stage we let the stepsizes be $\eta_l^{(r,2)} = 0$, $\eta_g^{(r,2)} = \eta/K$. We also tune the percentage of rounds before switching to the second stage in the range as

$$\{10^{-2}, 10^{-1.625}, 10^{-1.25}, 10^{-0.875}, 10^{-0.5}\} \tag{10}$$

For multi-stage methods ($S > 2$), the number of total stages is problem-dependent. In superstage 1, we first tune the

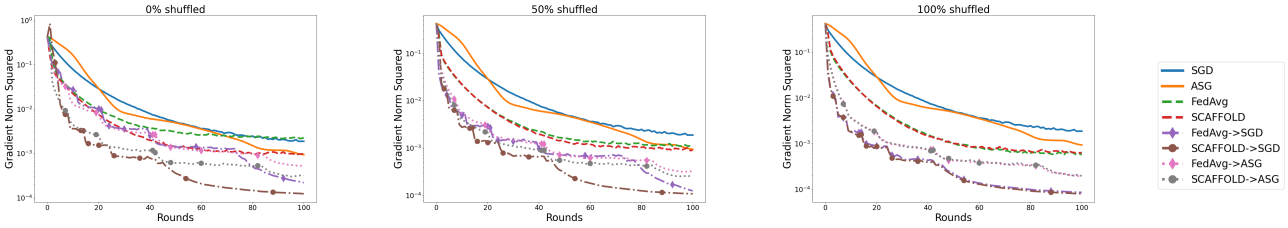

Figure 1. Stochastic gradients are computed with 1% of the data. Each datapoint is the average of 1000 runs. Plot titles denote data homogeneity (§ 4). "X→Y" denotes a multistage algorithm with X as the first superstage and Y as the second superstage. Across all heterogeneity levels, the multistage algorithms perform the best. Markers denote when a stage transition occurs.

Stage 1 stepsize η (as well as the fraction of total rounds allocated to Stage 1) in the range (9). Let $R^{(1)}$ be the number of rounds in the first stage. For subsequent stages $s \geq 2$, we use step sizes $\eta_l^{(r,s)} = \eta_g^{(r,s)} = \frac{\eta}{2^{s-1}}$ and $R^{(s)} = 2^s R^{(1)}$. Once the stepsize decreases to η/K, we enter the second superstage. Since we evaluate only minibatch algorithms in superstage 2, we use the same parameters as superstage 1, except the starting stepsize is $\eta_g^{(r,1)} = \frac{\eta}{K}$ while $\eta_l^{(r,s)} = 0$.

Results. Figure 1 compares the convergence of multi-stage algorithms to their single-stage counterparts in the stochastic gradient setting (minibatches are 1% of a client's data), over $R = 100$ rounds. The curves marked "X → Y" indicate a multi-stage algorithm that starts with Algorithm X in (Super-)Stage 1 and transitions to Algorithm Y in (Super-)Stage 2. Diamonds indicate stage transitions. Each such curve represents the better curve between the two-stage and the multi-stage ($S > 2$) version. We vary the homogeneity of the dataset from 0% (left) to 100% (right). We observe that in all homogeneity settings, the multi-stage algorithm demonstrates (constant) improvements over either baseline alone. These improvements grow more pronounced as data becomes more homogeneous. We also find that multistage algorithms outperform two-stage algorithms (and both outperform single-stage algorithms).

5. Conclusion

In this paper, we show that multistage parameter schedules improve FL communication complexity in theory and in practice. Future work includes expanding the ideas in this paper to the convex and non-convex setting.

References

Aybat, N. S., Fallah, A., Gurbuzbalaban, M., and Ozdaglar, A. A universally optimal multistage accelerated stochastic gradient method. *arXiv preprint arXiv:1901.08022*, 2019.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

Charles, Z. and Konečnỳ, J. On the outsized importance of learning rates in local update methods. *arXiv preprint arXiv:2007.00878*, 2020.

Fallah, A., Ozdaglar, A., and Pattathil, S. An optimal multistage stochastic gradient method for minimax problems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 3573–3579. IEEE, 2020.

Ghadimi, S. and Lan, G. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework. *SIAM Journal on Optimization*, 22(4):1469–1492, 2012.

Gorbunov, E., Hanzely, F., and Richtárik, P. Local sgd: Unified theory and new efficient methods. *arXiv preprint arXiv:2011.02828*, 2020.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Karimireddy, S. P., Jaggi, M., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020a.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020b.

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local sgd on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pp. 4519–4529. PMLR, 2020.

Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., and Stich, S. A unified theory of decentralized sgd with changing topology and local updates. In *International Conference on Machine Learning*, pp. 5381–5393. PMLR, 2020.

LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Schmidt, M., Babanezhad, R., Ahmed, M., Defazio, A., Clifton, A., and Sarkar, A. Non-uniform stochastic average gradient method for training conditional random fields. In *artificial intelligence and statistics*, pp. 819–828. PMLR, 2015.

Stich, S. U. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

Wang, J. and Joshi, G. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

Woodworth, B., Patel, K. K., and Srebro, N. Minibatch vs local sgd for heterogeneous distributed learning. *arXiv preprint arXiv:2006.04735*, 2020a.

Woodworth, B., Patel, K. K., Stich, S., Dai, Z., Bullins, B., Mcmahan, B., Shamir, O., and Srebro, N. Is local sgd better than minibatch sgd? In *International Conference on Machine Learning*, pp. 10334–10343. PMLR, 2020b.

Zinkevich, M., Weimer, M., Smola, A. J., and Li, L. Parallelized stochastic gradient descent. In *NIPS*, volume 4, pp. 4. Citeseer, 2010.

A. Proof of Theorem 1

First, let the total budget for the number of rounds be $2R$. We run the two stages of the algorithm with R rounds.

In the first stage, we run Algorithm 1 with

$$\eta = \eta_g^{(r,s)} = \eta_l^{(r,s)} = \min\{\frac{1}{4\beta}, \frac{\log(\max\{2, \min\{\frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 n\mu^2 R^2 K^2}{\sigma^2}, \frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 \mu^2 R^3 K^3}{6\beta K(\sigma^2 + 2K\zeta^2))}\}\}}{\mu KR}$$

$S = 1$, $\beta^{(s)} = 0$, $w_k^{r,s} = w_k^r = (1 - \mu\eta)^{rK+k+1}$. Note this is just running Local SGD (Gorbunov et al., 2020) with the above stepsizes. Therefore, from the proof of Gorbunov et al. (2020)(Eq.82), we know that we get the following convergence rate for the output of the algorithm

$$\mathbb{E}\, F(\hat{x}^1) - F(x^*) \leq 8\beta \exp(-\frac{\kappa KR}{8})\mathbb{E}\|\hat{x}^0 - x^*\|^2 + \frac{2(\Phi + 1)\sigma^2}{\mu NKR} + \frac{6\beta(\Phi^2 + 1)\sigma^2}{\mu^2 KR^2} + \frac{12\beta(\Phi^2 + 1)\zeta^2}{\mu^2 R^2} \tag{11}$$

where $\Phi = \log(\max\{2, \min\{\frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 n\mu^2 R^2 K^2}{\sigma^2}, \frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 \mu^2 R^3 K^3}{6\beta K(\sigma^2 + 2K\zeta^2))}\}\}$.

In the second stage, we run Algorithm 1 on $\hat{x}^1$ with $\beta^{(s)} = 0$ and

1. If $R > \frac{4\beta}{\mu}$ and $r < \lceil \frac{R}{2} \rceil$: $\eta_l^{(r,2)} = 0$, $\eta_g^{(r,2)} = \eta_g = \frac{1}{4\beta}$, $w_k^{r,2} = 0$
2. If $R > \frac{4\beta}{\mu}$ and $r \geq \lceil R/2 \rceil$: $\eta_g^{(r,2)} = \eta_g = \frac{2}{\mu(\kappa - R - \lceil \frac{R}{2} \rceil)}$, $w_k^{r,2} = (1 - \mu\eta_g)^{-(rK+k+1)} = (\kappa + r - \lceil \frac{R}{2} \rceil)^2$
3. If $R \leq \frac{4\beta}{\mu}$: $\eta_g^{(r,2)} = \frac{1}{4\beta}$, $w_k^{r,2} = (1 - \mu\eta_g)^{-(rK+k+1)}$.

Note that this is running Minibatch SGD with the above stepsize schedule. From the proof from Woodworth et al. (2020a), the output of this algorithm is

$$\mathbb{E}\, F(\hat{x}^2) - F(x^*) \leq 128\kappa(\mathbb{E}\, F(\hat{x}^1) - F(x^*)) \exp(-\frac{R}{8\kappa}) + \frac{72\sigma^2}{\mu NKR} \tag{12}$$

Plugging in (11), we get the claim.

B. Proof of Theorem 2

First, let the total budget for the number of rounds be $2R$. We run the two stages of the algorithm with R rounds.

In the first stage, we run Algorithm 1 with

$$\eta = \eta_g^{(r,s)} = \eta_l^{(r,s)} = \min\{\frac{1}{4\beta}, \frac{\log(\max\{2, \min\{\frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 n\mu^2 R^2 K^2}{\sigma^2}, \frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 \mu^2 R^3 K^3}{6\beta K(\sigma^2 + 2K\zeta^2))}\}\}}{\mu KR}$$

$S = 1$, $\beta^{(s)} = 0$, $w_k^{r,s} = w_k^r = (1 - \mu\eta)^{rK+k+1}$. Note this is just running Local SGD (Gorbunov et al., 2020) with the above stepsizes. Therefore, from the proof of Gorbunov et al. (2020), we know that we get the following convergence rate for the output of the algorithm

$$\mathbb{E}\, F(\hat{x}^1) - F(x^*) \leq 8\beta \exp(-\frac{\kappa KR}{8})\mathbb{E}\|\hat{x}^0 - x^*\|^2 + \frac{2(\Phi + 1)\sigma^2}{\mu NKR} + \frac{6\beta(\Phi^2 + 1)\sigma^2}{\mu^2 KR^2} + \frac{12\beta(\Phi^2 + 1)\zeta^2}{\mu^2 R^2} \tag{13}$$

where $\Phi = \log(\max\{2, \min\{\frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 n\mu^2 R^2 K^2}{\sigma^2}, \frac{\mathbb{E}\|\hat{x}^0 - x^*\|^2 \mu^2 R^3 K^3}{6\beta K(\sigma^2 + 2K\zeta^2))}\}\}$.

Next we run Algorithm 1 on $\hat{x}^1$ with $\eta_l^{(r,s)} = 0$ and

$$\eta_g^{(r,s)} = \eta_g = \min\{\frac{1}{K\beta}, \frac{\log(\max\{2, \min\{\frac{2\Delta_1 N\mu KR}{\sigma^2}, \frac{2\Delta_1 \mu^2 NKR^3}{\beta\sigma^2}\})\}}{\mu KR^2}\} \tag{14}$$

$\beta^{(s)} = \frac{1-\sqrt{\mu\eta_g K}}{1+\sqrt{\mu\eta_g K}}$, $S = 1$, and $w_k^{r,s} = 0$ for all k, r, s except for $w_1^{R_1,1} = 1$ and $\Delta_1 \leq 8\beta\exp(-\frac{\kappa R}{8})\mathbb{E}\|\hat{x}^0 - x^*\|^2 + \frac{2(\Phi+1)\sigma^2}{\mu NKR} + \frac{6\beta(\Phi^2+1)\sigma^2}{\mu^2 KR^2} + \frac{12\beta(\Phi^2+1)\zeta^2}{\mu^2 R^2}$. From Theorem 4 we know that

$$\mathbb{E}\, F(\hat{x}^2) - F(x^*) \leq 2\Delta_1 \exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2(\Phi_2^{1/2}+1)}{\mu NKR} + \frac{\beta\sigma^2(\Phi_2^{3/2}+1)}{\mu^2 NKR^3} \tag{15}$$

where $\Phi_2 = \log(\max\{2, \min\{\frac{2\Delta_1\mu K}{\sigma^2}, \frac{2\Delta\mu^2K^3}{\beta\sigma^2}\})\}$.

Plugging in Δ_1 gets us the desired conclusion.

C. Theorem 4

Theorem 4. *Consider Nesterov's Accelerated Stochastic gradient method with the iteration sequence*

$$x_{k+1/2} = (1+\beta)x_k - \beta x_{k-1},\ x_{k+1} = x_{k+1/2} - \alpha\tilde{\nabla}F(x_{k+1/2}) \tag{16}$$

then

$$\mathbb{E}\, F(x_K) - F(x^*) \leq 2\Delta\exp(-\frac{K}{\sqrt{\kappa}}) + \frac{\sigma^2(\Phi^{1/2}+1)}{\mu K} + \frac{\beta\sigma^2(\Phi^{3/2}+1)}{\mu^2 K^3} \tag{17}$$

Proof. From Aybat et al. (2019) (Eq.13), we know that

$$\mathbb{E}\, V_{k+1} \leq (1-\sqrt{\alpha\mu})(\mathbb{E}\, V_k) + \frac{\sigma^2\alpha}{2}(1+\alpha\beta) \tag{18}$$

where $F(x_k) - F(x^*) \leq V_k \leq 2(F(x_k) - F(x^*))$.

If we unroll this, we get

$$\mathbb{E}\, V_K \leq (1-\sqrt{\alpha\mu})^K\, \mathbb{E}\, V_0 + \frac{\sigma^2\alpha^{1/2}}{2\mu^{1/2}} + \frac{\sigma^2\alpha^{3/2}\beta}{2\mu^{1/2}} \leq \exp(-\sqrt{\alpha\mu}K)\, \mathbb{E}\, V_0 + \frac{\sigma^2\alpha^{1/2}}{2\mu^{1/2}} + \frac{\sigma^2\alpha^{3/2}\beta}{2\mu^{1/2}} \tag{19}$$

Now if

$$\alpha = \min\{\frac{1}{\beta}, \frac{\log(\max\{2, \min\{\frac{2\Delta N\mu K}{\sigma^2}, \frac{2\Delta\mu^2 NK^3}{\beta\sigma^2}\})\}}{\mu K^2}\} \tag{20}$$

observe that if $\frac{1}{\beta} \leq \frac{\log(\max\{2,\min\{\frac{2\Delta N\mu K}{\sigma^2}, \frac{2\Delta\mu^2 NK^3}{\beta\sigma^2}\})\}}{\mu K^2}$ we know that letting $\Phi = \log(\max\{2, \min\{\frac{2\Delta N\mu K}{\sigma^2}, \frac{2\Delta\mu^2 NK^3}{\beta\sigma^2}\})\}$

$$\mathbb{E}\, V_K \leq \mathbb{E}\, V_0 \exp(-\frac{K}{\sqrt{\kappa}}) + \frac{\sigma^2\Phi^{1/2}}{2\mu K} + \frac{\beta\sigma^2\Phi^{3/2}}{2\mu^2 K^3} \tag{21}$$

and if $\frac{1}{\beta} > \frac{\log(\max\{2,\min\{\frac{2\Delta\mu K}{\sigma^2}, \frac{2\Delta\mu^2 K^3}{\beta\sigma^2}\})\}}{\mu K^2}$

$$\mathbb{E}\, V_K \leq \frac{\sigma^2}{\mu K} + \frac{\beta\sigma^2}{\mu^2 K^3} + \frac{\sigma^2\Phi^{1/2}}{2\mu K} + \frac{\beta\sigma^2\Phi^{3/2}}{2\mu^2 K^3} \tag{22}$$

where we used that $EV_0 \leq 2\Delta$.

So altogether, we have that

$$\mathbb{E}\, F(x_K) - F(x^*) \leq 2\Delta\exp(-\frac{K}{\sqrt{\kappa}}) + \frac{\sigma^2(\Phi^{1/2}+1)}{\mu K} + \frac{\beta\sigma^2(\Phi^{3/2}+1)}{\mu^2 K^3} \tag{23}$$

□

D. Proof of Theorem 3

Let x' be the output of multistage LSGD from the first half of the rounds. Then we know that the output of this has guarantee (from Thm 5)

$$\mathbb{E}F(x') - F(x^*) \leq \mathcal{O}(\kappa\Delta\exp(-\frac{KR}{\kappa}) + \frac{\beta\sigma^2}{\mu^2BNKR} + \frac{\beta^2\sigma^2}{\mu^3BKR^2} + \frac{\beta^2\zeta^2}{\mu^3R^2}) \tag{24}$$

Next, observe that the second half of the rounds are running M-ASG from (Aybat et al., 2019). That has a guarantee of (Corollary 3.8)

$$\mathbb{E}\, F(\hat{x}) - F(x^*) \leq \mathcal{O}(\Delta_1\exp(-\frac{R}{\sqrt{\kappa}}) + \frac{\sigma^2}{\mu NKR}) \tag{25}$$

where

$$\Delta_1 \leq \mathcal{O}(\kappa\Delta\exp(-\frac{KR}{\kappa}) + \frac{\beta\sigma^2}{\mu^2BNKR} + \frac{\beta^2\sigma^2}{\mu^3BKR^2} + \frac{\beta^2\zeta^2}{\mu^3R^2}) \tag{26}$$

Which gives the theorem statement if we let $\sigma = \frac{\sigma^2}{B}$ and $K = \lceil\kappa\rceil$.

E. Proof of Theorem 5

Theorem 5.

$$\eta_g^{(r,1)} = \eta_l^{(r,1)} = \frac{1}{4\beta}, R^{(1)}K \geq 1$$
$$\eta_g^{(r,1)} = \eta_l^{(r,1)} = \frac{1}{\beta 2^{s+2}}, R^{(s)}K = \lceil p2^{s+2}\kappa\log(2)\rceil$$

where $p \geq 3$ is some arbitrary positive number. Also set $\beta^{(s)} = 0$. Let ClientOPT *be one round of either SS-Local-SGD or Local SGD, except each local step is minibatched with a minibatch of size B, and have* ClientWeightedAverage *simply return the last iterate. Furthermore, require that $K \leq \lceil p\kappa\log(2)\rceil$, and $\lceil p\kappa\log(2)\rceil$ mod $K = 0$. Then if $R^{(1)}K = R/C$ for some constant and $KR \geq 2\kappa$,*

$$\mathbb{E}F(\hat{x}) - F(x^*) \leq \mathcal{O}(\kappa\Delta\exp(-\frac{KR}{\kappa}) + \frac{\beta\sigma^2}{\mu^2BNKR} + \frac{\beta^2\sigma^2}{\mu^3BKR^2} + \frac{\beta^2\zeta^2}{\mu^3R^2}) \tag{27}$$

Proof. We start by proving that

$$\text{err}_s \leq \frac{\exp(-(R_1K)/(4\kappa))}{2^{p(s-1)}}\text{err}_0 + \frac{\sigma^2}{2^s\mu\beta N} + \frac{6}{2^{2s}\mu\beta}(K\sigma^2 + 2K^2\zeta^2) \tag{28}$$

Where $\text{err}_s = \text{err}_s = \mathbb{E}\,\|x^{R_s,s} - x^*\|^2$. We prove this by induction on s. Recall that from Theorem 6 we have

$$\mathbb{E}\,\|x^{r,s} - x^*\|^2 \leq (1-\mu\eta)^{rK}\,\mathbb{E}\,\|x^{0,s} - x^*\|^2 + \frac{\eta\sigma^2}{\mu N} + 6\kappa\eta^2(K\sigma^2 + 2K^2\zeta^2) \tag{29}$$

Starting with the base case, with $\eta = \frac{1}{4\beta}$, $R_1K \geq 1$, we have

$$\text{err}_1 \leq \exp(-\frac{R_1K}{4\kappa})\text{err}_0 + \frac{\sigma^2}{4\beta\mu N} + \frac{3}{2\beta\mu}(K\sigma^2 + 2K^2\zeta^2) \tag{30}$$

Now let us assume the result holds for s and prove it for $s+1$. Using Theorem 6 for stage $s+1$ with $\eta = \frac{1}{\beta 2^{s+3}}$ and $R_{s+1}K = \lceil p2^{s+3}\kappa\log(2)\rceil$ gets us

$$\text{err}_{s+1} \leq \exp(-p\log(2))\text{err}_s + \frac{\sigma^2}{2^{s+3}\mu\beta N} + \frac{6}{2^{2(s+3)}\mu\beta}(K\sigma^2 + 2K^2\zeta^2) \tag{31}$$

Now using the induction hypothesis, we know

$$\text{err}_s \leq \frac{\exp(-(R^{(1)}K)/(4\kappa))}{2^{p(s-1)}}\text{err}_0 + \frac{\sigma^2}{2^s\mu\beta N} + \frac{6}{2^{2s}\mu\beta}(K\sigma^2 + 2K^2\zeta^2) \tag{32}$$

And substituting it back into (28) while having $p \geq 3$ gets us

$$\text{err}_{s+1} \leq \frac{\exp(-(R^{(1)}K)/(4\kappa))}{2^{ps}}\text{err}_0 + \frac{\sigma^2}{2^{s+1}\mu\beta N} + \frac{6}{2^{2s+2}\mu\beta}(K\sigma^2 + 2K^2\zeta^2) \tag{33}$$

□

Using the following lemma, we can translate this into a convergence rate:

Lemma 1. *Let $R^{(1)}K \geq 1$ and $R^{(s)}K = \lceil p2^{s+2}\kappa\log(2)\rceil$. Then we know that $2^k \geq \Theta(1)(\frac{RK - R^{(1)}K}{p\kappa})$.*

Proof. Comes from a simple unrolling of the stage lengths; similar to the proof in (Fallah et al., 2020). □

So if we plug in $R^{(1)}K = \frac{R}{C}$ for some constant $C \geq 2$ and $KR \geq 2\kappa$, we get the convergence rate in the theorem statement by applying smoothness.

F. Proof of Theorem 6

Theorem 6. *Suppose we ran Algorithm 1 for one stage with stepsizes $\eta_l^{(r,s)} = \eta_g^{(r,s)} = \eta$, with $B^{(s)} = 0$ and return the last iterate. Furthermore, require that $\eta \leq \frac{1}{\beta}$. Then*

$$\mathbb{E}\,\|x^{r,s} - x^*\|^2 \leq (1-\mu\eta)^{rK}\,\mathbb{E}\,\|x^{0,s} - x^*\|^2 + \frac{\eta\sigma^2}{\mu N} + 6\kappa(K\eta^2\sigma^2 + 2K^2\eta^2\zeta^2) \tag{34}$$

From Gorbunov et al. (2020) (Eq.82), we have the following for LSGD:

$$\mathbb{E}\,\|x^{t+1} - x^*\|^2 \leq (1-\mu\eta)\,\mathbb{E}\,\|x^t - x^*\|^2 + 2\beta\eta\,\mathbb{E}\,V_t + \eta^2\frac{\sigma^2}{N} \tag{35}$$

where $V_t = \frac{1}{N}\sum_{i=1}^N \mathbb{E}\,\|x_i^t - x^t\|^2$, where $t = Kr + k$ where r is current round, k is step on current round. x_i^t is client i's iterate at the t-th step. $x^t = \frac{1}{N}\sum_{i=1}^N x_i^t$, and η is both the inner stepsize and outer stepsize (i.e. $\eta_l^{(r,s)} = \eta$, $\eta_g^{(r,s)} = \eta$).

From Woodworth et al. (2020a) we know that for LSGD

$$V_t \leq 3K\eta^2\sigma^2 + 6K^2\eta^2\zeta^2 \tag{36}$$

So together,

$$\mathbb{E}\,\|x^{t+1} - x^*\|^2 \leq (1-\mu\eta)\,\mathbb{E}\,\|x^t - x^*\|^2 + 6\beta\eta(K\eta^2\sigma^2 + 2K^2\eta^2\zeta^2) + \frac{\sigma^2\eta^2}{N} \tag{37}$$

And unrolling this,

$$\mathbb{E}\,\|x^T - x^*\|^2 \leq (1-\mu\eta)^T\,\mathbb{E}\,\|x^0 - x^*\|^2 + \frac{\eta\sigma^2}{\mu N} + 6\kappa(K\eta^2\sigma^2 + 2K^2\eta^2\zeta^2) \tag{38}$$

G. Using SS-Local-SGD instead of Local SGD

In this section we show how one can use SS-Local-SGD instead of Local SGD as ClientOPT.

Theorem 7. *Let $V_t = \frac{1}{N}\sum_{i=1}^N \mathbb{E}\,\|x_i^t - x^t\|^2$, where $t = Kr + k$ where r is current round, k is step on current round, x_i^t is client i's iterate at the t-th step. Then SS-Local-SGD has*

$$V_t \leq 48K^2\eta^2\zeta^2 + 9\eta^2\sigma^2 \tag{39}$$

Proof.

$$V_t \leq \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbb{E}\left\|x_i^t - x_j^t\right\|^2 \tag{40}$$

By Jensen's. Let $g_i(x_i^t)$ be the SS-Local-SGD local update for client i at iteration t. Let $\mathbb{E}_t$ denote expectation conditioned on everything up to the $t-1$'th iteration. Let y^t be the last synchronized iterate before iteration t. In the following we use the fact that the control iterates are computed with minibatches of size K at the clients.

$$\begin{aligned}
&\mathbb{E}\left\|x_i^t - x_j^t\right\|^2 \\
&\leq \mathbb{E}\left\|x_i^{t-1} - x_j^{t-1} - \eta\,\mathop{\mathbb{E}}_{t-1} g_i(x_i^{t-1}) + \eta\,\mathop{\mathbb{E}}_{t-1} g_j(x_j^{t-1})\right\|^2 + \eta^2\sigma^2 \\
&= \mathbb{E}\left\|x_i^{t-1} - x_j^{t-1} - \eta(\tilde{\nabla}F(y^{t-1}) - \tilde{\nabla}F_i(y^{t-1}) - \nabla F_i(x_i^{t-1})) + \eta(\tilde{\nabla}F(y^{t-1}) - \tilde{\nabla}F_j(y^{t-1}) + \nabla F_j(x_j^{t-1}))\right\|^2 + \eta^2\sigma^2 \\
&= \mathbb{E}\left\|x_i^{t-1} - x_j^{t-1} \pm \eta\nabla F(x_i^{t-1}) \pm \eta\nabla F(x_j^{t-1}) \pm \eta\nabla F(y^{t-1})\right. \\
&\quad \left. - \eta(-\tilde{\nabla}F_i(y^{t-1}) - \nabla F_i(x_i^{t-1})) + \eta(-\tilde{\nabla}F_j(y^{t-1}) + \nabla F_j(x_j^{t-1}))\right\|^2 + \eta^2\sigma^2 \\
&\leq (1 + \frac{1}{K-1})\,\mathbb{E}\left\|x_i^{t-1} - x_j^{t-1} - \eta\nabla F(x_i^{t-1}) + \eta\nabla F(x_j^{t-1})\right\|^2 \\
&\quad + \eta^2 K\,\mathbb{E}\left\|\nabla F(x_i^{t-1}) - \nabla F_i(x_j^{t-1}) - \nabla F(x_j^{t-1}) + \nabla F_j(x_j^{t-1})\right. \\
&\quad \left. + \nabla F(y^{t-1}) - \nabla F_i(y^{t-1}) - \nabla F(y^{t-1}) + \nabla F_j(y^{t-1})\right\|^2 \\
&\quad + 3\eta^2\sigma^2 \\
&\leq (1 + \frac{1}{K-1})\,\mathbb{E}\left\|x_i^{t-1} - x_j^{t-1}\right\|^2 + 16K\eta^2\zeta^2 + 3\eta^2\sigma^2
\end{aligned}$$

So altogether,

$$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbb{E}\left\|x_i^t - x_j^t\right\|^2 \leq \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(1 + \frac{1}{K-1})\,\mathbb{E}\left\|x_i^{t-1} - x_j^{t-1}\right\|^2 + 16K\eta^2\zeta^2 + 3\eta^2\sigma^2 \tag{41}$$

Unrolling this until the last time the iterates were synchronized, we get

$$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbb{E}\left\|x_i^t - x_j^t\right\|^2 \leq 3K(16K\eta^2\zeta^2 + 3\eta^2\sigma^2) \tag{42}$$

Which implies

$$V_t \leq 48K^2\eta^2\zeta^2 + 9\eta^2\sigma^2 \tag{43}$$

□

This can be substituted into Theorem 6 to get the multistage rate for Theorem 3, or it can be substituted into Theorem G.3 in (Gorbunov et al., 2020) to get a convergence rate that differs from (11) by only constants in the terms. Thus we can use SS-Local-SGD instead of Local-SGD in any of the theorems presented in the main paper with only minor constant changes.

H. Additional Experimental Results

Figure 2 compares the convergence of multi-stage algorithms to their single-stage counterparts in the deterministic setting (full batch gradients), over $R = 200$ rounds. This is different from Fig. 1, which uses stochastic gradients. The curves marked "X $\rightarrow$ Y" indicate a multi-stage algorithm that starts with Algorithm X in (Super-)Stage 1 and transitions to Algorithm Y in (Super-)Stage 2. Each such curve represents the better (smaller) curve between the two-stage and the multi-stage ($S > 2$) version of the listed algorithms. We vary the homogeneity of the dataset from 0% (left) to 100% (right). We observe that in all homogeneity settings, the multi-stage algorithm demonstrates constant-order improvements

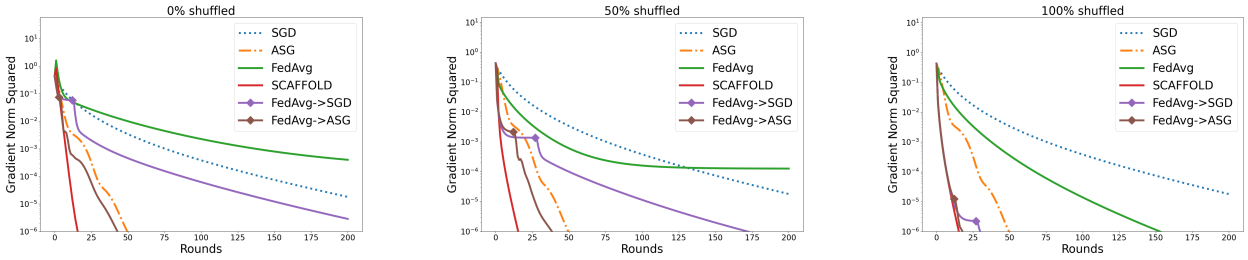

Figure 2. All gradients are deterministic. A plot's title denotes how homogeneous the data is across the clients, as described in Section 4. "X→Y" denotes a multistage algorithm with X as the first stage and Y as the second stage. Across all heterogeneity levels, SS-Local-SGD (with no scheduling) is the best, followed by LSGD→ASG. Diamonds denote when stage transitions occur.

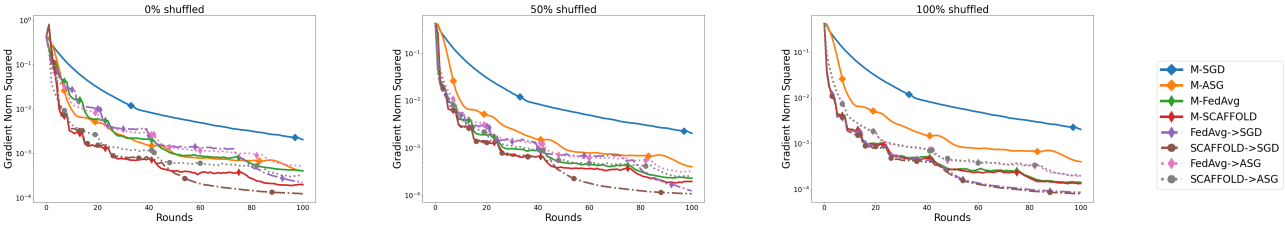

Figure 3. Stochastic gradients are computed with 1% of the data. Each datapoint is the average of 1000 runs. Plot titles denote data homogeneity (§ 4). "X→Y" denotes a multistage algorithm with X as the first superstage and Y as the second superstage. Across all heterogeneity levels, the multistage algorithms perform the best. Markers denote when a stage transition occurs.

in convergence rates over either baseline alone. These improvements are most visible in the early stages of training, and they grow more pronounced as data becomes more homogeneous. We also find that for the multi-stage curves in this plot, the two-stage version had better performance than the multi-stage version. This may be because without noise, we can aggressively set our Minibatch SGD step size without using multi-staging, so only two stages suffice.

Figure 3 compares the convergence of our two super-stage algorithms with multistage variants of SGD, ASG, FedAvg, and M-SCAFFOLD. For M-FedAvg and M-SCAFFOLD, we simply run the algorithms as we run the multi-stage methods as described in Section 4, except the first super-stage lasts the entire run. For M-ASG and M-SGD, we run the algorithms the same way we run the multi-stage methods as described in Section 4, except the second super-stage lasts the entire run and the initial stepsize is η. We find here that our algorithms with two superstages perform best.

Figure 4 gives hyperparameter ablation studies for the SCAFFOLD→SGD experiment at 100% shuffled dataset across clients. We are generally able to take larger initial stepsizes with multi-staging, which is what gives us the gain we observe.

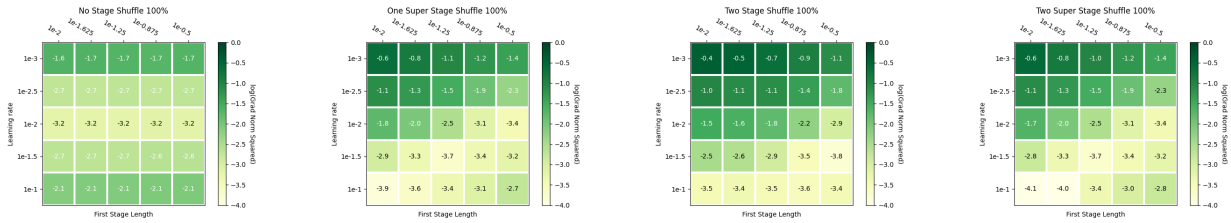

Figure 4. Stochastic gradients are computed with 1% of the data. Each datapoint is the average of 1000 runs. These are ablation studies of initial stage length and initial stepsize for SCAFFOLD→SGD, when the data is shuffled across all clients completely. No stage refers to simply running SCAFFOLD for a fixed step size. One super stage refers to running SCAFFOLD in a multistage manner (i.e. the first super stage lasts the whole run). Two stage refers to running SCAFFOLD with the initial learning rate specified and switching to SGD after the initial stage length with learning rate η/K. Two super stage is run as specified in Sec 4. Overall, multistaging appears to allow us to take larger initial step sizes, which is what is giving us our gain.