NET-FLEET: Achieving Linear Convergence Speedup for Fully Decentralized Federated Learning with Heterogeneous Data

Xin Zhang*, Minghong Fang+, Zhuqing Liu+, Haibo Yang+, Jia Liu+, and Zhengyuan Zhu*
*Department of Statistics, Iowa State University
+Department of Electrical and Computer Engineering, The Ohio State University

ABSTRACT

Federated learning (FL) has received a surge of interest in recent years thanks to its benefits in data privacy protection, efficient communication, and parallel data processing. Also, with appropriate algorithmic designs, one could achieve the desirable *linear speedup for convergence* effect in FL. However, most existing works on FL are limited to systems with i.i.d. data and centralized parameter servers and results on decentralized FL with heterogeneous datasets remains limited. Moreover, whether or not the linear speedup for convergence is achievable under *fully decentralized* FL with data heterogeneity remains an open question. In this paper, we address these challenges by proposing a new algorithm, called NET-FLEET, for fully decentralized FL systems with data heterogeneity. The key idea of our algorithm is to enhance the local update scheme in FL (originally intended for communication efficiency) by incorporating a recursive gradient correction technique to handle heterogeneous datasets. We show that, under appropriate parameter settings, the proposed NET-FLEET algorithm achieves a linear speedup for convergence. We further conduct extensive numerical experiments to evaluate the performance of the proposed NET-FLEET algorithm and verify our theoretical findings.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**.

KEYWORDS

Decentralized federated learning, optimization, algorithm design

ACM Reference Format:
Xin Zhang*, Minghong Fang+, Zhuqing Liu+, Haibo Yang+, Jia Liu+, and Zhengyuan Zhu*. 2022. NET-FLEET: Achieving Linear Convergence Speedup for Fully Decentralized Federated Learning with Heterogeneous Data. In *The Twenty-third International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc '22), October 17–20, 2022, Seoul, Republic of Korea.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3492866.3549723

1 INTRODUCTION

Federated learning (FL) is a powerful distributed training paradigm for modern large-scale machine learning [1, 2, 10, 11, 13, 16, 22, 33, 35–37]. FL leverages a large number of workers to collaboratively learn a global model. Mathematically, FL aims to solve an optimization problem in the form of:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) \triangleq \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x}), \tag{1}$$

where $f_i(\mathbf{x}) \triangleq \mathbb{E}_{\zeta\sim\mathcal{D}_i}[f_i(\mathbf{x};\zeta_i)]$ is the loss function of the data distribution $\mathcal{D}_i$ at worker i, and m is the number of workers. Different from traditional learning algorithms where data are collected and stored in a centralized server, FL allows the training data distributed at the workers, which could be smart phones, robots, network sensors, or other local information sources. A global model can be trained without the need to share the workers' data over the network, thus helping preserve data privacy. However, FL also faces several major technical challenges:

(C1). **Data Heterogeneity:** In conventional distributed learning, the data are either globally available or randomly shuffled and assigned to each worker. Thus, it is safe to assume that the data distributions at the workers are identical, i.e. $\mathcal{D}_i = \mathcal{D}_j, \forall i \in [m]$. Unfortunately, in FL systems, data are generated locally at each worker based on their own circumstances. As a result, data heterogeneity among the workers is unavoidable. Such data heterogeneity imposes significant challenges in designing FL algorithms and their training performance analysis.

(C2). **Unreliable Centralized Server:** Most current distributed learning systems are based on the server-worker architecture, where workers are coordinated by a centralized server. However, the centralized server may suffer several limitations, e.g., vulnerability to cyber-attacks and being a significant communication bottleneck. Additionally, in the context of FL, it is sometimes hard or even infeasible to find a trustworthy centralized server with whom all workers are willing to share information.

The above key challenges motivate us to consider *fully decentralized* FL systems (i.e., *without* any centralized server) deployed over peer-to-peer networks. Toward this end, in this paper, we focus on the fundamental "linear speedup for convergence" problem for decentralized FL under data heterogeneity. In the literature, it is well-known that the centralized-server-aided FL enjoys the "linear speedup for convergence" property. Specifically, the work in [28, 39] showed that the celebrated FedAvg algorithm and its variants under the homogeneous data setting can achieve a convergence rate of $O(1/\sqrt{mKS})$ with a sufficiently large communication rounds S, where m is the number of workers and K is the number of local update rounds. Notably, the $O(1/\sqrt{mKS})$ convergence rate implies a "linear speedup" with respect to the number of workers m. This is because, to attain an ϵ-accuracy in convergence, an algorithm with a convergence rate $O(1/\sqrt{S})$ takes $O(1/\epsilon^2)$ steps. In contrast,

MobiHoc '22, October 17–20, 2022, Seoul, Republic of Korea

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/3492866.3549723

an algorithm with a convergence rate $O(1/\sqrt{mS})$ needs $O(1/m\epsilon^2)$ steps (the hidden constant in Big-O is the same). In this sense, the convergence rate $O(1/\sqrt{mS})$ implies a *linear speedup* with respect to the number of workers. Such a linear speedup is highly desirable because it implies that one can efficiently leverage the massive parallelism in large-scale FL systems. However, under the data heterogeneity and unreliable centralized server challenges outline in (C1-C2), a fundamental open question arises: *Can we still achieve the state-of-the art linear speedup for convergence, i.e.,* $O(1/\sqrt{mKS})$*, under a fully decentralized FL system with data heterogeneity?*

In this paper, we give an *affirmative* answer to this question and propose a new *recursive gradient correction* based fully decentralized FL algorithm. Our main contributions are summarized as follows:

- To circumvent the unreliable centralized server challenge, we propose a fully decentralized network FL algorithm called *Decentralized Networked Federated Learning with Recursive Gradient Correction (NET-FLEET)*. In NET-FLEET, there is no centralized server and workers only need to share information with their neighboring nodes in each communication round. Similar to FedAvg-type algorithms, our proposed NET-FLEET algorithm allows the workers to run multiple local updates between two consecutive communication rounds with their neighbors, so as to reduce the communication load. By eliminating the centralized server, our NET-FLEET algorithm achieves gains in both robustness and flexibility.
- By proposing a new *recursively corrected stochastic gradient estimator* technique, our NET-FLEET algorithm works with decentralized network systems where workers hold heterogeneous datasets. It is worth noting that, although the conventional gradient tracking method [25, 26, 32] shares some similarity with our technique, the conventional gradient tracking method *cannot* be directly adopted in decentralized FL since the gradient estimators for local updates are not clearly defined in conventional gradient tracking. In contrast, our new corrected gradient estimator efficiently approximates the global stochastic gradient, so that it can handle data heterogeneity in decentralized FL.
- We establish theoretical guarantees for the convergence performance of NET-FLEET. The key challenge in the analysis is to examine the local model consensus error caused by *multiple* local updates contained in one round of fully decentralized model averaging. So far, most theoretical results in the FL literature rely on the assumption of homogenous datasets or gradient dissimilarity conditions. In this work, we relax these conditions and show that our proposed algorithm enjoys an $O(1/\sqrt{mSK})$ convergence rate with *arbitrary* heterogeneous datasets. Our result implies a *linear speedup* for convergence with respect to the worker number. Notably, our analysis and convergence results do not require the bounded gradient and homogeneous data assumptions, which could be of independent interest to general non-convex FL problems.

Collectively, our results in this paper contribute to the state of the art of decentralized FL with data heterogeneity. The rest of the paper is organized as follows. In Section 2, we review the literature to put our work in comparative perspectives. In Section 3, we formally state decentralized FL problem and propose our NET-FLEET algorithm. The convergence rate and complexity analysis of our algorithms are provided in Section 4. We provide numerical results in Section 5 to verify the theoretical results of our algorithms. In Section 6, we provide concluding remarks and discussions.

2 RELATED WORK

In this section, we provide a quick overview on recent related work on FL algorithms with homogeneous and heterogeneous datasets, as well as algorithms for fully decentralized FL in the literature.

1) FL with Homogenous Datasets: The federated averaging (FedAvg) algorithm, also known as "Local SGD," was first developed by [23] as a heuristic approach to address FL. FedAvg lets workers run K successive SGD updates with local data before communicating with the central server, thus achieving better communication efficiency than the traditional parallel SGD. Since then, FedAvg has sparked a large number of follow-ups that focus on theoretical performance of FL with homogeneous data (see, e.g., [20, 28–30, 39]). Under the homogeneous data assumption, most of the works provide a linear speedup for convergence, i.e. an $O(1/\sqrt{mSK})$, for a sufficiently large communication rounds S, which matches the state-of-the-art convergence rate of the parallel SGD [3, 6]. Furthermore, it has also been shown in [20] that FedAvg enjoys a better generalization performance than parallel SGD. We refer readers to excellent recent surveys [10, 16] for a comprehensive review.

2) FL with Heterogeneous Datasets: More recently, researchers have started to investigate the performance of FedAvg and its variants for FL with heterogeneous datasets. The work in [43] first showed that the accuracy of FL degrades significantly for neural networks trained on highly skewed heterogeneous datasets. They explained such accuracy degradation by the weight divergence, which can be quantified by the Wasserstein distance between the population data distributions and the workers' data distributions. To mitigate such worker-drift effects, they proposed a strategy to improve training with heterogeneous data by sharing a small subset of data between all the workers. So far, most of the existing theoretical work in the literature (see, e.g., [7, 27, 31, 39]) analyzed FedAvg's worker-drift with a (G, B)-bounded gradient dissimilarity assumption (GBD assumption), i.e., $\frac{1}{m}\sum_{i=1}^{m}\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq G^2 + B^2\|\nabla f(\mathbf{x})\|^2$, $\forall i \in [m]$. With the (G, B)-GBD assumption. These works showed that FedAvg could achieve a linear speedup for convergence with the rounds of local updates K being $\sqrt[3]{S}/m$. To relax the extra assumption on gradients, the work in [19] proposed a Variance Reduced Local SGD (VRL-SGD) algorithm for FL with heterogeneous data. VRL-SGD introduces an auxiliary variable to track average deviation between the local gradients and the corresponding global gradient of the same model parameters, and uses it to approximate the global gradients during the local SGD updates.

To further reduce the communication complexity, the work in [34] recently developed a generalized FedAvg (G-FedAvg) algorithm with two-sided learning rates and improved K to be as large as S/m. In G-FedAvg, the workers first run local updates with a local step-size, then upload the local parameter changes to the centralized server. Upon receiving workers' information, the server updates the global model parameter with the local changes and a server-side step-size. Due to the two-sided learning rates, the G-FedAvg achieved a linear speedup for convergence with a large K. But their analysis and convergence results are still limited by

the dissimilarity of local gradients. The work in [12] proposed a Stochastic Controlled Averaging (SCAFFOLD) algorithm, which corrects the worker-drift problem also by utilizing two-sided learning rates and control variables. SCAFFOLD estimates the worker-drift by the difference between the server-side and worker-side control variables and uses it to correct the local update. After K rounds of local updates, the workers send the local parameter changes to the centralized server for server-side update. By using the two-sided step-sizes and control variables, SCAFFOLD achieves a linear speedup for convergence without making assumptions on gradients. However, the aforementioned algorithms only work for the systems with a centralized parameter server.

3) Decentralized FL Algorithms: Decentralized FL has also received increasing attention recently, which is motivated by the fact that in some FL scenarios, the centralized server is not trustable. For example, the work in [17] proposed a Local Decentralized SGD (LD-SGD) algorithm for decentralized FL. LD-SGD can be viewed as a variant of the well-known Decentralized SGD (DSGD) algorithm [18, 24, 40, 41]. In LD-SGD, the workers perform multiple local updates and then communicate with their neighbors to perform one round of parameter aggregation. It is shown that LD-SGD could achieve a linear speedup for convergence under the bounded gradient assumption. Recently, the work in [5] developed a periodic decentralized momentum SGD (PD-SGDM) algorithm, which uses the gradient momentum term to improve the convergence performance. With a bounded gradient assumption, PD-SGDM can achieve a linear speedup for convergence as long as the rounds of local updates is bounded by $K = \sqrt[3]{S}/m$, which matches the number of local updates of the FedAvg algorithm. The work in [38] also proposed a decentralized momentum SGD algorithm with local updates. Unlike the PD-SGDM which assumes the bounded gradient, [38] leverages the generalized GBD assumption to handle the data heterogeneity and achieve the same linear speedup. In this work, we aim to achieve a linear speedup for decentralized federated learning without any assumption on gradient boundedness.

The most related work to our NET-FLEET is the decentralized FL stochastic gradient tracking (DSGT) algorithm proposed by [22]. In DSGT, the workers first run K rounds local SGD updates and then perform one round of stochastic gradient tracking update. However, the authors only provided a convergence analysis for the case with $K = 0$, i.e., no local update. In comparison, our NET-FLEET algorithm employs a local update scheme with a new *recursive gradient correction* technique. We show that NET-FLEET achieves a linear speedup for convergence with local updates rounds $K = \sqrt[3]{S}/m$ *without* any bounded gradient assumption.

3 PROBLEM STATEMENT AND ALGORITHM DESIGN

In this section, we will first state the fully decentralized FL problem. Then, we will present our NET-FLEET algorithm.

3.1 Decentralized Federated Learning

In the fully decentralized FL scenario, the workers form a peer-to-peer network system, which can be represented by an undirected connected graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$. Here, $\mathcal{N}$ and $\mathcal{L}$ are the sets of workers and edges, respectively, with $|\mathcal{N}| = m$. The workers are capable of local computation and communicating with their neighboring workers via the edges in $\mathcal{L}$. The goal of fully decentralized FL is to have the workers *distributively* and *collaboratively* solving the global optimization problem in the following form:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x}), \tag{2}$$

where each local objective function $f_i(\mathbf{x}) \triangleq \mathbb{E}_{\zeta\sim\mathcal{D}_i} f_i(\mathbf{x};\zeta)$ is only observable to worker i and not necessarily convex. Here, $\mathcal{D}_i$ represents the distribution of the dataset at node i, which is *heterogeneous* across workers. To solve Problem (2) in a decentralized fashion, one can reformulate Problem (2) in the following equivalent form by introducing a local model copy at each worker:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i = \mathbf{x}_j, \qquad \forall (i,j) \in \mathcal{L}. \end{aligned} \tag{3}$$

where $\mathbf{x} \triangleq [\mathbf{x}_1^\top, \cdots, \mathbf{x}_m^\top]^\top$, and $\mathbf{x}_i$ is an introduced local copy at worker i. To solve Problem (3), we consider an ϵ^2-stationary point $\mathbf{x}$ defined as follows:

$$\underbrace{\Big\|\frac{1}{m}\sum_{i=1}^{m}\nabla f_i(\bar{\mathbf{x}})\Big\|^2}_{\text{Global gradient magnitude}} + \underbrace{\frac{1}{m}\sum_{i=1}^{m}\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2}_{\text{Consensus error}} \le \epsilon^2, \tag{4}$$

where $\bar{\mathbf{x}} \triangleq \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i$ represents the global average across all workers. Unlike the ϵ^2-stationary point for centralized FL, the above criterion in Eq. (4) includes two components: the first term is the gradient norm of the global loss function and the second term is the average consensus error across all local copies. In this work, we aim to develop an efficient algorithm to attain an ϵ^2-stationary point for fully decentralized FL with heterogeneous datasets and study its speedup performance as the number of workers increases.

3.2 The NET-FLEET Algorithm

Now, we present our Decentralized Networked Federated Learning with Recursive Gradient Correction (NET-FLEET) algorithm. To solve Problem (1) in decentralized network systems where workers reach a *consensus* on a global optimal solution, a common approach in the literature is to let workers aggregate neighboring information through a consensus matrix $\mathbf{W} \in \mathbb{R}^{m\times m}$. Let $[\mathbf{W}]_{ij}$ represent the element in the i-th row and the j-th column in $\mathbf{W}$. Then, a consensus matrix $\mathbf{W}$ should satisfy the following properties:

(a) *Doubly Stochastic:* $\sum_{i=1}^{m}[\mathbf{W}]_{ij} = \sum_{j=1}^{m}[\mathbf{W}]_{ij} = 1$.
(b) *Symmetric:* $[\mathbf{W}]_{ij} = [\mathbf{W}]_{ji}, \forall i, j \in \mathcal{N}$.
(c) *Network-Defined Sparsity Pattern:* $[\mathbf{W}]_{ij} > 0$ if $(i,j) \in \mathcal{L}$; otherwise $[\mathbf{W}]_{ij} = 0, \forall i, j \in \mathcal{N}$.

The above properties imply that the eigenvalues of $\mathbf{W}$ are real and can be sorted as $-1 < \lambda_m(\mathbf{W}) \le \cdots \le \lambda_2(\mathbf{W}) < \lambda_1(\mathbf{W}) = 1$. We define the second-largest eigenvalue in magnitude of $\mathbf{W}$ as $\lambda \triangleq \max\{|\lambda_2(\mathbf{W})|, |\lambda_m(\mathbf{W})|\}$ for further notation convenience. It can be seen later that λ plays an important role in the step-size selection and characterizing the algorithm's convergence rate.

Similar to the centralized-server-based FL, a key defining feature in decentralized FL is that it allows workers to update the local model parameters *multiple rounds* before workers' communication

and model averaging. However, with heterogeneous data at different workers, the update directions (i.e., the stochastic gradients) are not identically distributed. Thus, after several local update rounds, the local parameters will move towards their worker-side optimum $\mathbf{x}^{*(i)}$, where $\mathbf{x}^{*(i)} = \arg\min f_i(\mathbf{x})$. This phenomenon may cause divergence of the algorithm and is often referred to as the "worker-drift problem." Moreover, the lack of a centralized sever further worsens the worker-drift problem. To address this challenge, in our NET-FLEET algorithm, we introduce an auxiliary parameter $\mathbf{y}^{(i)}$ at each worker i to approximate the global stochastic gradients. Our NET-FLEET algorithm is illustrated in Algorithm 1.

Specifically, NET-FLEET has K inner loops at each worker for the local updates between two consecutive outer loop iterations for inter-worker communications. Also, there are S rounds of inter-worker communications. At each outer loop iteration s, workers share the local model parameter $\mathbf{x}_{s,0}^{(i)}$ and the corrected gradient parameter $\mathbf{y}_{s,0}^{(i)}$ with neighboring workers, and initialize the inner-loop's starting points as $\mathbf{x}_{s,1}^{(i)}$ and $\mathbf{y}_{s,1}^{(i)}$ based on the neighboring average and local stochastic gradient update. Then, within the local inner loops, the update of $\mathbf{y}^{(i)}$ follows a recursive structure:

$$\mathbf{y}_{s,k+1}^{(i)} = \mathbf{y}_{s,k}^{(i)} + \mathbf{g}_{s,k+1}^{(i)} - \mathbf{g}_{s,k}^{(i)}, \ \forall k \in 1, \cdots, K-1, \tag{5}$$

where s and k are the indices of outer and inner loops, respectively, and $\mathbf{g}_{s,k}^{(i)} = \nabla f_i(\mathbf{x}_{s,k}^{(i)}; \boldsymbol{\zeta}_{s,k}^{(i)})$ is the local stochastic gradient with random sample $\boldsymbol{\zeta}_{s,k}^{(i)}$. In (5), it can be easily verified that the correction term follows $\mathbf{y}_{s,k}^{(i)} - \mathbf{g}_{s,k}^{(i)} = \mathbf{y}_{s,1}^{(i)} - \mathbf{g}_{s,1}^{(i)} = \sum_{j\in\mathcal{N}_i}[\mathbf{W}]_{ij}\mathbf{y}_{s,0}^{(i)} - \mathbf{g}_{s,0}^{(i)}$, which measures the difference between the local stochastic gradient and neighboring weighted-average update direction. By adding such correction term to $\mathbf{g}_{s,k+1}^{(i)}$, $\mathbf{y}_{s,k+1}^{(i)}$ will be close to the global stochastic gradient as outer loop iteration s gets large. Note that in NET-FLEET, the model parameter $\mathbf{x}$ is updated SK times, but the number of information communication rounds between workers is only S times. Thus, compared with traditional decentralized learning algorithms, NET-FLEET reduces the overall communication cost by a $1/K$ factor.

Remark 1. Some important remarks regarding our recursive gradient correction technique are in order. First, we note that the idea of gradient correction has appeared in the literature, including stochastic variance reduction (SVR) method in SVRG[9]/SPIDER[4], gradient tracking (GT) method in GNSD[21]/GT-DSGD[32], etc. However, the key differences between our method and these existing works are: 1) The SVR method requires a precise global gradient estimation at each outer loop iteration, while in our method the outer loops' gradient estimator is based on an *inexact* neighboring averaging and recursive correction; 2) The GT method is designed with a single-loop structure and demands one round of communication after each local update, thus suffering *high communication costs*. This limitation is due to the iterates' contraction result in the conventional convergence analysis for the GT method (cf. [21, Lemma 3]), which does not hold for multiple local updates. In contrast, our new recursive gradient correction method works with *multiple* local updates under decentralized FL. In this sense, the GT method is a special case of our method when local updates $K = 1$.

Algorithm 1 The NET-FLEET Algorithm.

Input: Initial point $\mathbf{x}^0$, learning rate η, communication rounds S, local update rounds K.

Set $\mathbf{x}_{0,0}^{(i)} = \mathbf{x}^0$ and $\mathbf{y}_{0,0}^{(i)} = \mathbf{g}_{0,0}^{(i)} = \nabla f_i(\mathbf{x}_{0,0}^{(i)}; \boldsymbol{\zeta}_{0,0}^{(i)})$ at worker i, for all $i \in [m]$.
for $s = 0, \cdots, S-1$ **do**
  for worker i, $i \in [m]$ **do**
    Share $(\mathbf{x}_{s,0}^{(i)}, \mathbf{y}_{s,0}^{(i)})$ with neighboring nodes;
    Update $\mathbf{x}_{s,1}^{(i)} = \sum_{j\in\mathcal{N}_i}[\mathbf{W}]_{ij}\mathbf{x}_{s,0}^{(j)} - \eta\mathbf{y}_{s,0}^{(i)}$;
    Calculate $\mathbf{g}_{s,1}^{(i)} = \nabla f_i(\mathbf{x}_{s,1}^{(i)}; \boldsymbol{\zeta}_{s,1}^{(i)})$;
    Correct $\mathbf{y}_{s,1}^{(i)} = \sum_{j\in\mathcal{N}_i}[\mathbf{W}]_{ij}\mathbf{y}_{s,0}^{(j)} + \mathbf{g}_{s,1}^{(i)} - \mathbf{g}_{s,0}^{(i)}$;
    for $k = 1, \cdots, K-1$ **do**
      Update $\mathbf{x}_{s,k+1}^{(i)} = \mathbf{x}_{s,k}^{(i)} - \eta\mathbf{y}_{s,k}^{(i)}$;
      Calculate $\mathbf{g}_{s,k+1}^{(i)} = \nabla f_i(\mathbf{x}_{s,k+1}^{(i)}; \boldsymbol{\zeta}_{s,k+1}^{(i)})$;
      Correct $\mathbf{y}_{s,k+1}^{(i)} = \mathbf{y}_{s,k}^{(i)} + \mathbf{g}_{s,k+1}^{(i)} - \mathbf{g}_{s,k}^{(i)}$
    end for
    Set $\mathbf{x}_{s+1,0}^{(i)} = \mathbf{x}_{s,K}^{(i)}$, $\mathbf{y}_{s+1,0}^{(i)} = \mathbf{y}_{s,K}^{(i)}$, $\mathbf{g}_{s+1,0}^{(i)} = \mathbf{g}_{s,K}^{(i)}$;
  end for
end for

4 THEORETICAL PERFORMANCE ANALYSIS

In this section, we will establish the convergence properties of our proposed NET-FLEET algorithm. Due to space limitation, we outline the key steps of the proofs of Theorem 1. We relegate the proof details to the supplementary material. We start with stating the following assumptions:

Assumption 1. *The objectives $f(\cdot)$ and $f_i(\cdot)$ satisfy:*

(1) $f(\mathbf{x})$ is bounded from below, i.e., there exists an $\mathbf{x}^* \in \mathbb{R}^p$, such that $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, $\forall \mathbf{x} \in \mathbb{R}^p$;

(2) The function $f_i(\mathbf{x})$ is continuously differentiable and has L-Lipschitz continuous gradients, i.e., there exists a constant $L > 0$ such that $|\nabla f_i(\mathbf{x}_1) - \nabla f_i(\mathbf{x}_2)| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|_2$, $\forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^p$;

(3) The stochastic gradient is unbiased and has bounded variance with respect to the local data distribution, i.e., $\mathbb{E}_{\zeta\sim\mathcal{D}_i}[\nabla f_i(\mathbf{x};\zeta)] = \nabla f_i(\mathbf{x})$ and $Var_{\zeta\sim\mathcal{D}_i}[\nabla f_i(\mathbf{x};\zeta)] \leq \sigma^2$ for some constant $\sigma > 0$.

It is worth noting that we do not need the conventional bounded gradient variability assumption in most of the literature of FL with non-i.i.d. datasets. To analyze the algorithm convergence, we define a potential function as

$$\mathfrak{P}_{s,k} \triangleq f(\bar{\mathbf{x}}_{s,k}) + \frac{1}{m^2K}\sum_{i=1}^{m}(\|\mathbf{x}_{s,k}^{(i)} - \bar{\mathbf{x}}_{s,k}\|^2 + C_1\eta^2\|\mathbf{y}_{s,k}^{(i)} - \bar{\mathbf{y}}_{s,k}\|^2),$$

where $C_1 = 6(1+\lambda K-\lambda)K/(1-\lambda)^2$ and $\bar{\mathbf{x}}_{s,k} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_{s,k}^{(i)}$, $\bar{\mathbf{y}}_{s,k} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{y}_{s,k}^{(i)}$. With the above assumptions and definitions, we are now in a position to present the main convergence result for our NET-FLEET algorithm as follows:

Theorem 1 (Convergence of NET-FLEET). *Under Assumption 1, if the step-size η in Algorithm 1 satisfies:*

$$\eta \le \min\Big\{\frac{1}{3L}, \frac{1}{mL^2K^2}, \frac{(1-\lambda)}{\sqrt{12(1+\lambda K-\lambda)KL^2}}, \sqrt{\frac{1-\lambda}{24L^2K^2}}, \frac{(1-\lambda)^3mK}{144}, \sqrt{\frac{m(1-\lambda)^2}{144LK^2}}, \frac{(1-\lambda)}{3(1+\lambda K-\lambda)mKL^2}, \frac{(1-\lambda)^2(1+\lambda K-\lambda)m}{144K}\Big\},$$

then Algorithm 1 has the following convergence result:

$$\frac{1}{SK}\sum_{s=0}^{S-1}\sum_{k=0}^{K-1}\mathbb{E}\big[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\sum_{i=1}^{m}\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2\big] \le \frac{2\mathbb{E}[\mathfrak{P}_{0,0}-\mathfrak{P}_{S,0}]}{SK\eta}+\frac{3L\sigma^2\eta}{m}+\frac{72\eta\sigma^2}{(1-\lambda)^2m}+\frac{72(1+\lambda K-\lambda)\eta\sigma^2}{(1-\lambda)^3Km}. \quad (6)$$

Several important remarks for Theorem 1 are in order. First, the convergence metric in Theorem 1 is $\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\sum_{i=1}^{m}\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2$, where the first term is the global gradient magnitude for the non-convex objectives and the second term is the average consensus error across all local parameters in the network system. Although depending on the Lipschitz constant L, this metric does not lose generality because we can change the metric to be problem instance-independent by removing L^2 from the second term, which is due to $\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{1}{m}\sum_{i=1}^{m}\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2 \le \frac{1}{\min\{1,L^2\}}(\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\sum_{i=1}^{m}\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2)$. With the metric in Theorem 1 going to zero, we have that all local parameters will asymptotically be equal and reach a first-order stationary point of the global objective function $f(\cdot)$. Moreover, Theorem 1 provides a finite-time convergence rate guarantee for our NET-FLEET algorithm.

Second, for the convergence error on the right-hand-side (RHS) of Eq. (6), with simple derivations, the first term can be bounded as:

$$\frac{2}{SK\eta}\mathbb{E}[\mathfrak{P}_{0,0}-\mathfrak{P}_{S,0}] \le \frac{2}{SK\eta}\left[f(\mathbf{x}^0)+\frac{C_1\eta^2}{m^2K}\sum_{i=1}^{m}\|\mathbf{y}_{0,0}^{(i)}-\bar{\mathbf{y}}_{0,0}\|^2-f(\mathbf{x}^*)\right],$$

which is dependent on the initialization. The third and fourth terms are affected by the network topology: a sparser network (i.e., λ is closer to 1) will have larger values in these two terms.

Third, the range of step-size η is also dependent on the network topology. A sparse network leads to a smaller step-size. In the following, we show that by properly selecting the parameters, our proposed NET-FLEET can achieve a linear speedup for convergence:

Corollary 2 (Linear Speedup). *Under Assumption 1, by setting $K = S^{1/3}/m$ and $\eta = O(\sqrt{m/SK})$, if the numbers of global and local communication rounds are sufficiently large such that $SK \ge m^{1/3}$, then NET-FLEET has the following convergence rate:*

$$\frac{1}{SK}\sum_{s=0}^{S-1}\sum_{k=0}^{K-1}\mathbb{E}\big[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\sum_{i=1}^{m}\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2\big] = O\Big(\frac{\mathbb{E}[\mathfrak{P}_{0,0}-\mathfrak{P}_{S,0}]}{\sqrt{SKm}}+\frac{\sigma^2}{\sqrt{SKm}}\Big), \quad (7)$$

which implies a linear speed up for convergence.

It is worth noting that our algorithm achieves the same $K = S^{1/3}/m$ number of local updates as in [5] *without* any bounded gradient assumption.

4.1 Proof Sketch of Theorem 1

Due to space limitation, we provide a proof sketch of Theorem 1 and relegate the proof details to our online technical report[42]. For better readability, in this section, we organize the proof of Theorem 1 into several key lemmas. Our first step to prove Theorem 1 is to show the descent property of our NET-FLEET algorithm, which is stated in the following lemma:

Lemma 1. *Under Assumption 1, the following inequality holds for any outloop s in Algorithm 1:*

$$\begin{aligned}&\mathbb{E}[f(\bar{\mathbf{x}}_{s,K})-f(\bar{\mathbf{x}}_{s,0})]\\ &\le -\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2]-\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}\Big[\|\frac{1}{m}\sum_{i=1}^{m}\nabla f_i(\mathbf{x}_{s,k}^{(i)})\|^2\Big]\\ &+\frac{L\eta^2}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2]+\frac{L^2\eta}{2m}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}\Big[\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2\Big]. \quad (8)\end{aligned}$$

Although Lemma 1 appears to be similar to conventional analysis, its proof is highly non-trivial. In (8), we focus on the descending upper bound for each two outloop local model parameters, between which have K inner loop SGD updates, while the conventional analysis on gradient tracking method studies on two successive local model parameters with only one round of SGD update. More Specifically, we note that the RHS of (8) contains the consensus error of local model parameters $\sum_{k=0}^{K-1}\sum_{i=1}^{m}\mathbb{E}[\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2]$, which sums across not only the worker number m but also inner loop iterations K. In decentralized FL, we hope that the algorithm works with *large* m and *large* K to support large-scale systems and reduce communication costs, respectively, which in turn leads to a large consensus error. This large consensus error makes the algorithm harder to converge compared to decentralized learning algorithms. Therefore, in what follows, we will establish the error bound for the consensus error in Lemma 2. Unlike the conventional gradient-tracking analysis that simply focuses on one iteration (cf., e.g., Lemma 3 in [21]), our analysis studies the consensus error across *multiple inner loop iterations*, which thus is novel and more challenging.

Lemma 2. *Under Assumption 1, we have the following bounds for the consensus error in Algorithm 1:*

$$\begin{aligned}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\|\mathbf{x}_{s,k}^{(i)}-\bar{\mathbf{x}}_{s,k}\|^2 &\le (1+\lambda(K-1))\sum_{i=1}^{m}\|\mathbf{x}_{s,0}^{(i)}-\bar{\mathbf{x}}_{s,0}\|^2\\ &+\frac{\eta^2K^2}{1-\lambda}\sum_{k=0}^{K-1}\sum_{i=1}^{m}\|\mathbf{y}_{s,k}^{(i)}-\bar{\mathbf{y}}_{s,k}\|^2, \quad (9)\\ \sum_{k=0}^{K-1}\sum_{i=1}^{m}\|\mathbf{y}_{s,k}^{(i)}-\bar{\mathbf{y}}_{s,k}\|^2 &\le (1+\lambda(K-1))\sum_{i=1}^{m}\|\mathbf{y}_{s,0}^{(i)}-\bar{\mathbf{y}}_{s,0}\|^2\\ &+\frac{24KL^2}{1-\lambda}\sum_{i=1}^{m}\|\mathbf{x}_{s,0}^{(i)}-\bar{\mathbf{x}}_{s,0}\|^2+\frac{6mK\sigma^2}{1-\lambda}+\frac{12\eta^2K^2L^2}{1-\lambda}\\ &\times\sum_{k=0}^{K-1}\sum_{i=1}^{m}\|\mathbf{y}_{s,k}^{(i)}-\bar{\mathbf{y}}_{s,k}\|^2+\frac{12m\eta^2K^2L^2}{1-\lambda}\sum_{t=0}^{k-1}\|\bar{\mathbf{y}}_{s,t}\|^2. \quad (10)\end{aligned}$$

From (9)-(10), we can see that the consensus errors on $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ are coupled. Moreover, the error bounds are accumulated as inner loop rounds K and worker number m increase. This observation

suggests that we need to judiciously design a potential function $\mathfrak{P}_{s,k}$, so that the linear speedup for convergence remains achievable.

By combining Lemmas 1 and 2 and after some algebraic simplifications, we can conclude that:

$$\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2] \le \mathbb{E}[\mathfrak{P}_{s,0}-\mathfrak{P}_{s,K}] - \frac{\eta C_{\nabla \mathbf{f}}}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\nabla \mathbf{f}}_{s,k}\|^2]$$
$$-\frac{C_{\mathbf{x}}}{m^2K}\mathbb{E}[\sum_{i=1}^{m}\|\mathbf{x}_{s,0}^{(i)}-\bar{\mathbf{x}}_{s,0}\|^2] - \frac{C_{\mathbf{y}}C_1\eta^2}{m^2K}\mathbb{E}[\sum_{i=1}^{m}\|\mathbf{y}_{s,0}^{(i)}-\bar{\mathbf{y}}_{s,0}\|^2]$$
$$+(\frac{1}{2}+\frac{12C_1LK\eta^2}{(1-\lambda)m}+\frac{72LK^2\eta^2}{(1-\lambda)^2m})\frac{LK\eta^2\sigma^2}{m}+\frac{36K\eta^2\sigma^2}{(1-\lambda)^2m}+\frac{6C_1\eta^2\sigma^2}{(1-\lambda)mK},$$

where $C_{\nabla \mathbf{f}}$, $C_{\mathbf{x}}$ and $C_{\mathbf{y}}$ are three constants dependent on the step-size η (see detailed definitions in the supplementary material). Then, by properly choosing the step-size, we can ensure that $C_{\nabla \mathbf{f}}$, $C_{\mathbf{x}}$ and $C_{\mathbf{y}}$ are positive, and so terms associated with them can be dropped. Finally, by telescoping the above inequality, we arrive at the desired result as stated in Theorem 1 and the proof is complete.

5 EXPERIMENTAL EVALUATION

In this section, we evaluate our NET-FLEET algorithm on MNIST [15] and CIFAR-10 [14] datasets. Our experiments are conducted with four NVIDIA Tesla V100 GPUs.

1) Datasets and Learning Models: *1-a) MNIST with Convolutional Neural Networks (CNN):* We train a CNN classifier on the MNIST [15] dataset. The adopted CNN model has two convolutional layers (size $3 \times 3 \times 16$), each of which is followed by a max-pooling layer with size 2×2 and then a fully connected layer. The ReLU activation is used for the two convolutional layers and the "softmax" activation is used at the output layer. *1-b) CIFAR-10 with Residual Neural Networks (ResNet):* We experiment with classification problems over the CIFAR-10 [14] dataset with the ResNet18 [8] model. *1-c) Dataset Partition:* For independent and identically distributed (i.i.d.) data partition, all workers can access the same global dataset; in the case of non-i.i.d. heterogeneous data partition, we use the same data partition strategy as in [34] that each worker can access data with at most two labels. Specifally, for the non-i.i.d. setting, we first sort the training data by label, then divide all the training data into 250 shards with 200 data samples, and randomly assign two shards to each client.

2) Network System Model: We consider a decentralized network system with 50 workers. The network topology $\mathcal{G}$ is generated by the Erdös-Rènyi random graph. Without specification, we set the edge connectivity probability $p_c = 0.5$ for the random graph generation. The consensus matrix is chosen as $\mathbf{W} = \mathbf{I} - \frac{2\mathbf{L}}{3\lambda_{\max}(\mathbf{L})}$, where $\mathbf{L}$ is the Laplacian matrix of $\mathcal{G}$ and $\lambda_{\max}(\mathbf{L})$ denotes the largest eigenvalue of $\mathbf{L}$.

3) Baselines and Parameter Settings: We compare our NET-FLEET algorithm with the state-of-the-art LD-SGD [17], GT-SGD [32] and DSGD [18] on decentralized network systems. The number of local update rounds K is set to 10 for NET-FLEET and LD-SGD. For MNIST on CNN, we choose the initial step-size as 0.01 and reduce the step-size to by half for every 100 iterations. The local batch size is fixed at 32. For CIFAR-10 on ResNet, we choose the step-size as 0.001. The local batch size is fixed at 128 for CIFAR-10 training.

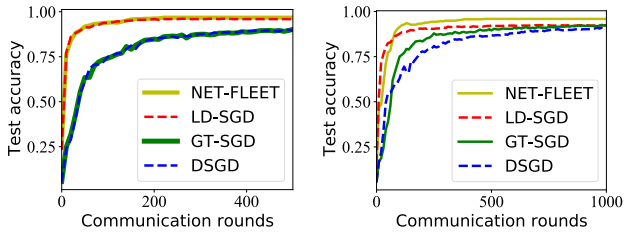
4) Performance Comparisons:

We compare the test accuracy with respect to the numbers of communication rounds and training samples. To better visualize the results, the test accuracies are smoothed by averaging the values in a window of size 10. Fig. 1 illustrates the results of decentralized algorithms of CNN on MNIST. In Fig. 1 (a), we can see that NET-FLEET and LD-SGD have similar performances under i.i.d. data partition and significantly outperform DSGD and GT-SGD with the same communication rounds. Fig. 1 (b) shows that under heterogeneous data, NET-FLEET outperforms the other algorithms: with 1000 communication rounds, the testing accuracy of NET-FLEET is 5% higher than that of LD-SGD and 8% higher than those of DSGD and GT-SGD.

Fig. 2 illustrates the results of NET-FLEET for ResNet model on CIFAR-10 dataset. In Fig. 2(a), we can see that NET-FLEET and LD-SGD have similar performances under i.i.d. data partition and significantly outperform DSGD and GT-SGD with the same number of communication rounds. Fig. 2(b) shows that under heterogeneous data partition, NET-FLEET outperforms the other algorithms: with 500 communication rounds, the NET-FLEET achieves higher test accuracy than that of LD-SGD, DSGD and GT-SGD.

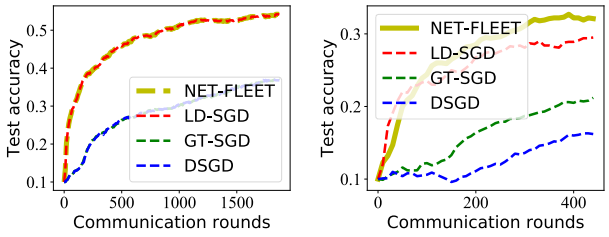
5) Impact of the Local Update Rounds:

A key feature in FL algorithms is that the workers are allowed to perform multiple local parameter updates. In this experiment, we examine the impact of different number of local update rounds on



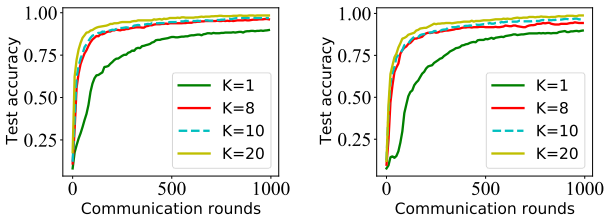
(a) The i.i.d. case. (b) The non-i.i.d. case.

Figure 1: Test accuracy of CNN on MNIST by different decentralized learning algorithms.



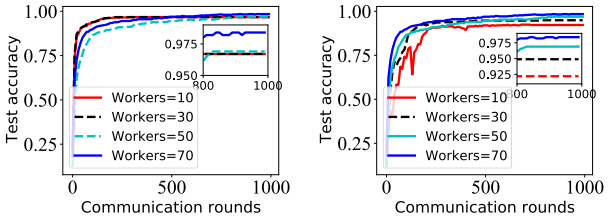
(a) The i.i.d. case. (b) The non-i.i.d. case.

Figure 2: Test accuracy of ResNet on CIFAR-10 decentralized learning algorithms.



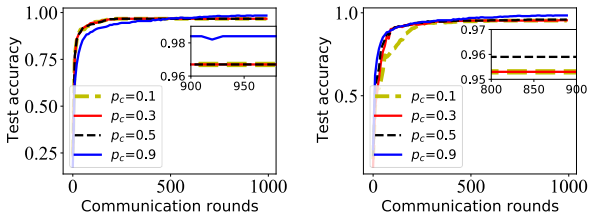
(a) The i.i.d. case. (b) The non-i.i.d. case.

Figure 3: Test accuracy of CNN on MNIST with different local update rounds.



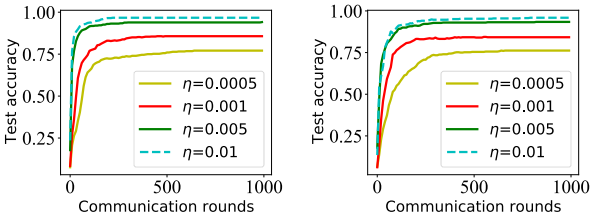
(a) The i.i.d. case. (b) The non-i.i.d. case.

Figure 4: Test accuracy of CNN on MNIST with different number of workers.



(a) The i.i.d. case. (b) The non-i.i.d. case.

Figure 5: Test accuracy of CNN on MNIST with different edge connection probability p_c.



(a) The i.i.d. case. (b) The non-i.i.d. case.

Figure 6: Test accuracy of CNN on MNIST with different local step-size.

the training performance. We run NET-FLEET to solve classification problems with the CNN model over the MNIST [15] dataset. We fix the step-size at 0.01, edge connectivity p_c at 0.5, local batch size at 32, and worker number at 50. We choose the number of local update rounds K from the discrete set $\{1, 8, 10, 20\}$. Fig. 3 shows the performance of NET-FLEET with different number of local update rounds K. As shown in Fig. 3, the test accuracy increases as K increases under both the i.i.d. and heterogeneous data settings: with communication rounds being fixed at 1000, NET-FLEET with $K = 1$ has accuracy less than 80%. In contrast, with $K = 8$, $K = 10$ and $K = 20$, NET-FLEET achieves more than 95% testing accuracy.

6) Impact of the Number of Workers: We conduct the following experiments with different number of workers. In this experiment, we choose the number of workers from the discrete set $\{10, 30, 50, 70\}$ and fix the step-size at 0.01, local update rounds at 10, edge connectivity p_c at 0.5, and local batch size at 32. As shown in Fig. 4, convergence results with different number of workers have similar performances in i.i.d case. NET-FLEET achieves 95% accuracy in the i.i.d case. In the non-i.i.d heterogeneous case, we can see that as the number of workers decreases, the convergence rate decreases. NET-FLEET obtains an accuracy around 96% with 10 workers and achieves more than 97.5% test accuracy with 70 workers in i.i.d case. In heterogeneous data case, NET-FLEET's accuracy is approximately 92.5% with 10 workers and achieves a test accuracy more than 97.5% with 70 workers.

7) Impact of the Edge Connectivity Probability:

For the decentralized network system, the network graph $\mathcal{G}$ is generated by the Erdös-Rènyi random graph with edge connection probability p_c. In the first experiment, we examine the impact of different p_c-values on the training performance with the CNN model over the MNIST dataset. We choose the p_c-value from the discrete set $\{0.1, 0.3, 0.5, 0.9\}$ and fix the number of workers at 50, local update rounds at 10, step-size at 0.01, and local batch size at 32. Fig. 5 shows that the convergence result with different edge connectivity p_c-values have similar performances in the i.i.d case. The experiments achieve a 96% accuracy in the i.i.d case. In the heterogeneous data case, we can see that as p_c increases, the test accuracy increases slightly, which shows that the learning performance of NET-FLEET is insensitive to the p_c-value.

8) Impact of the Step-size: In this experiment, we choose the step-size from the discrete set $\{0.0005, 0.001, 0.005, 0.01\}$ and fix worker number at 50, local update rounds at 10, edge connectivity p_c at 0.5, and local batch size at 32, global batch size at 512. As shown in Fig. 6, larger local step-sizes lead to faster convergence rates in both i.i.d and non-i.i.d cases. NET-FLEET achieves accuracy less than 75% with a step-size 0.0005, and obtains more than 95% test accuracy with a step-size 0.01.

6 CONCLUSION

In this paper, we studied fully decentralized federated learning with data heterogeneity. A novel federated learning algorithm named NET-FLEET was proposed for fully decentralized network systems. Our NET-FLEET algorithm allows the workers to keep the local data and run multiple local update steps during the training, thus maintaining local data privacy and reducing the communication costs. We showed that with properly selected parameters, our algorithm achieves the state-of-the-art linear speedup for convergence,

i.e., an $O(1/\sqrt{mKS})$ convergence rate, where m is the number of workers, and S and K are the numbers of communication and local update rounds, respectively. Extensive numerical studies verified the theoretical performance results of our proposed algorithm.

ACKNOWLEDGMENTS

This work has been supported in part by NSF grants CAREER CNS-2110259, CNS-2112471, CNS-2102233, CCF-2110252, CCF 1934884, and SES 1952007.

REFERENCES

[1] 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* (2018), 59–67.

[2] Cao, X., Fang, M., Liu, J., and Gong, N. Z. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *ISOC Network and Distributed System Security Symposium (NDSS)* (2021).

[3] Dekel, O., Gilad-Bachrach, R., Shamir, O., and Xiao, L. Optimal distributed online prediction using mini-batches. *The Journal of Machine Learning Research 13* (2012), 165–202.

[4] Fang, C., Li, C. J., Lin, Z., and Zhang, T. Spider: near-optimal non-convex optimization via stochastic path integrated differential estimator. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (2018), pp. 687–697.

[5] Gao, H., and Huang, H. Periodic stochastic gradient descent with momentum for decentralized training. *arXiv preprint arXiv:2008.10435* (2020).

[6] Ghadimi, S., and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization 23*, 4 (2013), 2341–2368.

[7] Haddadpour, F., and Mahdavi, M. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425* (2019).

[8] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[9] Johnson, R., and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems 26* (2013), 315–323.

[10] 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. *Foundations and Trends in Machine Learning 14*, 1–2 (2021), 1–210.

[11] Kang, J., Xiong, Z., Niyato, D., Zou, Y., Zhang, Y., and Guizani, M. Reliable federated learning for mobile networks. *IEEE Wireless Communications 27*, 2 (2020), 72–80.

[12] 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* (2020), PMLR, pp. 5132–5143.

[13] Khanduri, P., Sharma, P., Yang, H., Hong, M., Liu, J., Rajawat, K., and Varshney, P. K. Achieving optimal sample and communication complexities for non-iid federated learning. In *ICML Workshop on Federated Learning for User Privacy and Data Confidentiality* (2021).

[14] Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images.

[15] LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *Available: http://yann. lecun. com/exdb/mnist* (1998).

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

[17] Li, X., Yang, W., Wang, S., and Zhang, Z. Communication efficient decentralized training with multiple local updates. *arXiv preprint arXiv:1910.09126* (2019).

[18] 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* (2017), vol. 30.

[19] Liang, X., Shen, S., Liu, J., Pan, Z., Chen, E., and Cheng, Y. Variance reduced local SGD with lower communication complexity. *arXiv preprint arXiv:1912.12844* (2019).

[20] Lin, T., Stich, S. U., Patel, K. K., and Jaggi, M. Don't use large mini-batches, use local sgd. *arXiv preprint arXiv:1808.07217* (2018).

[21] Lu, S., Zhang, X., Sun, H., and Hong, M. Gnsd: A gradient-tracking based nonconvex stochastic algorithm for decentralized optimization. In *2019 IEEE Data Science Workshop (DSW)* (2019), IEEE, pp. 315–321.

[22] Lu, S., Zhang, Y., and Wang, Y. Decentralized federated learning for electronic health records. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)* (2020), IEEE, pp. 1–5.

[23] 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* (2017), PMLR, pp. 1273–1282.

[24] Nedic, A., and Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control 54*, 1 (2009), 48–61.

[25] Pu, S., and Nedić, A. Distributed stochastic gradient tracking methods. *Mathematical Programming* (2020), 1–49.

[26] Qu, G., and Li, N. Harnessing smoothness to accelerate distributed optimization. *IEEE Transactions on Control of Network Systems 5*, 3 (2017), 1245–1260.

[27] Sahu, A. K., Li, T., Sanjabi, M., Zaheer, M., Talwalkar, A., and Smith, V. On the convergence of federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127 3* (2018).

[28] Stich, S. U. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767* (2018).

[29] Stich, S. U., and Karimireddy, S. P. The error-feedback framework: Better rates for sgd with delayed gradients and compressed updates. *Journal of Machine Learning Research 21* (2020), 1–36.

[30] Wang, J., and Joshi, G. Cooperative sgd: A unified framework for the design and analysis of local-update sgd algorithms. *Journal of Machine Learning Research 22* (2021).

[31] Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications 37*, 6 (2019), 1205–1221.

[32] Xin, R., Khan, U. A., and Kar, S. An improved convergence analysis for decentralized online stochastic non-convex optimization. *IEEE Transactions on Signal Processing 69* (2021), 1842–1858.

[33] Xu, J., Glicksberg, B. S., Su, C., Walker, P., Bian, J., and Wang, F. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research* (2020), 1–19.

[34] Yang, H., Fang, M., and Liu, J. Achieving linear speedup with partial worker participation in non-i.i.d. federated learning. In *International Conference on Learning Representations* (2021).

[35] Yang, H., Liu, J., and Bentley, E. S. Cfedavg: achieving efficient communication and fast convergence in non-iid federated learning. In *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)* (2021), IEEE, pp. 1–8.

[36] Yang, H., Zhang, X., Khanduri, P., and Liu, J. Anarchic federated learning. In *International Conference on Machine Learning* (2022), PMLR, pp. 25331–25363.

[37] Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST) 10*, 2 (2019), 1–19.

[38] Yu, H., Jin, R., and Yang, S. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. In *International Conference on Machine Learning* (2019), PMLR, pp. 7184–7193.

[39] Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 5693–5700.

[40] Yuan, K., Ling, Q., and Yin, W. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization 26*, 3 (2016), 1835–1854.

[41] Zeng, J., and Yin, W. On nonconvex decentralized gradient descent. *IEEE Transactions on Signal Processing 66*, 11 (2018), 2834–2848.

[42] Zhang, X., Fang, M., Liu, Z., Yang, H., Liu, J., and Zhu, Z. Net-fleet: Achieving linear convergence speedup for fully decentralized federated learning with heterogeneous data. https://kevinliu-osu.github.io/publications/FLEET_TR.pdf.

[43] Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582* (2018).

A PROOF OF MAIN RESULTS

For notation convenience, we define the following variables: $\widetilde{\mathbf{W}} = \mathbf{W} \otimes \mathbf{I}_m$, $\mathbf{g}_{s,k}^{(i)} = \nabla f_i(\mathbf{x}_{s,k}^{(i)}; \boldsymbol{\zeta}_{s,k}^{(i)})$, $\nabla \mathbf{f}_{s,k}^{(i)} = \nabla f_i(\mathbf{x}_{s,k}^{(i)})$, and $\mathbf{a}_{s,k} = [\mathbf{a}_{s,k}^{(i)\top}, \cdots, \mathbf{a}_{s,k}^{(i)\top}]^\top$ and $\bar{\mathbf{a}}_{s,k} = \frac{1}{m}\sum_{i=1}^m \mathbf{a}_{s,k}^{(i)}$, for $\mathbf{a} \in \{\mathbf{x}, \mathbf{y}, \mathbf{g}, \nabla \mathbf{f}\}$. Here $\bar{\mathbf{y}}_{s,k} = \bar{\mathbf{g}}_{s,k}$ because of $\bar{\mathbf{y}}_{s,0} = \bar{\mathbf{g}}_{s,0}$. Also, we define matrix $\mathbf{Q} \triangleq \mathbf{I} - (\frac{1}{m}\mathbf{1}\mathbf{1}^\top) \otimes \mathbf{I}$, so it holds that $\mathbf{Q}\mathbf{a}_{s,k} = \mathbf{a}_{s,k} - \mathbf{1} \otimes \bar{\mathbf{a}}_{s,k}$.

A.1 Proof of Lemma 1

Proof. From the L-smoothness of f and $\bar{\mathbf{x}}_{s,k+1} = \bar{\mathbf{x}}_{s,k} - \eta\bar{\mathbf{y}}_{s,k} = \bar{\mathbf{x}}_k - \eta\bar{\mathbf{g}}_{s,k}$, we have

$$f(\bar{\mathbf{x}}_{s,k+1}) \le f(\bar{\mathbf{x}}_{s,k}) - \langle \nabla f(\bar{\mathbf{x}}_{s,k}), \bar{\mathbf{x}}_{s,k+1} - \bar{\mathbf{x}}_{s,k}\rangle + \frac{L}{2}\|\bar{\mathbf{x}}_{s,k+1} - \bar{\mathbf{x}}_{s,k}\|^2 = f(\bar{\mathbf{x}}_{s,k}) - \eta\langle \nabla f(\bar{\mathbf{x}}_{s,k}), \bar{\mathbf{g}}_{s,k}\rangle + \frac{L\eta^2}{2}\|\bar{\mathbf{g}}_{s,k}\|^2. \tag{11}$$

Since $\mathbb{E}[\mathbf{g}_{s,k}^{(i)}|\mathcal{F}_{s,k}] = \nabla f_{s,k}^{(i)}$, we have

$$\mathbb{E}[f(\bar{\mathbf{x}}_{s,k+1})|\mathcal{F}_{s,k}] \le f(\bar{\mathbf{x}}_{s,k}) - \eta\langle \nabla f(\bar{\mathbf{x}}_{s,k}), \overline{\nabla \mathbf{f}}_{s,k}\rangle + \frac{L\eta^2}{2}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2|\mathcal{F}_{s,k}] = f(\bar{\mathbf{x}}_{s,k}) - \frac{\eta}{2}\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2 - \frac{\eta}{2}\|\overline{\nabla \mathbf{f}}_{s,k}\|^2 + \frac{\eta}{2}\|\nabla f(\bar{\mathbf{x}}_{s,k}) - \overline{\nabla \mathbf{f}}_{s,k}\|^2 + \frac{L\eta^2}{2}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2|\mathcal{F}_{s,k}] \le f(\bar{\mathbf{x}}_{s,k}) - \frac{\eta}{2}\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2 - \frac{\eta}{2}\|\overline{\nabla \mathbf{f}}_{s,k}\|^2 + \frac{L^2\eta}{2m}\|\mathbf{Q}\mathbf{x}_{s,k}\|^2 + \frac{L\eta^2}{2}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2|\mathcal{F}_{s,k}]. \tag{12}$$

Taking full expectation on the above inequality and telescoping from $k = 0$ to $K-1$ yields:

$$\mathbb{E}[f(\bar{\mathbf{x}}_{s,K}) - f(\bar{\mathbf{x}}_{s,0})] \le -\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2] - \frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\nabla \mathbf{f}}_{s,k}\|^2] + \frac{L\eta^2}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2] + \frac{L^2\eta}{2m}\sum_{k=0}^{K-1}\mathbb{E}[\|\mathbf{Q}\mathbf{x}_{s,k}\|^2]. \tag{13}$$

□

A.2 Proof of Lemma 2

Proof. First, for any $\mathbf{x}_t$ and $\lambda = \max\{|\lambda_2|, |\lambda_m|\}$, we have:

$$\|\widetilde{\mathbf{W}}\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 = \|\widetilde{\mathbf{W}}(\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t)\|^2 \le \lambda^2\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2. \tag{14}$$

Note that $\mathbf{x}_{s,k} = \widetilde{\mathbf{W}}\mathbf{x}_{s,0} - \eta\sum_{t=0}^{k-1}\mathbf{y}_{s,t}$ and $\bar{\mathbf{x}}_{s,k} = \bar{\mathbf{x}}_{s,0} - \eta\sum_{t=0}^{k-1}\bar{\mathbf{y}}_{s,t}$. Thus, we have

$$\|\mathbf{Q}\mathbf{x}_{s,k}\|^2 = \|\widetilde{\mathbf{W}}\mathbf{x}_{s,0} - \eta\sum_{t=0}^{k-1}\mathbf{y}_{s,t} - \mathbf{1}\otimes(\bar{\mathbf{x}}_{s,0} - \eta\sum_{t=0}^{k-1}\bar{\mathbf{y}}_{s,t})\|^2 \overset{(a)}{\le} (1+c_1)\|\widetilde{\mathbf{W}}\mathbf{x}_{s,0} - \mathbf{1}\otimes\bar{\mathbf{x}}_{s,0}\|^2 + (1+\frac{1}{c_1})\eta^2\|\sum_{t=0}^{k-1}\mathbf{y}_{s,t} - \mathbf{1}\otimes\bar{\mathbf{y}}_{s,t}\|^2 \overset{(b)}{\le} \lambda\|\mathbf{x}_{s,0} - \mathbf{1}\otimes\bar{\mathbf{x}}_{s,0}\|^2 + \frac{\eta^2}{1-\lambda}\|\sum_{t=0}^{k-1}\mathbf{y}_{s,t} - \mathbf{1}\otimes\bar{\mathbf{y}}_{s,t}\|^2 \overset{(c)}{\le} \lambda\|\mathbf{Q}\mathbf{x}_{s,0}\|^2 + \frac{\eta^2 k}{1-\lambda}\sum_{t=0}^{k-1}\|\mathbf{Q}\mathbf{y}_{s,t}\|^2, \tag{15}$$

where (a) follows from $\|\mathbf{x}+\mathbf{y}\|^2 \le (1+c)\|\mathbf{x}\|^2 + (1+1/c)\|\mathbf{y}\|^2$ for any $c > 0$, (b) follows from (14) with $c_1 = 1/\lambda - 1$, and (c) follows from the Jensen's inequality.

Since $\mathbf{y}_{s,k} = \mathbf{y}_{s,k-1} + \mathbf{g}_{s,k} - \mathbf{g}_{s,k-1} = \widetilde{\mathbf{W}}\mathbf{y}_{s,0} + \mathbf{g}_{s,k} - \mathbf{g}_{s,0}$ and $\bar{\mathbf{y}}_{s,k} = \bar{\mathbf{y}}_{s,0} + \bar{\mathbf{g}}_{s,k} - \bar{\mathbf{g}}_{s,0}$, it follows that

$$\|\mathbf{Q}\mathbf{y}_{s,k}\|^2 = \|\widetilde{\mathbf{W}}\mathbf{y}_{s,0} + \mathbf{g}_{s,k} - \mathbf{g}_{s,0} - \mathbf{1}\otimes(\bar{\mathbf{y}}_{s,0} + \bar{\mathbf{g}}_{s,k} - \bar{\mathbf{g}}_{s,0})\|^2 \le \lambda\|\mathbf{Q}\mathbf{y}_{s,0}\|^2 + \frac{1}{1-\lambda}\|\mathbf{g}_{s,k} - \mathbf{g}_{s,0}\|^2 \le \lambda\|\mathbf{Q}\mathbf{y}_{s,0}\|^2 + \frac{3}{1-\lambda}(2m\sigma^2 + L^2\|\mathbf{x}_{s,k} - \mathbf{x}_{s,0}\|^2), \tag{16}$$

Note that the term $\|\mathbf{x}_{s,k} - \mathbf{x}_{s,0}\|^2$ can be bounded as:

$$\|\mathbf{x}_{s,k} - \mathbf{x}_{s,0}\|^2 = \|\widetilde{\mathbf{W}}\mathbf{x}_{s,0} - \eta\sum_{t=0}^{k-1}\mathbf{y}_{s,t} - \mathbf{x}_{s,0}\|^2 = 8\|\mathbf{x}_{s,0} - \mathbf{1}\otimes\bar{\mathbf{x}}_{s,0}\|^2 + 2\eta^2 k\sum_{t=0}^{k-1}\|\mathbf{y}_{s,t}\|^2 \overset{(a)}{\le} 8\|\mathbf{x}_{s,0} - \mathbf{1}\otimes\bar{\mathbf{x}}_{s,0}\|^2 + 2\eta^2 k\sum_{t=0}^{k-1}(2\|\mathbf{1}\otimes\bar{\mathbf{y}}_{s,t}\|^2 + 2\|\mathbf{y}_{s,t} - \mathbf{1}\otimes\bar{\mathbf{y}}_{s,t}\|^2) \le 8\|\mathbf{Q}\mathbf{x}_{s,0}\|^2 + 4\eta^2 k\sum_{t=0}^{k-1}\|\mathbf{Q}\mathbf{y}_{s,t}\|^2 + 4\eta^2 mk\sum_{t=0}^{k-1}\|\bar{\mathbf{y}}_{s,t}\|^2, \tag{17}$$

where (a) is due to the fact that $\|\widetilde{\mathbf{W}} - \mathbf{I}\| \le 2$.

Thus, by plugging (17) into (16), we have

$$\|\mathbf{Q}\mathbf{y}_{s,k}\|^2 \le \lambda\|\mathbf{Q}\mathbf{y}_{s,0}\|^2 + \frac{6m\sigma^2}{1-\lambda} + \frac{24L^2}{1-\lambda}\|\mathbf{Q}\mathbf{x}_{s,0}\|^2 + \frac{12k\eta^2L^2}{1-\lambda}\sum_{t=0}^{k-1}\|\mathbf{Q}\mathbf{y}_{s,t}\|^2 + \frac{12mk\eta^2L^2}{1-\lambda}\sum_{t=0}^{k-1}\|\bar{\mathbf{y}}_{s,t}\|^2. \tag{18}$$

□

A.3 Proof of Theorem 1

Proof. By combining the results from Lemma 1 and Lemma 2, we have

$$\mathbb{E}[f(\bar{\mathbf{x}}_{s,K}) - f(\bar{\mathbf{x}}_{s,0})] \le -\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2] - \frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\nabla \mathbf{f}}_{s,k}\|^2] - \frac{L^2\eta}{2m}\sum_{k=0}^{K-1}\mathbb{E}[\|\mathbf{Q}\mathbf{x}_{s,k}\|^2] + \frac{L\eta^2}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2] + \frac{L^2\eta}{m}(1+\lambda(K-1)) \times \mathbb{E}\|\mathbf{Q}\mathbf{x}_{s,0}\|^2 + \frac{\eta^3L^2K^2}{m(1-\lambda)}\sum_{k=0}^{K-1}\mathbb{E}\|\mathbf{Q}\mathbf{y}_{s,k}\|^2. \tag{19}$$

Also, from Lemma 2, for some constant C_1 (to be determined later), it follows that

$$(\|\mathbf{Q}\mathbf{x}_{s,K}\|^2 + C_1\eta^2\|\mathbf{Q}\mathbf{y}_{s,K}\|^2) - (\|\mathbf{Q}\mathbf{x}_{s,0}\|^2 + C_1\eta^2\|\mathbf{Q}\mathbf{y}_{s,0}\|^2) \le -(1-\lambda-\frac{24C_1L^2\eta^2}{1-\lambda})\|\mathbf{Q}\mathbf{x}_{s,0}\|^2 - (1-\lambda)C_1\eta^2\|\mathbf{Q}\mathbf{y}_{s,0}\|^2 + \frac{\eta^2K + 12C_1KL^2\eta^4}{1-\lambda}\sum_{k=0}^{K-1}\|\mathbf{Q}\mathbf{y}_{s,k}\|^2 + \frac{12mC_1L^2K^2\eta^4}{1-\lambda} \times \sum_{k=0}^{K-1}\|\bar{\mathbf{y}}_{s,k}\|^2 + \frac{6mC_1\eta^2\sigma^2}{1-\lambda}. \tag{20}$$

Thus, combining (19) and (20), we have

$$\begin{aligned}
&\mathbb{E}[f(\bar{\mathbf{x}}_{s,K})-f(\bar{\mathbf{x}}_{s,0})+\frac{1}{m^2K}(\|\mathbf{Q}\mathbf{x}_{s,K}\|^2+C_1\eta^2\|\mathbf{Q}\mathbf{x}_{s,K}\|^2)\\
&-\frac{1}{m^2K}(\|\mathbf{Q}\mathbf{x}_{s,0}\|^2+C_1\eta^2\|\mathbf{Q}\mathbf{x}_{s,0}\|^2)]\\
\le&-\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2]-\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\nabla\mathbf{f}}_{s,k}\|^2]-\frac{L^2\eta}{2m}\sum_{k=0}^{K-1}\mathbb{E}[\|\mathbf{Q}\mathbf{x}_{s,k}\|^2]\\
&+\frac{6C_1\eta^2\sigma^2}{(1-\lambda)mK}+(\frac{L\eta^2}{2}+\frac{12C_1L^2K\eta^4}{(1-\lambda)m})\sum_{k=0}^{K-1}\mathbb{E}[\|\bar{\mathbf{g}}_{s,k}\|^2]+\frac{3\eta^2}{(1-\lambda)m^2}\\
&\times\sum_{k=0}^{K-1}\mathbb{E}\|\mathbf{Q}\mathbf{y}_{s,k}\|^2-(1-\lambda-\frac{24C_1L^2\eta^2}{1-\lambda}-(1+\lambda(K-1))mKL^2\eta)\\
&\times\frac{1}{m^2K}\mathbb{E}[\|\mathbf{Q}\mathbf{x}_{s,0}\|^2]-(1-\lambda)\frac{C_1\eta^2}{m^2K}\mathbb{E}[\|\mathbf{Q}\mathbf{y}_{s,0}\|^2],
\end{aligned}\tag{21}$$

by setting $\eta\le\min\{1/mL^2K^2,1/\sqrt{12C_1L^2}\}$.

From Lemma 2, with $\eta\le\sqrt{\frac{1-\lambda}{24L^2K^2}}$, it holds that

$$\begin{aligned}
\sum_{k=0}^{K-1}\|\mathbf{Q}\mathbf{y}_{s,k}\|^2\le&\,2(1+\lambda(K-1))\|\mathbf{Q}\mathbf{y}_{s,0}\|^2+\frac{12mK\sigma^2}{1-\lambda}\\
&+\frac{48KL^2}{1-\lambda}\|\mathbf{Q}\mathbf{x}_{s,0}\|^2+\frac{24m\eta^2K^2L^2}{1-\lambda}\sum_{t=0}^{k-1}\|\bar{\mathbf{y}}_{s,t}\|^2.
\end{aligned}\tag{22}$$

By plugging (22) into (21), we have

$$\begin{aligned}
&\mathbb{E}[f(\bar{\mathbf{x}}_{s,K})-f(\bar{\mathbf{x}}_{s,0})+\frac{1}{mK}(\|\mathbf{Q}\mathbf{x}_{s,K}\|^2+C_1\eta^2\|\mathbf{Q}\mathbf{x}_{s,K}\|^2)-\frac{1}{mK}(\|\mathbf{Q}\mathbf{x}_{s,0}\|^2\\
&+C_1\eta^2\|\mathbf{Q}\mathbf{x}_{s,0}\|^2)]\le-\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2]-\frac{L^2\eta}{2m}\sum_{k=0}^{K-1}\mathbb{E}[\|\mathbf{Q}\mathbf{x}_{s,k}\|^2]\\
&-\frac{\eta C_{\nabla\mathbf{f}}}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\nabla\mathbf{f}}_{s,k}\|^2]-\frac{C_{\mathbf{x}}}{m^2K}\mathbb{E}[\|\mathbf{Q}\mathbf{x}_{s,0}\|^2]-\frac{C_{\mathrm{y}}C_1\eta^2}{m^2K}\mathbb{E}[\|\mathbf{Q}\mathbf{y}_{s,0}\|^2]\\
&+(\frac{L\eta^2}{2}+\frac{12C_1L^2K\eta^4}{(1-\lambda)m}+\frac{72L^2K^2\eta^4}{(1-\lambda)^2m})\frac{K\sigma^2}{m}+\frac{36K\eta^2\sigma^2}{(1-\lambda)^2m}+\frac{6C_1\eta^2\sigma^2}{(1-\lambda)mK}.
\end{aligned}\tag{23}$$

where $C_{\nabla\mathbf{f}}\triangleq 1-L\eta-\frac{24C_1L^2K\eta^3}{(1-\lambda)m}-\frac{144L^2K^2\eta^3}{(1-\lambda)^2m}$, $C_{\mathbf{x}}\triangleq 1-\lambda-\frac{24C_1L^2\eta^2}{1-\lambda}-(1+\lambda(K-1))mKL^2\eta-\frac{144L^2\eta^2K^2}{(1-\lambda)^2}$, $C_{\mathrm{y}}\triangleq 1-\lambda-\frac{6(1+\lambda K-\lambda)K}{C_1(1-\lambda)}$. By setting $C_1=\frac{6(1+\lambda K-\lambda)K}{(1-\lambda)^2}$, we have $C_{\mathrm{y}}=0$. By letting $\eta\le\min\{\sqrt{\frac{m(1-\lambda)^3}{144(1+\lambda K-\lambda)LK^2}},\sqrt{\frac{m(1-\lambda)^2}{144LK^2}},1/3L\}$, we have $C_{\nabla\mathbf{f}}\ge 0$. Also, letting $\eta\le\min\{\frac{(1-\lambda)}{3(1+\lambda K-\lambda)mKL^2},\frac{(1-\lambda)^3mK}{144},\frac{(1-\lambda)^2(1+\lambda K-\lambda)m}{144K}\}$, we have $C_{\mathbf{x}}\ge 0$.

With the above parameter setting and the proposed potential function, we have

$$\begin{aligned}
\frac{\eta}{2}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\|\mathbf{Q}\mathbf{x}_{s,k}\|^2]\le&\,\mathbb{E}[\mathfrak{P}_{s,0}-\mathfrak{P}_{s,K}]\\
&+\frac{3LK\sigma^2\eta^2}{2m}+\frac{36K\eta^2\sigma^2}{(1-\lambda)^2m}+\frac{36(1+\lambda K-\lambda)\eta^2\sigma^2}{(1-\lambda)^3m},
\end{aligned}\tag{24}$$

by further setting $\eta\le\min\{\sqrt{\frac{m(1-\lambda)^3}{144(1+\lambda K-\lambda)LK^2}},\sqrt{\frac{m(1-\lambda)^2}{144LK^2}}\}$.

Telescoping (24) for s from 0 to $S-1$ and multiplying the factor $2SK/\eta$ on both sides, we have

$$\begin{aligned}
&\frac{1}{SK}\sum_{s=0}^{S-1}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\|\mathbf{Q}\mathbf{x}_{s,k}\|^2]\le\frac{2\mathbb{E}[\mathfrak{P}_{0,0}-\mathfrak{P}_{S,0}]}{SK\eta}\\
&+\frac{3L\sigma^2\eta}{m}+\frac{72\eta\sigma^2}{(1-\lambda)^2m}+\frac{72(1+\lambda K-\lambda)\eta\sigma^2}{(1-\lambda)^3Km}
\end{aligned}\tag{25}$$

This completes the proof of Theorem 1. □

A.4 Proof of Corollary 2

Proof. Recall from Theorem 1 that the condition on the step-size is

$$\begin{aligned}
\eta\le\min\{&\frac{1}{3L},\underbrace{\frac{1}{mL^2K^2}}_{\triangleq r_1},\underbrace{\frac{(1-\lambda)}{\sqrt{12(1+\lambda K-\lambda)KL^2}}}_{\triangleq r_2},\underbrace{\sqrt{\frac{1-\lambda}{24L^2K^2}}}_{\triangleq r_3},\\
&\underbrace{\sqrt{\frac{m(1-\lambda)^2}{144LK^2}}}_{\triangleq r_5},\underbrace{\frac{(1-\lambda)}{3(1+\lambda K-\lambda)mKL^2}}_{\triangleq r_6},\\
&\underbrace{\frac{(1-\lambda)^3mK}{144}}_{\triangleq r_7},\underbrace{\frac{(1-\lambda)^2(1+\lambda K-\lambda)m}{144K}}_{\triangleq r_8}\}.
\end{aligned}\tag{26}$$

where (a) follows from plugging $C_1=6(1+\lambda K-\lambda)K/(1-\lambda)^2$, and (b) is due to $r_4\le r_5$.

Setting $K=\sqrt[4]{SK/m^3}$ (i.e. $K=S^{1/3}/m$), we have

$$\begin{aligned}
r_1&=\frac{1}{mL^2K^2}=\frac{\sqrt{m}}{L^2\sqrt{SK}}=O(\frac{\sqrt{m}}{\sqrt{SK}}),\\
r_2&=\frac{(1-\lambda)}{\sqrt{12(1+\lambda K-\lambda)KL^2}}\overset{(a)}{\ge}\frac{(1-\lambda)}{\sqrt{12}KL}=\frac{(1-\lambda)m^{3/4}}{\sqrt{12}L(SK)^{1/4}}>O(\frac{\sqrt{m}}{\sqrt{SK}}),\\
r_3&=\sqrt{\frac{1-\lambda}{24L^2K^2}}=\sqrt{\frac{1-\lambda}{24L^2}}\frac{m^{3/4}}{(SK)^{1/4}}>O(\frac{\sqrt{m}}{\sqrt{SK}})\\
r_5&=\sqrt{\frac{m(1-\lambda)^2}{144LK^2}}=\sqrt{\frac{(1-\lambda)^2}{144L}}\frac{m^{7/4}}{(SK)^{1/4}}>O(\frac{\sqrt{m}}{\sqrt{SK}})\\
r_6&=\frac{(1-\lambda)}{3(1+\lambda K-\lambda)mKL^2}\ge\frac{(1-\lambda)}{3mK^2L^2}=\frac{(1-\lambda)\sqrt{m}}{3L^2\sqrt{SK}}=O(\frac{\sqrt{m}}{\sqrt{SK}})\\
r_7&=O((SKm)^{1/4})\overset{(b)}{>}O(\frac{\sqrt{m}}{\sqrt{SK}})\\
r_8&=\frac{(1-\lambda)^2(1+\lambda K-\lambda)m}{144K}\ge\frac{(1-\lambda)^2\lambda m}{144}=O(m)>O(\frac{\sqrt{m}}{\sqrt{SK}}),
\end{aligned}$$

where (a) follows from $K\ge 1+\lambda K-\lambda$ and (b) follows from $SK\ge m^{1/3}$. Then we can set $\eta=O(\sqrt{m}/\sqrt{SK})$ and have the following convergence bound:

$$\begin{aligned}
&\frac{1}{SK}\sum_{s=0}^{S-1}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}_{s,k})\|^2+\frac{L^2}{m}\|\mathbf{Q}\mathbf{x}_{s,k}\|^2]\\
\le&O\Big(\frac{2\mathbb{E}[\mathfrak{P}_{0,0}-\mathfrak{P}_{S,0}]}{\sqrt{SKm}}+\frac{3L\sigma^2}{\sqrt{SKm}}+\frac{72\sigma^2}{(1-\lambda)^2\sqrt{SKm}}+\frac{72\sigma^2}{(1-\lambda)^3\sqrt{SKm}}\Big).
\end{aligned}$$

This completes the proof. □