Bi-directional Adaptive Communication for Heterogenous Distributed Learning

Dmitrii Avdiukhin [1] **Nikita Ivkin** [2] **Sebastian U. Stich** [3] **Vladimir Braverman** [4]

Abstract

Communication constraints are a key bottleneck in distributed optimization, in particularly bandwidth and latency can be limiting factors when devices are connected over commodity networks, such as in Federated Learning. State-of-the-art techniques tackle these challenges by advanced compression techniques or delaying communication rounds according to predefined schedules. We present a new scheme that adaptively skips communication (broadcast and client uploads) by detecting slow-varying updates. The scheme automatically adjusts the communication frequency independently for each worker and the server. By utilizing an error-feedback mechanism – borrowed from the compression literature – we prove that the convergence rate is the same as for batch gradient descent in the convex and nonconvex smooth cases. We show reduction of the total number of communication rounds between server and clients needed to achieve a targeted accuracy, even in the case when the data distribution is highly non-IID.

1. Introduction

With the data moving to the edge devices, training large scale machine learning models unavoidably shifts towards the distributed settings (Kairouz et al., 2019). More and more applications require large number of workers cooperating in training one shared machine learning model, with each worker typically holding its small share of data and has very limited network bandwidth. Often such settings are driven by privacy concerns, so the data should not leave the device, among examples are personal smartphones, geo-destributed devices storing medical data, sensor networks, etc (Tomlinson et al., 2009; Brisimi et al., 2018).

In this paper, we address the problem of data parallel stochastic optimization with a central node coordinating computation of stochastic gradients on N edge nodes (or workers/clients):

$$\min_{x\in\mathbb{R}^d} f(x) \quad \text{with} \quad f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)\,.$$

Here $i \in [N]$ denotes the worker identifier, $f_i(x)\colon \mathbb{R}^d \to \mathbb{R}$ the loss function with respect to the model state $x \in \mathbb{R}^d$ on the worker $i \in [N]$. We assume that we can only query stochastic gradients for each $f_i(x)$, that is, we have access to a stochastic oracle $\mathbf{g}_i(\mathbf{x})$ with $\mathbb{E}[\mathbf{g}_i(\mathbf{x})] = \nabla f_i(\mathbf{x})$.

In the parameter server model stochastic optimization is performed via the following iterative steps: at time step t (1) each worker node computes a stochastic gradient $\mathbf{g}_i^{(t)}(\mathbf{x}^{(t)})$ on a local mini-batch of data, (2) each worker communicates the estimated gradient to the server, (3) the server performs a gradient update step, $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \frac{\alpha}{N}\sum_{i=1}^{N}\mathbf{g}_i^{(t)}$, where α denotes the learning rate, and finally, (4) the server broadcasts $\mathbf{x}^{(t+1)}$ to all workers.

With the number of workers growing and with additional network resources getting more and more expensive, the speed of communicating local stochastic gradients and broadcasting the updates became one of the main performance bottlenecks. Major attempts to ease that communication bottleneck fall into two categories: compressing the messages and reducing the communication frequency.

A rich variety of compressors, including diverse sparsification (Alistarh et al., 2018; Ivkin et al., 2019; Stich et al., 2018) and quantization (Alistarh et al., 2017; Bernstein et al., 2018) techniques, can drastically speedup the communication by dropping the message size from $O(d)$ down to $O(\log d)$ (Alistarh et al., 2017) or even $O(1)$ (Alistarh et al., 2018; Stich et al., 2018) per worker. Nevertheless, all the workers and the server are still required to communicate at every iteration, which can be infeasible in the Federated Learning setting when number of workers is in millions, and each worker have very limited network access (i.e. smart-

[*]Equal contribution [1]Indiana University, Bloomington [2]Amazon, New York [3]EPFL, Switzerland [4]Google, Mountain View. Correspondence to: Dmitrii Avdiukhin <davdyukh@iu.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.

phones, or sensor networks) (Kairouz et al., 2019). Reducing the communication frequency can be achieved by either fixed communication delay (a.k.a. LocalSGD (Zinkevich et al., 2010; Stich, 2019)) or via more adaptive communication protocols (Shokri Ghadikolaei et al., 2021; Chen et al., 2018). Both can be effectively incorporated into the error accumulation framework (Stich & Karimireddy, 2020) to guarantee the convergence. Our method falls into the class of adaptive communication protocols. Note that it is very well orthogonal to message compressing methods, thus can be efficiently combined with it.

While several compression methods (Tang et al., 2019; Philippenko & Dieuleveut, 2020) address the cost of broadcasting the updates (downlink), current approaches reducing the communication frequency, such as e.g. (Shokri Ghadikolaei et al., 2021), often neglect the cost of downlink and focus primarily on the cost of sending the gradients (uplink). In this paper we present a new communication protocol PROCRASTINATOR which detects slow-varying updates and gradients and choose to adaptively skip communication rounds – both server broadcasts and worker uploads, thus effectively reducing the latency for both.

1.1. Related Works

For training machine learning models in data centers, parameter-servers are used to aggregate the updates from participating devices and to orchestrate training (Dean et al., 2012; Keskar et al., 2017). In Federated Learning this server-based approach has been extend to train over decentralized data at larger scale (McMahan et al., 2016; Kairouz et al., 2019). To alleviate communication bottlenecks for exchanging updates between devices, several techniques have been proposed:

(i) Compressing updates: Computed gradients are compressed using vector sparsification – communicate only top k or random k coordinates of the gradient vector (Aji & Heafield, 2017; Stich et al., 2018), value quantization – efficiently encode every gradient value to use smaller number of bits (Alistarh et al., 2017; Bernstein et al., 2018) and random projections (Ivkin et al., 2019; Vogels et al., 2019; Rothchild et al., 2020b). The downside of these approaches is an inevitable loss of information during compression, which can potentially increase the number of iterations.

(ii) Local SGD (Zinkevich et al., 2010; Stich, 2019)**:** Every client performs multiple steps using the local gradients, and after that the results are averaged. Unfortunately, for some iteration, not all workers may be available for computation or communication, and a practical approach is to use updates from the clients which communicated first. **Federated Averaging** (McMahan et al., 2017; Li et al., 2019) mimics this behavior by subsampling the workers whose updates are used at the current iterations. However, these approaches require client communication and server broadcast even when the communicated data didn't significantly change.

(iii) Sparse communication on decentralized topologies: While relying on a parameter server to aggregate updates (Dean et al., 2012) or communication-heavy all-reduce, in decentralized training methods clients exchange model updates in a peer-to-peer fashion. This can alleviate communication bottlenecks in data-center training (Assran et al., 2019) and can be applied to arbitrary network topologies (Lian et al., 2017; Tang et al., 2018).

These techniques have been refined in follow up works and to some extent are orthogonal to each other, i.e. they can be applied on top of each other, see e.g. (Basu et al., 2020).

Most of the papers introducing the compression techniques only focus on compressing the uplink messages sent from the workers to the server, but do not compress donwlink broadcast messages from the server to the workers (Alistarh et al., 2017; Wu et al., 2018; Stich et al., 2018; Alistarh et al., 2018; Mishchenko et al., 2019; Gorbunov et al., 2020; Stich & Karimireddy, 2020; Stich, 2020; Rothchild et al., 2020a). A few recent works study bi-directional compression (Tang et al., 2019; Zheng et al., 2019; Liu et al., 2020; Yu et al., 2019; Philippenko & Dieuleveut, 2020). Decentralized techniques alleviate the broadcast by design and only exchange compressed messages (Tang et al., 2018; Koloskova et al., 2019; 2020a).

Distributed methods that use only intermittent communication most frequently communicate after a prescribed number of iterations (or epochs) on the local data (McMahan et al., 2017; Lin et al., 2020; Wang & Joshi, 2018), it is also possible to maintain a constant frequency only in expectation (Koloskova et al., 2020b), to increase the frequency during training (Wang & Joshi, 2019) or decrease the frequency (Haddadpour et al., 2019). However, for these methods, the communication frequency has to be fixed in advance.

In contrast, event triggered schemes do not follow prescribed communication patterns, but trigger communication events based on local (or global) decision rules, taking the problem data and algorithm state into account. Event triggered communication has been considered in the control community (Heemels et al., 2012; Dimarogonas et al., 2012) and optimization community (Kia et al., 2015; Chen & Ren, 2016; Hsieh et al., 2017; Chen et al., 2018; Kamp et al., 2019).

The most closely related work is the LENA (Shokri Ghadikolaei et al., 2021) framework, which was the first to introduce the combination of event-triggered communication and drifting. Drift can be seen as expected gradient step from a silent worker: if at the iteration t, worker i decided not to communicate, the server will assume the gradient of that worker is equal to the predefined drift value. Drift value can be updated when the next communication from that worker happens. While different triggering rules and drift strategies can be designed, it still requires broadcasting updated model parameters to all the workers every iteration, i.e. downlink stays the same. Thus even if all workers communicate infinitely rare, broadcasts will stay the same and effective reduction in latency is at most twice. In this paper, we challenge this limitation by introducing server triggers and global update drift (can be seen as a server drift), which is conceptually symmetric to the client drift. Main challenge in reducing the downlink is caused by dissynchronisation of workers drifts. In section 4 we compare our approach to the LENA framework (Shokri Ghadikolaei et al., 2021) experimentally.

1.2. Our contributions

Our main contribution is the framework PROCRASTINATOR (Algorithm 1) which allows both the server and the workers to control their communication, sending updates only when necessary. In a nutshell, each worker monitors the norm of an accumulated difference between the current gradient and the last communicated gradient, and delays communication until the norm of the accumulated difference passes a certain threshold. The server computes the average of the last communicated local gradients as an estimate of the average gradient which is broadcast to the workers in the similar pattern. Our framework allows one to control the communication frequency by specifying the appropriate thresholds. Regardless of the choices of the thresholds, we show that, with the appropriate step size, the algorithm converges to a local minimum:

Theorem 1 (Informal, see Theorem 5)**.** *For a Lipschitz function f, when the stochastic variance and the deviation between local gradients are bounded, after T iterations of Algorithm 2 we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 = O\left(\frac{1}{\sqrt{NT}} + \frac{1}{T^{2/3}}\right)$$

We emphasize that convergence of PROCRASTINATOR is a non-trivial result. The i-th client communicates when $\|\mathbf{e}_i^{(t+1)}\|^2 \geq A\|\mathbf{g}_i^{(t)}\|^2 + B$, where $\mathbf{e}_i^{(t+1)}$ is the accumulated error and $\mathbf{g}_i^{(t)}$ is the current stochastic gradient. Since $\mathbf{e}_i^{(t+1)}$ accumulates differences between gradients and the their estimate since last communication, it's possible that the estimate not only has a magnitude much larger compared with the current gradient, but also points at the arbitrary direction. Furthermore, since clients communicate their gradient at different times, their average can be not an estimation of the average gradient at *any* point. Therefore, it's not evident that using the average of local estimates as an estimate of the average gradient is the correct approach. Regardless, we show that with careful handling of the accumulated errors, the algorithm converges; moreover, its convergence rate is close to $O(1/\sqrt{NT})$ of distributed SGD (Bottou et al., 2018) and matches the latter when $N = O(\sqrt[3]{T})$ or when the number of iterations $T = \Omega(N^3)$ is sufficiently large.

Finally, we empirically show the convergence and communication improvements of our algorithm. We show that PROCRASTINATOR has the convergence rate similar to the distributed SGD and local SGD, while requiring significantly less communication compared with local SGD. Compared with LENA, our algorithm has similar number of client communications, while requiring substantially less server broadcasts.

2. Preliminaries

For a function $f\colon \mathbb{R}^n \to \mathbb{R}$, we consider the minimization problem $f(\mathbf{x}) \to \min$. We make the following standard assumptions (Bottou et al., 2018):

Assumption A. *$f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth, i.e. for all $\mathbf{x}, \mathbf{y}$: $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$.*

In distributed settings, we have N clients, where each client corresponds to its local function f_i such that $f(\mathbf{x}) = \operatorname{avg}_i f_i(\mathbf{x})$, where $\operatorname{avg}_i a_i = \frac{1}{N}\sum_{i=1}^N a_i$. Furthermore, for each client, we assume that we have access to the stochastic gradient oracle $\mathbf{g}_i$:

Assumption B. *For every client $i \in [N]$, the stochastic gradient is unbiased and has bounded variance:*

$$\mathbb{E}[\mathbf{g}_i(\mathbf{x})] = \nabla f_i(\mathbf{x}), \qquad \mathbb{E}\|\mathbf{g}_i(\mathbf{x}) - \nabla f_i(\mathbf{x})\|^2 \leq \sigma^2 .$$

Finally, we bound the deviation of local gradients from the global gradient:

Assumption C. *For every client $i \in [N]$, we have*

$$\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \alpha\|\nabla f(\mathbf{x})\|^2 + \beta .$$

We emphasize that Assumption C is significantly weaker compared with assumptions $\mathbb{E}\|\nabla F_i(\mathbf{x})\|^2 \leq G^2$ or $\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq G^2$ commonly used in the literature. Note that Assumption C is equivalent to the conditions $\|\nabla f_i(\mathbf{x})\|^2 \leq \alpha'\|\nabla f(\mathbf{x})\|^2 + \beta'$ and $\|\nabla f_i(\mathbf{x}) - \nabla f_j(\mathbf{x})\|^2 \leq \alpha''\|\nabla f(\mathbf{x})\|^2 + \beta''$.

3. Algorithm and Analysis

3.1. Algorithm

In distributed SGD, at every iteration, each client computes local gradient and communicates it to the server, which

Algorithm 1 PROCRASTINATOR
(for more detailed pseudocode refer to Appendix A)

1: **parameters:** step size γ, number of iterations T, client trigger parameters (A, B), server trigger parameters (C, D)
2: **inputs:** Initial point $\mathbf{x}^{(0)}$, stochastic gradient oracles $\mathbf{g}_i$
3: // drift and error accumulators for server and all clients
4: $\mathbf{u}^{(0)} \leftarrow \mathbf{0}$, $\mathbf{r}^{(0)} \leftarrow \mathbf{0}$, $\forall i : \mathbf{d}_i^{(0)} \leftarrow \mathbf{0}$, $\mathbf{e}_i^{(0)} \leftarrow \mathbf{0}$
5: **for** $t = 0, 1, 2, \dots, T-1$ **do**
6: **For each client $i \in [N]$:**
7: Compute local stochastic gradient $\mathbf{g}_i^{(t)}$
8: // Update local error: $+$ gradient $-$ local drift
9: $\mathbf{e}_i^{(t+1)} \leftarrow \mathbf{e}_i^{(t)} + \mathbf{g}_i^{(t)} - \mathbf{d}_i^{(t)}$
10: $\mathbf{d}_i^{(t+1)} \leftarrow \mathbf{d}_i^{(t)}$
11: // Check if the local error is large
12: **if** $\|\mathbf{e}_i^{(t+1)}\|^2 \geq A\|\mathbf{g}_i^{(t)}\|^2 + B$ **then**
13: // New local drift estimate
14: $\mathbf{d}_i^{(t+1)} \leftarrow \mathbf{g}_i^{(t)}$
15: **Send** $(\mathbf{e}_i^{(t+1)}, \mathbf{d}_i^{(t+1)})$ to the server
16: $\mathbf{e}_i^{(t+1)} \leftarrow 0$
17: **if** Didn't **receive** updated $(\mathbf{x}^{(t+1)}, \mathbf{u}^{(t+1)})$ from server **then**
18: // Use $\mathbf{u}^{(t)}$ for local step
19: $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)}$, $\quad \mathbf{u}^{(t+1)} \leftarrow \mathbf{u}^{(t)}$
Server Update:
20: Let $\mathcal{C}^{(t)} \subset [N]$ be the set of clients that send updated $(\mathbf{e}_i^{(t+1)}, \mathbf{d}_i^{(t+1)})$ at iteration t
21: **for** $i \in \mathcal{C}^{(t)}$ **do Receive** $(\mathbf{e}_i^{(t+1)}, \mathbf{d}_i^{(t+1)})$ from client i
22: **for** $i \notin \mathcal{C}^{(t)}$ **do** $\mathbf{d}_i^{(t+1)} \leftarrow \mathbf{d}_i^{(t)}$
23: // Update server error: + local step estimates - global step + communicated errors
24: $\mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t)} + \frac{1}{N}\sum_{i=1}^N(\mathbf{d}_i^{(t)} - \mathbf{u}^{(t)}) + \frac{1}{N}\sum_{i\in\mathcal{C}^{(t)}} \mathbf{e}_i^{(t+1)}$
25: $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)}$, $\quad \mathbf{u}^{(t+1)} \leftarrow \mathbf{u}^{(t)}$
26: // Check if the server error is large
27: **if** $\|\mathbf{r}^{(t+1)}\|^2 \geq C\|\mathrm{avg}_i(\mathbf{d}_i^{(t)})\|^2 + D$ **then**
28: // Average drift is local update at next iterations
29: $\mathbf{u}^{(t+1)} \leftarrow \frac{1}{N}\sum_{i=1}^N \mathbf{d}_i^{(t+1)}$
30: // Propagate server error
31: $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)} - \gamma\mathbf{r}^{(t+1)}$
32: **Broadcast** $(\mathbf{x}^{(t+1)}, \mathbf{u}^{(t+1)})$
33: $\mathbf{r}^{(t+1)} \leftarrow 0$

broadcasts the average gradient to all clients. After that, all clients perform a gradient descent step using this average. This approach requires each client and server to communicate at every iteration, which leads to extensive communication. The intuition behind our algorithm is to maintain estimates – of local gradient on the server and of average gradient on the clients – and communicate the actual values only when they significantly deviate from the estimates.

Our approach, PROCRASTINATOR, is presented as Algorithm 1. The server uses $\mathbf{d}_i^{(t)}$ to estimate local gradients (Lines 1 and 1) and clients use $\mathbf{u}^{(t)}$ to estimate the average gradient. Since $\mathbf{d}_i^{(t)}$ are the only estimates of local gradients available to the server, we compute $\mathbf{u}^{(t)}$ as an average of $\mathbf{d}_i^{(t)}$ (Line 1). Since $\mathbf{u}^{(t)}$ is the best estimate of the average gradient available to clients, the clients perform update $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)}$ instead of a gradient descent step (Line 1).

The key idea of our algorithm is to use *triggers* to control communication. Both client and server triggers have the following structure: they maintain an "error", which accumulates deviation of the actual value from the current estimate (Lines 9 and 24). When the accumulated error passes a certain threshold (Lines 12 and 27), a new estimate is communicated (Lines 15 and 32) and the error is reset. Note that, while the difference between the actual value and its estimate can be small, accumulated throughout multiple iterations it can substantially alter the algorithm behavior. To address this problem, the errors accumulate these differences since the last trigger activation.

3.2. Convergence Analysis

To analyze convergence of Algorithm 2, we introduce the sequence of *corrected iterates* $\{\mathbf{y}^{(t)}\}$ where $\mathbf{y}^{(0)} = \mathbf{x}^{(0)}$ and $\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \gamma\,\mathrm{avg}_i\,\mathbf{g}_i^{(t)}$. Unlike $\{\mathbf{x}^{(t)}\}$, the sequence uses gradients for updates, and such a sequence commonly used in the analysis of SGD with error-feedback (Stich et al., 2018; Karimireddy et al., 2019; Stich & Karimireddy, 2020). The sequence $\{\mathbf{y}^{(t)}\}$ has the following relation with $\{\mathbf{x}^{(t)}\}$:

Lemma 2. *For any t, $\mathbf{y}^{(t)} = \mathbf{x}^{(t)} - \gamma(\mathbf{r}^{(t)} + \mathrm{avg}_i\,\mathbf{e}_i^{(t)})$.*

Therefore, $\xi^{(t)} = \mathbf{r}^{(t)} + \mathrm{avg}_i\,\mathbf{e}_i^{(t)}$ is the full error. Based on the proof of Karimireddy et al. (2019, Theorem II), we have the following intermediate result:

Lemma 3. *Let $\xi^{(t)} = \mathbf{r}^{(t)} + \mathrm{avg}_i\,\mathbf{e}_i^{(t)}$. Then under Assumptions A and B, for every T we have*

$$\mathbb{E}[f(\mathbf{y}^{(T)})] \leq f(\mathbf{y}^{(0)}) - \sum_{t=0}^{T-1}\frac{\gamma}{2}(1 - L\gamma)\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + \frac{L^2\gamma^3}{2}\sum_{t=0}^{T-1}\mathbb{E}\|\xi^{(t)}\|^2$$

It remains to bound $\sum_{t=0}^{T-1}\mathbb{E}\|\xi^{(t)}\|^2$. The following lemma shows that it can be expressed in terms of $\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2$:

Lemma 4. *Under Assumption C, for every T we have:*

$$\sum_{t=0}^{T-1}\mathbb{E}\|\xi^{(t)}\|^2 \leq c_1\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + c_2 T\,,$$

where $c_1 = 6(1+A)(1+\alpha)$ and $c_2 = 6((1+A)\beta + (1+A)\sigma^2 + B)$.

Selecting $\gamma^2 \leq \frac{1}{4c_1 L^2}$ (and therefore $\gamma \leq \frac{1}{4L}$), from Lemma 3 we have:

$$\mathbb{E}[f(\mathbf{y}^{(T)})] \leq f(\mathbf{y}^{(0)}) - \frac{\gamma}{4}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + T\frac{L^2\gamma^3 c_2}{2}$$

After regrouping the terms and using that $f(\mathbf{y}^{(0)}) - \mathbb{E}[f(\mathbf{y}^{(T)})] \leq f_{\max}$:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 \leq \frac{4f_{\max}}{T\gamma} + \frac{4L\gamma\sigma^2}{2N} + 2L^2\gamma^2 c_2 \,.$$

We select the step size based on Koloskova et al. (2020b, Lemma 17). Intuitively, since the first term decreases with γ while the other terms increase, we need to select γ such that either $\gamma \leq \frac{1}{2\sqrt{c_1}L}$ (from the derivation above) or the first term is balanced with the second or the third one.

Theorem 5. *Under Assumptions A–C, after T iterations of Algorithm 2 with $\gamma = \min\left(\frac{\sqrt{2FN}}{\sqrt{L}\sigma}, \left(\frac{F}{c_2 L^2 T}\right)^{1/3}, \frac{1}{2L\sqrt{c_1}}\right)$, where c_1 and c_2 are defined as in Lemma 4, we have:*

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 \\
&= O\left(\frac{\sqrt{LF}\sigma}{\sqrt{NT}} + \left(\frac{FL\sqrt{c_2}}{T}\right)^{2/3} + \frac{FL\sqrt{c_1}}{T}\right) \\
&= O\left(\frac{1}{\sqrt{NT}} + \frac{1}{T^{2/3}}\right)
\end{aligned}$$

which matches the convergence rate $O(1/\sqrt{NT})$ of distributed SGD when $N = O(\sqrt[3]{T})$ or when the number of iterations $T = \Omega(N^3)$ is sufficiently large.

In the appendix, we present an additional result for the case when the objective is convex.

4. Experiments

In this section, we empirically show convergence and communication improvements of PROCRASTINATOR. We perform experiments on two datasets: MNIST (Lecun et al., 1998) and CIFAR-10 (Krizhevsky, 2012). For MNIST, we train a deep convolutional model with $\gamma = 0.1$ and batch size 8, while for CIFAR-10 we train the VGG neural network (Simonyan & Zisserman, 2014) with $\gamma = 0.01$ and batch size 100.

In our experiments, we consider the following approaches:

- **PROCRASTINATOR** with parameters (A, B, C, D) as in Algorithm 1.
- **LENA** (Shokri Ghadikolaei et al., 2021) with parameters (A, B). LENA is the special case of PROCRASTINATOR such that the server broadcasts at every iteration (or equivalently, $C = D = 0$).
- **Local SGD** with parameter *gap*. Each worker makes gradient descent step (i.e. $\mathbf{x}_i^{(t+1)} \leftarrow \mathbf{x}_i^{(t)} - \gamma\mathbf{g}_i^{(t)}$), and every *gap* iterations all worker synchronize their parameters.

For PROCRASTINATOR and LENA, we selected the best considered parameters[1]. For local SGD, we considered $gap = 1$ as the most basic baseline (the algorithm becomes the distributed SGD), and $gap = 5$, since with this gap, communication of Local SGD is close to per-iteration communication of PROCRASTINATOR and LENA.

Our results are shown in Figure 1. For each dataset, we report the following:

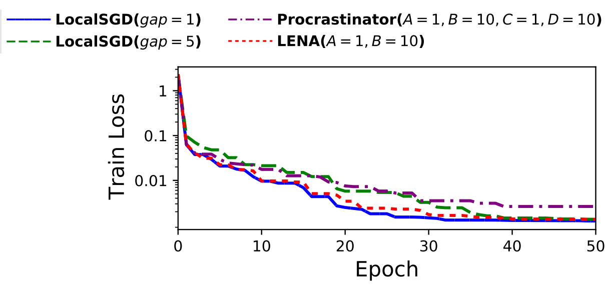
- Train loss with respect to the number of epochs, the number of client communications and the number of broadcasts.
- Client communication latency and broadcast latency. Namely, for each epoch we report the number of communications/broadcasts during that epoch.

For the MNIST dataset, Figure 1a shows that all algorithms have comparable convergence rates, and therefore, communication becomes the main deciding performance factor. In Figure 1c, LENA requires the least amount of communication, followed by PROCRASTINATOR and substantially outperforming local SGD and distributed SGD in terms of communication required to reach the final accuracy. However, LENA, similarly to distributed SGD, requires broadcasts at every iteration, and therefore is significantly outperformed by PROCRASTINATOR in this aspect.

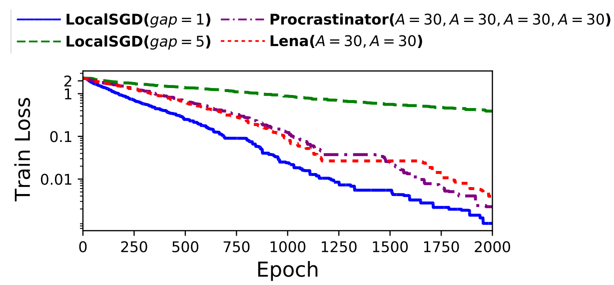
On CIFAR-10, algorithm behavior is noticeably different. In Figure 1b, distributed SGD clearly has the best performance, followed by PROCRASTINATOR and LENA. However, with respect to client communication, PROCRASTINATOR and LENA have the best performance, requiring 5x times less communication compared with distributed/local SGD. And due to broadcast requirements of LENA, PROCRASTINATOR shows the best communication performance: 3x times less broadcasts compared with distributed/local SGD.

Overall, PROCRASTINATOR outperforms local/distributed SGD with respect to communication. Compared with LENA, it shows similar client communication requirements, while requiring substantially less broadcasts.

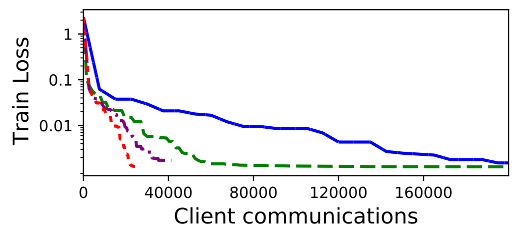
[1]We considered various combinations of parameters $A, B \in \{1, 10, 30, 60\}$. For larger values, the algorithms diverge. Due to the number of parameters, for PROCRASTINATOR we used $C = A$ and $D = B$.



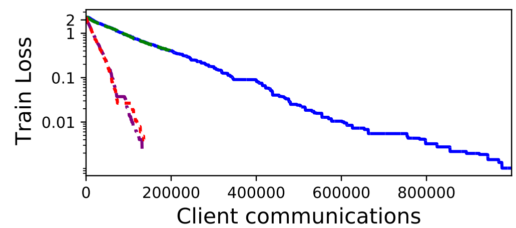
(a) MNIST: train loss w.r.t. epoch



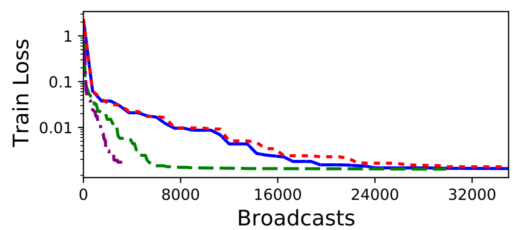
(b) CIFAR-10: train loss w.r.t. epoch



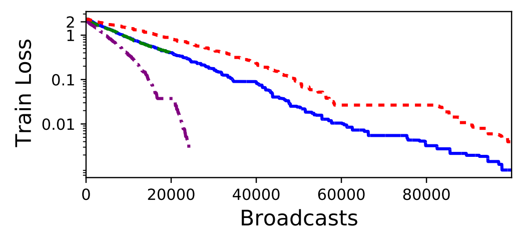
(c) MNIST: train loss w.r.t. client communications



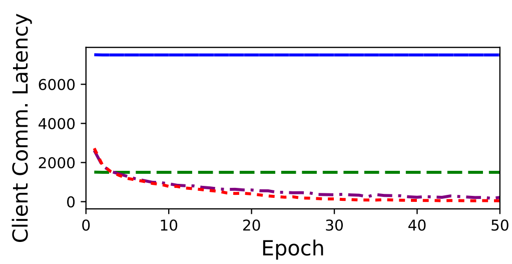
(d) CIFAR-10: train loss w.r.t. client communications



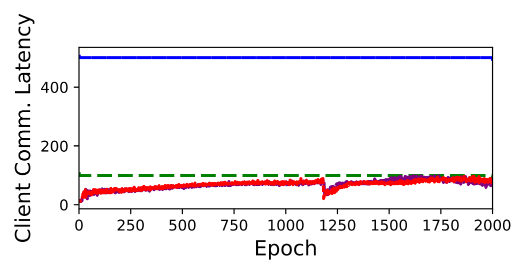
(e) MNIST: train loss w.r.t. broadcasts



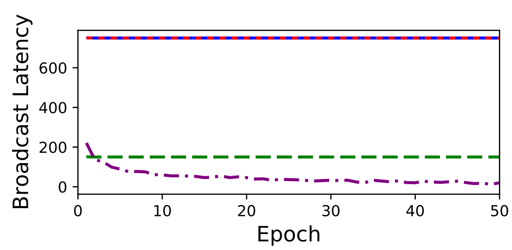
(f) CIFAR-10: train loss w.r.t. broadcasts



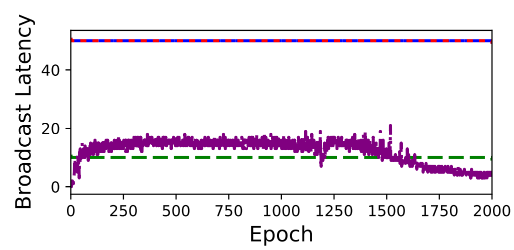
(g) MNIST: client communication latency



(h) CIFAR-10: client communication latency



(i) MNIST: broadcast latency



(j) CIFAR-10: broadcast latency

Figure 1. Convergence of distributed SGD (local SGD with $gap = 1$), local SGD with $gap = 5$, PROCRASTINATOR and LENA on MNIST (left) and CIFAR-10 datasets. For each dataset, we show train loss with respect to: the number of iterations, the number of communications from clients and the number of broadcasts. Additionally, we show the number of client communications and the number of broadcasts per epoch. The results show that PROCRASTINATOR and LENA have the best convergence w.r.t. client communication, while PROCRASTINATOR additionally has the best convergence w.r.t. broadcasts.

5. Conclusion

We present a new method – PROCRASTINATOR– to address the communication bottlenecks and latency in distributed optimization. As a distinguishing novelty, our scheme can automatically suppress broadcast of model updates to clients if the updated model state on the server does not deviate much from predicted values. If the clients do not receive a broadcast, they update their state according to a predefined rule which ensures that clients stay in sync – avoiding client drift (Kairouz et al., 2019; Karimireddy et al., 2020). This enables drastic savings in the total number of broadcasts, but also client uploads.

References

Aji, A. F. and Heafield, K. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 440–445, Copenhagen, Denmark, 2017. Association for Computational Linguistics. URL `http://aclweb.org/anthology/D17-1045`.

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pp. 1709–1720, 2017.

Alistarh, D., Hoefler, T., Johansson, M., Konstantinov, N., Khirirat, S., and Renggli, C. The convergence of sparsified gradient methods. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 5977–5987. Curran Associates, Inc., 2018. URL `http://papers.nips.cc/paper/7837-the-convergence-of-sparsified-gradient-methods.pdf`.

Assran, M., Loizou, N., Ballas, N., and Rabbat, M. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning*, pp. 344–353. PMLR, 2019.

Basu, D., Data, D., Karakus, C., and Diggavi, S. N. Qsparse-local-SGD: Distributed SGD with quantization, sparsification, and local computations. *IEEE J. Sel. Areas Inf. Theory*, 1(1):217–226, 2020. doi: 10.1109/jsait.2020.2985917. URL `https://doi.org/10.1109/jsait.2020.2985917`.

Bernstein, J., Wang, Y.-X., Azizzadenesheli, K., and Anandkumar, A. signSGD: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pp. 560–569. PMLR, 2018.

Bottou, L., Curtis, F., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018. URL `https://doi.org/10.1137/16M1080173`.

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., and Shi, W. Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67, 2018.

Chen, T., Giannakis, G., Sun, T., and Yin, W. LAG: Lazily aggregated gradient for communication-efficient distributed learning. In *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2018.

Chen, W. and Ren, W. Event-triggered zero-gradient-sum distributed consensus optimization over directed networks. *Automatica*, 65:90–97, 2016. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2015.11.015. URL `https://www.sciencedirect.com/science/article/pii/S0005109815004793`.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., Le, Q. V., and Ng, A. Y. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems*, pp. 1223–1231, 2012.

Dimarogonas, D. V., Frazzoli, E., and Johansson, K. H. Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, 57(5): 1291–1297, 2012. doi: 10.1109/TAC.2011.2174666.

Gorbunov, E., Kovalev, D., Makarenko, D., and Richtárik, P. Linearly converging error compensated SGD. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020. URL `https://arxiv.org/abs/2010.12292`.

Haddadpour, F., Kamani, M. M., Mahdavi, M., and Cadambe, V. R. Local SGD with periodic averaging: Tighter analysis and adaptive synchronization. In *Advances in Neural Information Processing Systems*, 2019.

Heemels, W., Johansson, K. H., and Tabuada, P. An introduction to event triggered and self-triggered control. In *IEEE Conference on Decision and Contro*, pp. 3270–3285, 2012.

Hsieh, K., Harlap, A., Vijaykumar, N., Konomis, D., Ganger, G. R., Gibbons, P. B., and Mutlu, O. Gaia: Geo-distributed machine learning approaching LAN speeds. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pp. 629–647, 2017.

Ivkin, N., Rothchild, D., Ullah, E., Braverman, V., Stoica, I., and Arora, R. Communication-efficient distributed SGD with sketching. *arXiv preprint arXiv:1903.04488*, 2019.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Kamp, M., Boley, M., Mock, M., Keren, D., Schuster, A., and Sharfman, I. Adaptive communication bounds for distributed online learning. *arXiv preprint arXiv:1911.12896*, 2019.

Karimireddy, S. P., Rebjock, Q., Stich, S., and Jaggi, M. Error feedback fixes SignSGD and other gradient compression schemes. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3252–3261. PMLR, 09–15 Jun 2019. URL `http://proceedings.mlr.press/v97/karimireddy19a.html`.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. SCAFFOLD: Stochastic controlled averaging for on-device federated learning. In *ICML - Proceedings of the 37th International Conference on Machine Learning*. PMLR, 2020. URL `https://arxiv.org/abs/1910.06378`.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017.

Kia, S. S., Cortés, J., and Martínez, S. Distributed convex optimization via continuous-time coordination algorithms with discrete-time communication. *Automatica*, 55:254–264, 2015. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2015.03.001. URL `https://www.sciencedirect.com/science/article/pii/S0005109815001053`.

Koloskova, A., Stich, S. U., and Jaggi, M. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 3478–3487. PMLR, 2019. URL `https://arxiv.org/abs/1902.00340`.

Koloskova, A., Lin, T., Stich, S. U., and Jaggi, M. Decentralized deep learning with arbitrary communication compression. *International Conference on Learning Representations*, 2020a. URL `https://arxiv.org/abs/1907.09356`.

Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., and Stich, S. U. A unified theory of decentralized SGD with changing topology and local updates. *Proceedings of the 37th International Conference on Machine Learning*, 2020b. URL `http://arxiv.org/abs/1602.05629`.

Krizhevsky, A. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems 30*, pp. 5330–5340. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7117-can-decentralized-algorithms-outperform-cent pdf`.

Lin, T., Stich, S. U., Patel, K. K., and Jaggi, M. Don't use large mini-batches, use local SGD. *International Conference on Learning Representations (ICLR)*, 2020. URL `https://arxiv.org/abs/1808.07217`.

Liu, X., Li, Y., Tang, J., and Yan, M. A double residual compression algorithm for efficient distributed learning. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 133–143. PMLR, 2020. URL `http://proceedings.mlr.press/v108/liu20a.html`.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

McMahan, H. B., Moore, E., Ramage, D., and y Arcas, B. A. Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629*, 2016. URL `http://arxiv.org/abs/1602.05629`.

Mishchenko, K., Gorbunov, E., Takáč, M., and Richtárik, P. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Philippenko, C. and Dieuleveut, A. Bidirectional compression in heterogeneous settings for distributed or federated learning with partial participation: tight convergence guarantees. *arXiv preprint arXiv:2006.14591*, 2020.

Rothchild, D., Panda, A., Ullah, E., Ivkin, N., Stoica, I., Braverman, V., Gonzalez, J., and Arora, R. FetchSGD: Communication-efficient federated learning with sketching. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8253–8265. PMLR, 13–18 Jul 2020a. URL `http://proceedings.mlr.press/v119/rothchild20a.html`.

Rothchild, D., Panda, A., Ullah, E., Ivkin, N., Stoica, I., Braverman, V., Gonzalez, J., and Arora, R. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pp. 8253–8265. PMLR, 2020b.

Shokri Ghadikolaei, H., Stich, S., and Jaggi, M. LENA: Communication-efficient distributed learning with self-triggered gradient uploads. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 3943–3951. PMLR, 2021. URL `http://proceedings.mlr.press/v130/shokri-ghadikolaei21a.html`.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Stich, S. U. Local SGD converges fast and communicates little. *International Conference on Learning Representations*, art. arXiv:1805.09767, 2019. URL `https://arxiv.org/abs/1805.09767`.

Stich, S. U. On communication compression for distributed optimization on heterogeneous data. *arXiv preprint arXiv:2009.02388*, 2020.

Stich, S. U. and Karimireddy, S. P. The error-feedback framework: SGD with delayed gradients. *Journal of Machine Learning Research*, 21(237):1–36, 2020. URL `http://jmlr.org/papers/v21/19-748.html`.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems 31*, pp. 4447–4458. Curran Associates, Inc., 2018.

Tang, H., Gan, S., Zhang, C., Zhang, T., and Liu, J. Communication compression for decentralized training. In *Advances in Neural Information Processing Systems 31*, pp. 7663–7673. Curran Associates, Inc., 2018.

Tang, H., Lian, X., Zhang, T., and Liu, J. Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. *arXiv preprint arXiv:1905.05957*, 2019.

Tomlinson, M., Solomon, W., Singh, Y., Doherty, T., Chopra, M., Ijumba, P., Tsai, A. C., and Jackson, D. The use of mobile phones as a data collection tool: a report from a household survey in south africa. *BMC medical informatics and decision making*, 9(1):51, 2009.

Vogels, T., Karimireddy, S. P., and Jaggi, M. PowerSGD: Practical low-rank gradient compression for distributed optimization. *arXiv preprint arXiv:1905.13727*, 2019.

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. URL `http://arxiv.org/abs/1808.07576`.

Wang, J. and Joshi, G. Adaptive communication strategies to achieve the best error-runtime trade-off in local-update SGD. In Talwalkar, A., Smith, V., and Zaharia, M. (eds.), *Proceedings of Machine Learning and Systems*, volume 1, pp. 212–229, 2019. URL `https://proceedings.mlsys.org/paper/2019/file/c8ffe9a587b126f152ed3d89a146b445-Paper.pdf`.

Wu, J., Huang, W., Huang, J., and Zhang, T. Error compensated quantized SGD and its applications to large-scale distributed optimization. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5325–5333. PMLR, 10–15 Jul 2018. URL `http://proceedings.mlr.press/v80/wu18d.html`.

Yu, Y., Wu, J., and Huang, L. Double quantization for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

Zheng, S., Huang, Z., and Kwok, J. T. Communication-efficient distributed blockwise momentum SGD with error-feedback. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

Zinkevich, M., Weimer, M., Li, L., and Smola, A. J. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 2595–2603, 2010.

A. Algorithm

Algorithm 2 shows the complete version of the algorithm.

Algorithm 2 Procrastinator

parameters: step size γ, number of iterations T, client trigger parameters (A, B), server trigger parameters (C, D)
inputs: Initial point $\mathbf{x}^{(0)}$, stochastic gradient oracles $\mathbf{g}_i$
$\mathbf{r}^{(0)} \leftarrow \mathbf{0}$ // Server error
Broadcast $\mathbf{x}^{(0)}$ to all clients
for each client $i \in [N]$ **do**
  $\mathbf{d}_i^{(0)} \leftarrow \mathbf{0}$ // client drift — server's estimation of client's gradient
  $\mathbf{u}^{(0)} \leftarrow \mathbf{0}$ // server drift during last broadcast — client's estimation of the global update
  $\mathbf{e}_i^{(0)} \leftarrow \mathbf{0}$ // Local error
for $t = 0, 1, 2, \ldots$ **do**

For each client $i \in [N]$:
  $\mathbf{g}_i^{(t)} \leftarrow \nabla F_i(\mathbf{x}^{(t)})$ // Compute local stochastic gradient

Client trigger:
  $\mathbf{e}_i^{(t+1/2)} \leftarrow \mathbf{e}_i^{(t)} + \mathbf{g}_i^{(t)} - \mathbf{d}_i^{(t)}$ // Update local error: + gradient - local drift
  // Check if the local error is large
  if $\|\mathbf{e}_i^{(t+1/2)}\|^2 \geq A\|\mathbf{g}_i^{(t)}\|^2 + B$ **then**
    $\mathbf{d}_i^{(t+1)} = \mathbf{g}_i^{(t)}$ // New local drift estimate
    Send $(\mathbf{e}_i^{(t+1/2)}, \mathbf{d}_i^{(t+1)})$ to the server
    $\mathbf{e}_i^{(t+1)} \leftarrow 0$
  else
    $\mathbf{d}_i^{(t+1)} \leftarrow \mathbf{d}_i^{(t)}$
    $\mathbf{e}_i^{(t+1)} \leftarrow \mathbf{e}_i^{(t+1/2)}$

  if Didn't **receive** updated $(\mathbf{x}^{(t+1)}, \mathbf{u}^{(t+1)})$ from server **then**
    $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)}, \quad \mathbf{u}^{(t+1)} \leftarrow \mathbf{u}^{(t)}$ // Use $\mathbf{u}^{(t)}$ for local step

Server Update:
  Let $\mathcal{C}^{(t)} \subset [N]$ be the set of clients that send updated $(\mathbf{e}_i^{(t+1/2)}, \mathbf{d}_i^{(t+1)})$ at iteration t
  for $i \in \mathcal{C}^{(t)}$ **do Receive** $(\mathbf{e}_i^{(t+1/2)}, \mathbf{d}_i^{(t+1)})$ from client i
  for $i \notin \mathcal{C}^{(t)}$ **do** $\mathbf{d}_i^{(t+1)} \leftarrow \mathbf{d}_i^{(t)}$

Server trigger:
  // Update server error: + local step estimates - global step + communicated errors
  $\mathbf{r}^{(t+1/2)} \leftarrow \mathbf{r}^{(t)} + \frac{1}{N}\sum_{i=1}^{N}(\mathbf{d}_i^{(t)} - \mathbf{u}^{(t)}) + \frac{1}{N}\sum_{i\in\mathcal{C}^{(t)}} \mathbf{e}_i^{(t+1/2)}$
  // Check if the server error is large
  if $\|\mathbf{r}^{(t+1/2)}\|^2 \geq C\|\mathrm{avg}_i(\mathbf{d}_i^{(t)})\|^2 + D$ **then**
    $\mathbf{u}^{(t+1)} \leftarrow \frac{1}{N}\sum_{i=1}^{N} \mathbf{d}_i^{(t+1)}$ // Average drift is local update at next iterations
    $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)} - \gamma\mathbf{r}^{(t+1/2)}$ // Propagate server error
    Broadcast $(\mathbf{x}^{(t+1)}, \mathbf{u}^{(t+1)})$
    $\mathbf{r}^{(t+1)} \leftarrow 0$
  else
    $\mathbf{x}^{(t+1)} \leftarrow \mathbf{x}^{(t)} - \gamma\mathbf{u}^{(t)}, \quad \mathbf{u}^{(t+1)} \leftarrow \mathbf{u}^{(t)}$ // Same as clients
    $\mathbf{r}^{(t+1)} \leftarrow \mathbf{r}^{(t+1/2)}$

B. Convergence Proof

B.1. Non-convex Case

Recall that $\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \gamma \operatorname{avg}_i(\mathbf{g}_i^{(t)})$. From Algorithm 2:

$$\mathbf{x}^{(t+1)} = \begin{cases} \mathbf{x}^{(t)} - \gamma \mathbf{u}^{(t)}, & \text{if broadcast doesn't happen at iteration } t \\ \mathbf{x}^{(t)} - \gamma \mathbf{u}^{(t)} - \gamma \mathbf{r}^{(t+1/2)}, & \text{if broadcast happens at iteration } t \end{cases}$$

We first show the relation between $\mathbf{x}^{(t)}$ and $\mathbf{y}^{(t)}$.

Lemma 6. *For any t, $\mathbf{y}^{(t)} = \mathbf{x}^{(t)} - \gamma(\mathbf{r}^{(t)} + \operatorname{avg}_i \mathbf{e}_i^{(t)})$.*

Proof. Proof by induction. The equality holds for $t = 0$. Assume that for some t, $\mathbf{y}^{(t)} = \mathbf{x}^{(t)} - \gamma(\mathbf{r}^{(t)} + \operatorname{avg}_i \mathbf{e}_i^{(t)})$.

Recall that $\mathcal{C}^{(t)}$ is the set of clients communicating at iteration t. By definition of $\mathbf{r}^{(t+1/2)}$ and $\mathbf{e}_i^{(t+1)}$, we have:

$$\begin{aligned}
&\mathbf{r}^{(t+1/2)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t+1)}) \\
&= \left(\mathbf{r}^{(t)} + \operatorname*{avg}_i(\mathbf{d}_i^{(t)} - \mathbf{u}^{(t)}) + \frac{1}{N}\sum_{i \in \mathcal{C}^{(t)}} \mathbf{e}_i^{(t+1/2)}\right) + \frac{1}{N}\sum_{i \notin \mathcal{C}^{(t)}} \mathbf{e}_i^{(t+1/2)} && \text{(Def. of } \mathbf{r}^{(t+1/2)} \text{ and } \mathbf{e}_i^{(t+1)}) \\
&= \mathbf{r}^{(t)} + \operatorname*{avg}_i(\mathbf{d}_i^{(t)} - \mathbf{u}^{(t)}) + \operatorname*{avg}_i(\mathbf{e}_i^{(t+1/2)}) \\
&= \mathbf{r}^{(t)} + \operatorname*{avg}_i(\mathbf{d}_i^{(t)} - \mathbf{u}^{(t)}) + \operatorname*{avg}_i(\mathbf{e}_i^{(t)}) + \operatorname*{avg}_i(\mathbf{g}_i^{(t)} - \mathbf{d}_i^{(t)}) && \text{(Def. of } \mathbf{e}_i^{(t+1/2)}) \\
&= \mathbf{r}^{(t)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t)}) + \operatorname*{avg}_i(\mathbf{g}_i^{(t)} - \mathbf{u}^{(t)})
\end{aligned} \tag{1}$$

If the broadcast doesn't happen at iteration t, then:

$$\begin{aligned}
\mathbf{y}^{(t+1)} - \mathbf{x}^{(t+1)} &= (\mathbf{y}^{(t)} - \mathbf{x}^{(t)}) - \gamma \operatorname*{avg}_i(\mathbf{g}_i^{(t)} - \mathbf{u}^{(t)}) \\
&= -\gamma(\mathbf{r}^{(t)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t)}) + \operatorname*{avg}_i(\mathbf{g}_i^{(t)} - \mathbf{u}^{(t)})) && \text{(IH)} \\
&= -\gamma(\mathbf{r}^{(t+1)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t+1)})) && \text{(Equation (1))}
\end{aligned}$$

If the broadcast happens, we have:

$$\begin{aligned}
\mathbf{y}^{(t+1)} - \mathbf{x}^{(t+1)} &= (\mathbf{y}^{(t)} - \mathbf{x}^{(t)}) - \gamma \operatorname*{avg}_i(\mathbf{g}_i^{(t)} - \mathbf{u}^{(t)} - \mathbf{r}^{(t+1/2)}) \\
&= -\gamma(\mathbf{r}^{(t)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t)}) + \operatorname*{avg}_i(\mathbf{g}_i^{(t)} - \mathbf{u}^{(t)}) - \mathbf{r}^{(t+1/2)}) && \text{(IH)} \\
&= -\gamma(\mathbf{r}^{(t+1/2)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t+1)}) - \mathbf{r}^{(t+1/2)}) && \text{(Equation (1))} \\
&= -\gamma(\mathbf{r}^{(t+1)} + \operatorname*{avg}_i(\mathbf{e}_i^{(t+1)})) && (\mathbf{r}^{(t+1)} = 0)
\end{aligned}$$

□

Lemma 7. *Let $\xi^{(t)} = \mathbf{r}^{(t)} + \operatorname{avg}_i \mathbf{e}_i^{(t)}$. Then for every T we have*

$$\mathbb{E}[f(\mathbf{y}^{(T)})] \le f(\mathbf{y}^{(0)}) - \sum_{t=0}^{T-1} \frac{\gamma}{2}(1 - L\gamma)\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + \frac{L^2\gamma^3}{2}\sum_{t=0}^{T-1}\mathbb{E}\|\xi^{(t)}\|^2$$

Proof. By the Descent Lemma:

$$\begin{aligned}
&\mathbb{E}_t[f(\mathbf{y}^{(t+1)})] \\
&\leq f(\mathbf{y}^{(t)}) - \langle \nabla f(\mathbf{y}^{(t)}), \mathbb{E}_t[\mathbf{y}^{(t+1)} - \mathbf{y}^{(t)}]\rangle + \frac{L}{2}\mathbb{E}_t\|\mathbf{y}^{(t+1)} - \mathbf{y}^{(t)}\|^2 \\
&= f(\mathbf{y}^{(t)}) - \gamma\langle \nabla f(\mathbf{y}^{(t)}), \mathbb{E}_t[\mathbf{g}^{(t)}]\rangle + \frac{L\gamma^2}{2}\mathbb{E}_t\|\mathbf{g}^{(t)}\|^2 \\
&= f(\mathbf{y}^{(t)}) - \gamma\langle \nabla f(\mathbf{y}^{(t)}), \nabla f(\mathbf{x}^{(t)})\rangle + \frac{L\gamma^2}{2}(\|\nabla f(\mathbf{x}^{(t)})\|^2 + \mathbb{E}_t\|\frac{1}{N}\sum_{i=1}^{N}(\mathbf{g}_i^{(t)} - \nabla f_i(\mathbf{x}^{(t)}))\|^2) \\
&= f(\mathbf{y}^{(t)}) - \gamma\langle \nabla f(\mathbf{y}^{(t)}), \nabla f(\mathbf{x}^{(t)})\rangle + \frac{L\gamma^2}{2}(\|\nabla f(\mathbf{x}^{(t)})\|^2 + \frac{\sigma^2}{N}),
\end{aligned}$$

where we used the fact that stochastic noises are independent. Since $\langle \nabla f(\mathbf{x}^{(t)}), \nabla f(\mathbf{x}^{(t)})\rangle = \|\nabla f(\mathbf{x}^{(t)})\|^2$ and using inequality $\langle a, b\rangle \leq \frac{1}{2}\|a\|^2 + \frac{1}{2}\|b\|^2$:

$$\begin{aligned}
&\mathbb{E}_t[f(\mathbf{y}^{(t+1)})] \\
&\leq f(\mathbf{y}^{(t)}) - \gamma\|\nabla f(\mathbf{x}^{(t)})\|^2 + \frac{L\gamma^2}{2}(\|\nabla f(\mathbf{x}^{(t)})\|^2 + \frac{\sigma^2}{N}) + \gamma\langle \nabla f(\mathbf{x}^{(t)}) - \nabla f(\mathbf{y}^{(t)}), \nabla f(\mathbf{x}^{(t)})\rangle \\
&\leq f(\mathbf{y}^{(t)}) - \gamma(1 - \frac{L\gamma}{2})\|\nabla f(\mathbf{x}^{(t)})\|^2 + \frac{L\gamma^2\sigma^2}{2N} + \frac{\gamma}{2}\|\nabla f(\mathbf{x}^{(t)}) - \nabla f(\mathbf{y}^{(t)})\|^2 + \frac{\gamma}{2}\nabla\|f(\mathbf{x}^{(t)})\|^2 \\
&\leq f(\mathbf{y}^{(t)}) - \frac{\gamma}{2}(1 - L\gamma)\|\nabla f(\mathbf{x}^{(t)})\|^2 + \frac{L\gamma^2\sigma^2}{2N} + \frac{L^2\gamma}{2}\|\mathbf{x}^{(t)} - \mathbf{y}^{(t)}\|^2 \\
&= f(\mathbf{y}^{(t)}) - \frac{\gamma}{2}(1 - L\gamma)\|\nabla f(\mathbf{x}^{(t)})\|^2 + \frac{L\gamma^2\sigma^2}{2N} + \frac{L^2\gamma^3}{2}\|\xi^{(t)}\|^2
\end{aligned}$$

Taking the expectation:

$$\mathbb{E}[f(\mathbf{y}^{(t)})] \leq f(\mathbf{y}^{(0)}) - \sum_{\tau=0}^{t-1}\frac{\gamma}{2}(1 - L\gamma)\mathbb{E}\|\nabla f(\mathbf{x}^{(\tau)})\|^2 + t\frac{L\gamma^2\sigma^2}{2N} + \frac{L^2\gamma^3}{2}\sum_{\tau=0}^{t-1}\mathbb{E}\|\xi^{(\tau)}\|^2 .$$

□

It suffices to bound $\sum_{\tau=0}^{t-1}\mathbb{E}\|\xi^{(\tau)}\|^2$.

Lemma 8. *Under Assumption C, for every T we have:*

$$\sum_{t=0}^{T-1}\mathbb{E}\|\xi^{(t)}\|^2 \leq c_1\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + c_2 T,$$

where $c_1 = 6(1 + A)(1 + \alpha)$ and $c_2 = 6((1 + A)\beta + (1 + A)\sigma^2 + B)$.

Proof. Since $\|\xi^{(t)}\|^2 \leq 2(\|\mathbf{r}^{(t)}\|^2 + \|\operatorname{avg}_i \mathbf{e}_i^{(t)}\|^2)$, we bound $\sum_t \mathbb{E}\|\mathbf{e}_i^{(t)}\|^2$ and $\sum_t \mathbb{E}\|\mathbf{r}^{(t)}\|^2$ separately.

Bounding $\sum_t \mathbb{E}\|\mathbf{e}_i^{(t)}\|^2$. If the client communicate to the server, then $\mathbf{e}^{(t+1)} = 0$. Otherwise, $\mathbf{e}^{(t+1)} = \mathbf{e}^{(t+1/2)}$, and by Line 2 of Algorithm 2, we have:

$$\mathbb{E}\|\mathbf{e}_i^{(t+1)}\|^2 \leq A\mathbb{E}\|\mathbf{g}_i^{(t)}\|^2 + B$$

and therefore

$$\sum_{t=0}^{T-1}\mathbb{E}\|\mathbf{e}_i^{(t)}\|^2 \leq A\sum_{t=0}^{T-1}\mathbb{E}\|\mathbf{g}_i^{(t)}\|^2 + BT$$

Bounding $\sum_t \mathbb{E}\|\mathbf{r}^{(t)}\|^2$. From Line 2 of Algorithm 2, we have:

$$\sum_{t=0}^{T-1} \|\mathbf{r}^{(t)}\|^2 \leq \sum_{t=0}^{T-1} \| \operatorname*{avg}_i(\mathbf{d}_i^{(t)})\|^2 \leq \sum_{t=0}^{T-1} \operatorname*{avg}_i \|\mathbf{d}_i^{(t)}\|^2 = \operatorname*{avg}_i \sum_{t=0}^{T-1} \|\mathbf{d}_i^{(t)}\|^2,$$

and therefore it suffices to bound $\sum_{t=0}^{T-1} \mathbb{E}\|\mathbf{d}_i^{(t)}\|^2$ for all i. For a client i, let t_1 and t_2 be the iterations such that the client communicated at times t_1 and t_2 and didn't communicate for any $t \in (t_1, t_2)$ ($t_1 = -1$ if t_2 is the earliest communication, $t_2 = T-1$ if t_1 is the latest communication). If $t_2 = t_1 + 1$, then $\sum_{t=t_1+1}^{t_2-1} \|\mathbf{d}_i^{(t)}\|^2$ is trivially 0. Otherwise, since the trigger didn't activate at iteration $(t_2 - 1)$, by Line 2 of Algorithm 2 we have

$$\|\mathbf{e}_i^{(t_2)}\|^2 = \| \sum_{t=t_1+1}^{t_2-1} (\mathbf{g}_i^{(t)} - \mathbf{d}_i^{(t)})\|^2 \leq A\|\mathbf{g}_i^{(t_2-1)}\|^2 + B$$

Using $\|a-b\|^2 \geq \frac{\|a\|^2}{2} - \|b\|^2$ and $\mathbf{d}_i^{(t_1+1)} = \mathbf{d}_i^{(t_1+2)} = \cdots = \mathbf{d}_i^{(t_2-1)}$:

$$\frac{1}{2}(t_2 - t_1 - 1) \sum_{t=t_1+1}^{t_2-1} \|\mathbf{d}_i^{(t)}\|^2 \leq \| \sum_{t=t_1+1}^{t_2-1} \mathbf{g}_i^{(t)}\|^2 + A\|\mathbf{g}_i^{(t_2-1)}\|^2 + B$$

By Cauchy-Schwarz, $\| \sum_{t=t_1+1}^{t_2-1} \mathbf{g}_i^{(t)}\|^2 \leq (t_2 - t_1 - 1) \sum_{t=t_1+1}^{t_2-1} \|\mathbf{g}_i^{(t)}\|^2$. Dividing the above inequality by $(t_2 - t_1 - 1)$, we have:

$$\sum_{t=t_1+1}^{t_2-1} \|\mathbf{d}_i^{(t)}\|^2 \leq 2 \sum_{t=t_1+1}^{t_2-1} \|\mathbf{g}_i^{(t)}\|^2 + \frac{2A}{t_2 - t_1 - 1}\|\mathbf{g}_i^{(t_2-1)}\|^2 + \frac{2B}{t_2 - t_1 - 1}$$

Since $\mathbf{d}_i^{(t_2)} = \mathbf{g}_i^{(t_1)}$ (defining $\mathbf{g}_i^{(-1)} = 0$ for $t_1 = -1$) and $t_2 - t_1 - 1 \geq 1$:

$$\sum_{t=t_1+1}^{t_2} \|\mathbf{d}_i^{(t)}\|^2 \leq 2 \sum_{t=t_1}^{t_2-1} \|\mathbf{g}_i^{(t)}\|^2 + 2A\|\mathbf{g}_i^{(t_2-1)}\|^2 + 2B$$

Finally, splitting $[0 : T-1]$ into $-1 = t_0 < t_2 < \ldots < t_k = T$ such that the i-th client communicates at iterations t_j, we have

$$\begin{aligned}
\sum_{t=0}^{T-1} \|\mathbf{d}_i^{(t)}\|^2 &= \sum_{j=0}^{k} \sum_{t=t_j+1}^{t_{j+1}} \|\mathbf{d}_i^{(t)}\|^2 \\
&\leq 2 \sum_{j=0}^{k} \left(\sum_{t=t_j+1}^{t_{j+1}} \|\mathbf{g}_i^{(t)}\|^2 + A\|\mathbf{g}_i^{(t_{j+1}-1)}\|^2 + B \right) \\
&\leq 2((1+A) \sum_{t=0}^{T-1} \|\mathbf{g}_i^{(t)}\|^2 + BT),
\end{aligned}$$

Bounding $\mathbf{g}_i^{(t)}$ in terms of $\nabla f(\mathbf{x}^{(t)})$. By Assumption C, we have

$$\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \alpha \|\nabla f(\mathbf{x})\|^2 + \beta$$

Using inequality $\|a-b\|^2 \geq \frac{1}{2}\|a\|^2 - \|b\|^2$, we have:

$$\frac{1}{2}\|\nabla f_i(\mathbf{x})\|^2 - \|\nabla f(\mathbf{x})\|^2 \leq \alpha \|\nabla f(\mathbf{x})\|^2 + \beta \implies \|\nabla f_i(\mathbf{x})\|^2 \leq 2(1+\alpha)\|\nabla f(\mathbf{x})\|^2 + 2\beta$$

Using $\mathbb{E}\|\mathbf{g}_i^{(t)}\|^2 = \mathbb{E}\|\nabla f_i(\mathbf{x}^{(t)})\|^2 + \mathbb{E}\|\mathbf{g}_i^{(t)} - \nabla f_i(\mathbf{x}^{(t)})\|^2 = \mathbb{E}\|\nabla f_i(\mathbf{x}^{(t)})\|^2 + \sigma^2$, for $\mathbf{g}_i^{(t)}$ we have

$$\mathbb{E}\|\mathbf{g}_i^{(t)}(\mathbf{x})\|^2 \leq 2(1+\alpha)\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + 2\beta + \sigma^2$$

Bounding $\sum_t \mathbb{E}\|\xi^{(t)}\|^2$. Putting the above bounds together, we have

$$\sum_{t=0}^{T-1} \mathbb{E}\|\xi^{(t)}\|^2 \leq 3((1+A)\sum_{t=0}^{T-1} \mathbb{E}\|\mathbf{g}_i^{(t)}\|^2 + BT)$$
$$\leq 6((1+A)(1+\alpha)\sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + (1+A)\beta T + (1+A)\sigma^2 T + BT). \qquad \square$$

Proof of the main theorem. By Lemma 7 and using bound on $\sum_{t=0}^{T-1} \mathbb{E}\|\xi^{(t)}\|^2$:

$$\begin{aligned}
&\mathbb{E}[f(\mathbf{y}^{(T)})] \\
&\leq f(\mathbf{y}^{(0)}) - \sum_{t=0}^{T-1} \frac{\gamma}{2}(1-L\gamma)\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + \frac{L^2\gamma^3}{2}\sum_{t=0}^{T-1} \mathbb{E}\|\xi^{(t)}\|^2 \\
&\leq f(\mathbf{y}^{(0)}) - \sum_{t=0}^{T-1} \frac{\gamma}{2}(1-L\gamma)\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + L^2\gamma^3(c_1\sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + Tc_2) \\
&= f(\mathbf{y}^{(0)}) - \sum_{t=0}^{T-1} \frac{\gamma}{2}(1-L\gamma - c_1 L^2\gamma^2)\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + c_2 T L^2\gamma^3 \\
&= f(\mathbf{y}^{(0)}) - \sum_{t=0}^{T-1} \frac{\gamma}{4}\mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 + T\frac{L\gamma^2\sigma^2}{2N} + c_2 T L^2\gamma^3,
\end{aligned}$$

where the last inequality obtained by selecting $\gamma \leq \frac{1}{4L}$ and $\gamma^2 \leq \frac{1}{4c_1L^2}$. Rearranging the terms and dividing by $\frac{\gamma T}{4}$, we have

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 \leq \frac{4(f(\mathbf{y}^{(0)}) - \mathbb{E}[f(\mathbf{y}^{(T)})])}{\gamma T} + \frac{2L\gamma\sigma^2}{N} + 4c_2L^2\gamma^2.$$

The rest of the proof follows that of Koloskova et al. (2020b, Lemma 17). Let $F = f(\mathbf{y}^{(0)}) - f^\star \geq f(\mathbf{y}^{(0)}) - \mathbb{E}[f(\mathbf{y}^{(T)})]$. Balancing the first two terms:

$$\frac{4F}{\gamma T} = \frac{2L\gamma\sigma^2}{N} \implies \gamma = \frac{\sqrt{2FN}}{\sqrt{L}\sigma}$$

Balancing the first and the last term:

$$\frac{4F}{\gamma T} = 4c_2L^2\gamma^2 \implies \gamma = \left(\frac{F}{c_2L^2T}\right)^{1/3}$$

Therefore, by selecting $\gamma = \min\left(\frac{\sqrt{2FN}}{\sqrt{L}\sigma}, \left(\frac{F}{c_2L^2T}\right)^{1/3}, \frac{1}{2L\sqrt{c_1}}\right)$:

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\mathbf{x}^{(t)})\|^2 = O\left(\frac{\sqrt{LF}\sigma}{\sqrt{NT}} + \left(\frac{FL\sqrt{c_2}}{T}\right)^{2/3} + \frac{FL\sqrt{c_1}}{T}\right). \qquad \square$$

B.2. Convex Case

Theorem 9. *Let f be a convex function satisfying Assumptions A-C. Let c_1 and c_2 be as defined in Lemma 8. Let $\{\mathbf{x}^{(t)}\}$ be the sequence from Algorithm 2 with $\gamma \leq \min\{\frac{1}{4(1+\alpha)}, \frac{1}{8\sqrt{c_1}L}\}$. Then*

$$\mathbb{E}[f(\frac{1}{T}\sum_{\tau=0}^{T-1} \mathbf{x}^{(\tau)}) - f(\mathbf{x}^\star)] \leq 16\gamma^2 c_2 L + \frac{2\|\mathbf{x}^{(0)} - \mathbf{x}^\star\|^2}{\gamma T} + 2\gamma(2\beta + \frac{\sigma^2}{N})$$

Therefore, for $\gamma = \Theta(1/\sqrt{T})$ we achieve $O(1/\sqrt{T})$ convergence rate.

Proof. Let $\mathbb{E}_t$ denote expectation conditioned on $\mathbf{x}^{(t)}, \mathbf{e}_i^{(t)}, \mathbf{r}^{(t)}$. Since $\mathbf{y}^{(t+1)} = \mathbf{y}^{(t)} - \gamma \operatorname{avg}_{i\in[N]}(\mathbf{g}_i^{(t)})$:

$$\begin{aligned}\mathbb{E}_t\|\mathbf{y}^{(t+1)} - \mathbf{x}^\star\|^2 &= \|\mathbf{y}^{(t)} - \gamma \operatorname*{avg}_{i\in[N]}(\mathbf{g}_i^{(t)}) - \mathbf{x}^\star\|^2 \\ &= \|\mathbf{y}^{(t)} - \mathbf{x}^\star\|^2 + \gamma^2\mathbb{E}_t\|\operatorname*{avg}_{i\in[N]}(\mathbf{g}_i^{(t)})\|^2 - 2\gamma\mathbb{E}_t\langle \operatorname*{avg}_{i\in[N]}(\mathbf{g}_i^{(t)}), \mathbf{y}^{(t)} - \mathbf{x}^\star\rangle \\ &= \|\mathbf{y}^{(t)} - \mathbf{x}^\star\|^2 + \gamma^2\mathbb{E}_t\|\operatorname*{avg}_{i\in[N]}(\mathbf{g}_i^{(t)})\|^2 - 2\gamma\langle \nabla f(\mathbf{x}^{(t)}), \mathbf{y}^{(t)} - \mathbf{x}^\star\rangle\end{aligned}$$

For the last term, we have:

$$\begin{aligned}-2\gamma\langle \nabla f(\mathbf{x}^{(t)}), \mathbf{y}^{(t)} - \mathbf{x}^\star\rangle &= -2\gamma\langle \nabla f(\mathbf{x}^{(t)}), \mathbf{y}^{(t)} - \mathbf{x}^{(t)}\rangle - 2\gamma\langle \nabla f(\mathbf{x}^{(t)}), \mathbf{x}^{(t)} - \mathbf{x}^\star\rangle \\ &\le 2\gamma^2\|\nabla f(\mathbf{x}^{(t)})\| \cdot \|\xi^{(t)}\| - 2\gamma\langle \nabla f(\mathbf{x}^{(t)}), \mathbf{x}^{(t)} - \mathbf{x}^\star\rangle\end{aligned}$$

Substituting this into the inequality above, by taking expectation and using telescoping, we have:

$$\begin{aligned}\mathbb{E}\|\mathbf{y}^{(t)} - \mathbf{x}^\star\|^2 \le{}& \|\mathbf{x}^{(0)} - \mathbf{x}^\star\|^2 + \gamma^2\sum_{\tau=0}^{t-1}\mathbb{E}\|\operatorname*{avg}_{i\in[N]}(\mathbf{g}_i^{(\tau)})\|^2 \\ &+ 2\gamma^2\sum_{\tau=0}^{t-1}\mathbb{E}[\|\nabla f(\mathbf{x}^{(\tau)})\| \cdot \|\xi^{(\tau)}\|] - 2\gamma\sum_{\tau=0}^{t-1}\mathbb{E}\langle \nabla f(\mathbf{x}^{(\tau)}), \mathbf{x}^{(\tau)} - \mathbf{x}^\star\rangle\end{aligned} \tag{2}$$

We'll simplify the terms on the right-hand side. Using the fact that stochastic noises are independent, as shown in Lemma 8:

$$\sum_{\tau=0}^{t-1}\mathbb{E}\|\mathbf{g}_i^{(\tau)}(\mathbf{x}^{(\tau)})\|^2 \le 2(1+\alpha)\sum_{\tau=0}^{t-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(\tau)})\|^2 + t(2\beta + \frac{\sigma^2}{N})$$

By Cauchy-Schwarz inequality and by Lemma 8:

$$\begin{aligned}\sum_{\tau=0}^{t-1}\mathbb{E}[\|\nabla f(\mathbf{x}^{(\tau)})\| \cdot \|\xi^{(\tau)}\|] &\le \sqrt{\mathbb{E}[\sum_{\tau=0}^{t-1}\|\nabla f(\mathbf{x}^{(\tau)})\|^2] \cdot \mathbb{E}[\|\sum_{\tau=0}^{t-1}\xi^{(\tau)}\|^2]} \\ &\le \sqrt{\mathbb{E}[\sum_{\tau=0}^{t-1}\|\nabla f(\mathbf{x}^{(\tau)})\|^2] \cdot \mathbb{E}[c_1\sum_{\tau=0}^{t-1}\|\nabla f(\mathbf{x}^{(\tau)})\|^2 + c_2 t]} \\ &\le \sqrt{c_1}\sum_{\tau=0}^{t-1}\mathbb{E}\|\nabla f(\mathbf{x}^{(\tau)})\|^2 + \sqrt{c_2}\sqrt{t}\sqrt{\mathbb{E}[\sum_{\tau=0}^{t-1}\|\nabla f(\mathbf{x}^{(\tau)})\|^2]}.\end{aligned}$$

By convexity:

$$-2\gamma\sum_{\tau=0}^{t-1}\mathbb{E}\langle \nabla f(\mathbf{x}^{(\tau)}), \mathbf{x}^{(\tau)} - \mathbf{x}^\star\rangle \le -2\gamma\sum_{\tau=0}^{t-1}\mathbb{E}[f(\mathbf{x}^{(\tau)}) - f(\mathbf{x}^\star)]$$

and for smooth convex functions we have:

$$\mathbb{E}\|\nabla f(\mathbf{x}^{(\tau)})\|^2 = \mathbb{E}\|\nabla f(\mathbf{x}^{(\tau)}) - \nabla f(\mathbf{x}^\star)\|^2 \le 2L\mathbb{E}[f(\mathbf{x}^{(\tau)}) - f(\mathbf{x}^\star)]$$

Let's denote $\sqrt{\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}[f(\mathbf{x}^{(\tau)})-f(\mathbf{x}^\star)]}$ as S_t. Substituting the bounds above into Equality (2) and dividing it by t:

$$0 \leq \frac{\|\mathbf{x}^{(0)}-\mathbf{x}^\star\|^2}{t} + 2\gamma^2(1+\alpha)S_t^2 + \gamma^2(2\beta + \frac{\sigma^2}{N}) + 2\gamma^2\sqrt{c_1}LS_t^2 + 4\gamma^2\sqrt{c_2 L}S_t - 2\gamma S_t^2$$

When $\gamma \leq \min\{\frac{1}{4(1+\alpha)}, \frac{1}{8\sqrt{c_1}L}\}$, it follows:

$$S_t^2 - 4\gamma\sqrt{c_2 L}S_t \leq \frac{\|\mathbf{x}^{(0)}-\mathbf{x}^\star\|^2}{\gamma t} + \gamma(2\beta + \frac{\sigma^2}{N})$$

By inequality $ab \leq \frac{a^2}{2} + \frac{b^2}{2}$, we have $4\gamma\sqrt{c_2 L}S_t \leq \frac{S_t^2}{2} + 8\gamma^2 c_2 L$, and therefore:

$$S_t^2 \leq 16\gamma^2 c_2 L + \frac{2\|\mathbf{x}^{(0)}-\mathbf{x}^\star\|^2}{\gamma t} + 2\gamma(2\beta + \frac{\sigma^2}{N})$$

Finally, by convexity:

$$S_t^2 = \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}[f(\mathbf{x}^{(\tau)})-f(\mathbf{x}^\star)] \geq \mathbb{E}[f(\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbf{x}^{(\tau)}) - f(\mathbf{x}^\star)]\,. \qquad \square$$