Leveraging Recursive Methods for Efficient Federated Learning

Jie Liu[†] *jie9@clemson.edu*
Department of Electrical and Computer Engineering
Clemson University, South Carolina, USA

Zuang Wang[†] *zuangw@clemson.edu*
Department of Electrical and Computer Engineering
Clemson University, South Carolina, USA

Yongqiang Wang[*] *yongqiw@clemson.edu*
Department of Electrical and Computer Engineering
Clemson University, South Carolina, USA

Reviewed on OpenReview: *https://openreview.net/forum?id=cVGagKtiVr*

Abstract

Federated learning algorithms perform multiple local updates on clients before communicating with the parameter server to reduce communication overhead and improve overall training efficiency. However, local updates also lead to the "client-drift" problem under non-IID data, which avoids convergence to the exact optimal solution under heterogeneous data distributions. To ensure accurate convergence, existing federated-learning algorithms employ auxiliary variables to locally estimate the global gradient or the drift from the global gradient, which, however, also incurs extra communication and storage overhead. In this paper, we propose a new recursion-based federated-learning architecture that completely eliminates the need for auxiliary variables while ensuring accurate convergence under heterogeneous data distributions. This new federated-learning architecture, called FedRecu, can significantly reduce communication and storage overhead compared with existing federated-learning algorithms with accurate convergence guarantees. More importantly, this novel architecture enables FedRecu to employ much larger stepsizes than existing federated-learning algorithms, thereby leading to much faster convergence. We provide rigorous convergence analysis of FedRecu under both convex and nonconvex loss functions, in both the deterministic gradient case and the stochastic gradient case. In fact, our theoretical analysis shows that FedRecu ensures $o(1/K)$ convergence to an accurate solution under general convex loss functions, which improves upon the existing achievable $O(1/K)$ convergence rate for general convex loss functions. Numerical experiments on benchmark datasets confirm the effectiveness of the proposed algorithm.

1 Introduction

Since its introduction in McMahan et al. (2017), federated learning has been extensively studied and widely applied across a range of domains, including natural language processing (Gupta et al., 2022; Ye et al., 2024; Liu et al., 2021; Lin et al., 2021), wireless networks (Tran et al., 2019; Chen et al., 2021; Niknam et al., 2020; Yang et al., 2019), neural-network training (Yurochkin et al., 2019; Venkatesha et al., 2021; Li et al., 2023; He et al., 2021), and mobile edge networks (Lim et al., 2020b; Luo et al., 2021; Khan et al., 2020; Lim et al.,

[†] These authors contributed equally to this work.
[*] Corresponding author

2020a). Unlike centralized learning which requires aggregating all data to a central server, federated learning allows training datasets to remain on individual clients. By letting individual clients perform local training on their respective datasets and periodically sharing model updates with a parameter server, federated learning enhances scalability, data privacy, and fault tolerance compared with centralized learning, and has garnered widespread attention in recent years (Ma et al., 2020; Glasgow et al., 2022; Woodworth et al., 2020; Yuan & Ma, 2020; Patel et al., 2024; Agarwal et al., 2018; M Ghari & Shen, 2024; Duan et al., 2023; Acar et al., 2021; Reisizadeh et al., 2020).

In federated learning, to reduce communication overhead, each client performs multiple local updates based on its local dataset before communicating with the parameter server to synchronize its local model parameter with those of other clients. However, this asynchronicity between local updates and communication operations leads to convergence errors when the data distribution is not IID (independent and identically distributed) across clients. More specifically, as pointed out in many existing results such as Karimireddy et al. (2020); Li et al. (2019); Malinovskiy et al. (2020); Charles & Konečnỳ (2020); Charles & Konečný (2021); Pathak & Wainwright (2020); Liu et al. (2026), the incorporation of multiple local updates between two communication rounds introduces a drift in a local client's update as it tends to let local clients converge to their own local optimal solutions rather than the global optimal solution, leading to inaccurate and unstable convergence. Although a diminishing stepsize can mitigate this "client-drift" problem, it inevitably slows down convergence and is often undesirable in many applications.

Recently, several algorithms have been proposed to tackle the "client-drift" problem and ensure accurate federated learning under a constant stepsize, with typical examples including SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), and FedTrack (Mitra et al., 2021a). SCAFFOLD mitigates the "client-drift" problem by using control variates that correct each client's local update direction to stay aligned with the global loss function, while FedLin and FedTrack addresses the "client-drift" problem by applying a linearized global correction to the aggregated gradient without requiring per-client control variates. The basic idea of these approaches is to let each client locally store and maintain auxiliary variables (in addition to the model parameters) to locally estimate the global gradient or the drift from it. However, these auxiliary variables incur significant overhead in storage and communication, particularly in high-dimensional federated-learning problems, because these auxiliary variables have the same dimension as the model parameters. In this paper, we propose a federated-learning architecture that can avoid using auxiliary variables while ensuring accurate convergence under non-IID data with a constant stepsize. The main contributions are summarized as follows:

- **New Algorithm**: We propose FedRecu which ensures accurate convergence in federated learning under **non-IID** data without using auxiliary variables. A key idea is the introduction of a recursive mechanism which enables each client to use gradients in both the current and previous steps in the update. The integration of the previous-step gradient is significant in that, through a judiciously designed update mechanism, it enables each client to **locally correct its local drift** and ensure accurate convergence. This design is inspired by EXTRA (Shi et al., 2015) but has a significant difference: EXTRA does not allow multiple local updates between communication rounds (it may diverge under multiple local updates), whereas our design of the update and interaction mechanisms ensures accurate convergence under multiple local updates.
- **Enhanced Communication and Memory Efficiency**: Our recursive mechanism significantly reduces communication overhead compared with existing federated-learning algorithms with "client-drift" correction. Our FedRecu only shares one variable (a linear combination of model parameters and gradients), which is drastically different from existing "client-drift" correction algorithms that have to share both the model parameter and an additional drift-correcting auxiliary variable. Moreover, our recursive mechanism eliminates the need for auxiliary variables used by existing methods (e.g., SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), FedTrack (Mitra et al., 2021a) and Scaffnew (Mishchenko et al., 2022)) to correct client drift, resulting in significantly lower memory requirements for storing intermediate variables compared to these algorithms.
- **Faster Convergence**: We prove that FedRecu achieves $o(1/K)$ convergence under general convex loss functions. This represents a significant improvement over the standard $O(1/K)$ convergence

typically observed in federated learning (for instance, the famous FedAvg has been shown in Glasgow et al. (2022) to be incapable of achieving faster than $O(1/K)$ convergence for general convex objectives, even under IID distributions; even after incorporating momentum, current algorithms still guarantee only $O(1/K)$ convergence when no additional heterogeneity constraints are imposed on convex loss functions (Xu et al., 2021; Liu et al., 2020; Cheng et al., 2023; Yang et al., 2022)). To the best of our knowledge, apart from Liu et al. (2026), FedRecu is among the **first** algorithms to ensure $o(1/K)$ convergence for federated learning under **general convex** loss functions. This stands in contrast to prior results, where $o(1/K)$ convergence have only been established under **special** conditions—such as gradient difference being uniformly bounded (Jiang et al., 2024), or Hessian difference being uniformly bounded (Kovalev et al., 2022). We also characterize the convergence of our algorithm under **nonconvex** loss functions and stochastic gradients, yielding results that outperform existing algorithms.

- **Larger Stepsizes**: We theoretically prove that our new algorithm structure allows using much larger stepsizes than existing federated-learning algorithms tackling "client-drift" caused by non-IID data. Our theoretical analysis finds that our stepsize can be at least 6, 8, 6, and 49 times larger than those used in Khaled et al. (2020), Mitra et al. (2021a), Mitra et al. (2021b), and Karimireddy et al. (2020), respectively (see Table 2).

2 Preliminaries

Notations We use $\mathbb{Z}^+$ and $\mathbb{R}^n$ to denote the sets of positive integers and real n-dimensional vectors, respectively. We write the inner product as $\langle x, y\rangle = \sum_{i=1}^n [x]_i[y]_i$ for $x, y \in \mathbb{R}^n$, where $[x]_j$ and $[y]_j$ are the j^{th} elements of the vectors x and y, respectively. We use $[A]_{ij}$ to denote the $(i,j)^{th}$ element of a matrix $A \in \mathbb{R}^{m\times n}$. We denote the transposes of $y \in \mathbb{R}^n$ and $A \in \mathbb{R}^{m\times n}$ as $y^{\mathbf{T}}$ and $A^{\mathbf{T}}$, respectively. We represent the Euclidean norm of $x \in \mathbb{R}^n$ as $\|x\| = \sqrt{\sum_{j=1}^n [x]_j^2}$. Given $a \in \mathbb{Z}^+$ and $b \in \mathbb{Z}^+$, a **mod** b represents the remainder of the division of a by b. We use $O(c(t))$ and $o(c(t))$ to represent sequences $d(t)$ satisfying $\limsup_{t\to+\infty} |\frac{d(t)}{c(t)}| < \infty$ and $\lim_{t\to\infty} \frac{d(t)}{c(t)} = 0$, respectively.

2.1 Problem Setting

We consider the following federated-learning problem over a client set $\mathcal{S} = \{1, 2, \cdots, N\}$:

$$\min_{x\in\mathbb{R}^n} f(x) = \frac{1}{N}\sum_{i=1}^N f_i(x), \tag{1}$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$ is the local loss function and is solely dependent on the local training data of client $i \in \mathcal{S}$. Due to non-IID data, the local optimum of $f_i(x)$ is generally different from the global optimum of $f(x)$. We make the following standard assumption on the local loss functions $f_i(x)$:

Assumption 1. *The loss function $f_i(x)$ of client $i \in \mathcal{S}$ is L-smooth over $\mathbb{R}^n$, that is, there exists a constant $L > 0$ such that*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \le L\|x - y\|$$

holds for any $x, y \in \mathbb{R}^n$.

From Assumption 1 and the definition of $f(x)$ in (1), we can easily obtain that $f(x)$ is also L-smooth. In addition, we make the following standard assumption to make sure that (1) has a solution:

Assumption 2. *The optimal solution set $\mathcal{X}^* = \{x^* \in \mathbb{R}^n | x^* = \arg\min_{x\in\mathbb{R}^n} f(x)\}$ is not empty, i.e., there exists at least one $x^* \in \mathbb{R}^n$ such that $f(x^*) \le f(x)$ holds for any $x \in \mathbb{R}^n$.*

Assumptions 1 and 2 are widely used in federated learning (Mitra et al., 2021a;b; Mukherjee et al., 2023; Qin et al., 2022; Karimireddy et al., 2020; Khaled et al., 2020). They are more general than assuming strong convexity or Polyak-Lojasiewicz (PL) condition on $f(x)$. It is worth noting that under Assumptions 1 and 2, the global optimal solution may not be unique.

3 Main Results

In this section, we first describe the core recursion-based update mechanism in Section 3.1. Then we summarize the detailed algorithm in Algorithm 1 in Section 3.2 and characterize its convergence performance for both general convex loss functions and nonconvex loss functions in Section 3.3 (deterministic gradients) and Section 3.4 (stochastic gradients).

3.1 Recursion-Based Mechanism

The core idea of our new algorithmic framework is using recursion to employ information in both the current step and the past step to generate the new model parameter. Specifically, the local update for agent i has the following form:

$$\begin{aligned}&\textbf{for } k\tau < t < (k+1)\tau\textbf{:}\\ &\quad x_i(t+1) = \underbrace{2x_i(t) - \alpha\nabla f_i(x_i(t))}_{\textbf{Current}} - \underbrace{x_i(t-1) + \alpha\nabla f_i(x_i(t-1))}_{\textbf{Past}},\\ &\textbf{end}\end{aligned} \tag{2}$$

where $x_i(t)$ is the local model parameter and α denotes the stepsize. Our design is inspired by the decentralized optimization algorithm EXTRA (Shi et al., 2015) but has a fundamental difference: EXTRA has two different consensus matrices multiplied on $x_i(t)$ and $x_i(t-1)$, respectively, whereas we remove such consensus coupling. This difference is key to ensuring the convergence of our algorithm when multiple updates are performed between communication rounds, whereas EXTRA only has provable convergence when one local update is conducted between two consecutive communication rounds (under multiple local updates, EXTRA is subject to the "client-drift" problem and can even diverge). The detailed algorithm is given in Section 3.2.

Remark 1. *The update in (2) effectively addresses client drifts by incorporating both the current local gradient, $\nabla f_i(x_i(t))$, and the past local gradient, $\nabla f_i(x_i(t-1))$, into the update rule. This dual-gradient-based mechanism ensures that the global optimum x^* is a fixed point of the iteration process—that is, the iterates will remain unchanged when initialized at the global optimum x^*. This stands in stark contrast to most existing algorithms without drift correction, whose updates rely solely on the current local gradient. Since the local gradient is generally nonzero at the global optimum x^* due to non-IID data (i.e., generally $\nabla f_i(x^*) \neq 0$ holds), the nonzero force exerted by the local gradient $\nabla f_i(x^*)$ will move such algorithms away from x^* even when initialized at x^*.*

3.2 Algorithm Descriptions

Some notations should be introduced before introducing our algorithm. The stepsize and the number of local updates are denoted as $\alpha > 0$ and $\tau \geq 1$, respectively. The model parameter of client $i \in \mathcal{S}$ at iteration time t is denoted as $x_i(t)$. Owing to the recursive update mechanism, FedRecu requires two initial values $x_i(-2)$ and $x_i(-1)$, which should follow the rules: $x_i(-2)$ can be arbitrarily chosen in $\mathbb{R}^n$ whereas $x_i(-1)$ should be set as $x_i(-1) = x_i(-2) - \alpha\nabla f_i(x_i(-2))$. Now, we are in a position to present the algorithm in Algorithm 1:

Unlike existing federated learning algorithms that rely on auxiliary variables to track the global gradient or the drift from it (e.g., SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), and FedTrack (Mitra et al., 2021a)), FedRecu does not use any additional variables. This leads to improved memory efficiency, despite the need to store model parameters from the previous step. To illustrate this, we compare FedRecu with FedLin in the convex setting. Each client in FedLin is required to store four n-dimensional variables: the local model parameter, the global model parameter, the auxiliary variable used to track the global gradient, and a running average of the local model parameter. (We exclude gradients from this count, as they can be recomputed from model parameters.) In contrast, FedRecu only requires each client to store two n-dimensional variables—the current and previous model parameters. A detailed memory comparison is provided in Table 1

In addition, in FedRecu, the information shared between clients and the parameter server is always one n-dimensional vector, which is a linear combination of the model parameter and the gradient (more specifically,

Algorithm 1 FedRecu

Initialization: the local training period $\tau \geq 1$, the stepsize $\alpha > 0$, the initial values $x_i(-2)$, and $x_i(-1)$ for any $i \in \mathcal{S}$.

for $t = -1$ **to** T **do**

  for each client $i = 1, 2, \cdots, N$ in parallel **do**

    if $t + 1$ **mod** $\tau = 0$ **then**

      Client i transmits $v_i(t) \triangleq 2x_i(t) - x_i(t-1) - \alpha\nabla f_i(x_i(t)) + \alpha\nabla f_i(x_i(t-1))$ to the parameter server and receives $\frac{1}{N}\sum_{j=1}^{N} v_j(t)$ from the parameter server. Then, each client i updates its model parameter as

$$x_i(t+1) = \frac{1}{N}\sum_{j=1}^{N} v_j(t). \tag{3}$$

    else if t **mod** $\tau = 0$ **then**

      Client i transmits $w_i(t) \triangleq x_i(t-1) + \alpha\nabla f_i(x_i(t)) - \alpha\nabla f_i(x_i(t-1))$ to the parameter server and receives $\frac{1}{N}\sum_{j=1}^{N} w_j(t)$ from the parameter server. Then, each client i updates its model parameter as

$$x_i(t+1) = 2x_i(t) - \frac{1}{N}\sum_{j=1}^{N} w_j(t). \tag{4}$$

    else

      Each client i does local updates

$$x_i(t+1) = 2x_i(t) - x_i(t-1) - \alpha\nabla f_i(x_i(t)) + \alpha\nabla f_i(x_i(t-1)). \tag{5}$$

    end if

  end for

end for

at $t = p\tau - 1$, $v_i(t)$ is shared between local clients and the parameter server and, at $t = p\tau$, $w_i(t)$ is shared). This is fundamentally different from existing "client-drift" correcting algorithms such as SCAFFOLD, FedLin, and FedTrack, where in each communication round, two n-dimensional vectors (the model parameter and an auxiliary variable estimating the global gradient or the derivation from it caused by non-IID data) are shared between each client and the parameter server. In fact, when $\tau = 1$, it can be seen that only one variable is shared in each communication round in our algorithm, which makes our communication overhead only half of that in SCAFFOLD, FedLin, and FedTrack. Table 1 provides a detailed comparison of FedRecu with existing algorithms regarding memory and communication requirements, which clearly shows the advantage of FedRecu in storage and communication overhead over existing counterpart algorithms.

Remark 2. *The recursive mechanism in FedRecu is inspired by the distributed optimization algorithm EXTRA (Shi et al., 2015). However, EXTRA only allows one local update in each communication round and directly extending it to incorporate multiple local updates still suffers from "client drifts" or even divergence. In contrast, FedRecu fundamentally revises the recursion and interaction mechanisms, and the associated proof techniques to accommodate multiple local updates. Specifically, we judiciously introduce different communication and update strategies for iterations $t = p\tau - 1$ and $t = p\tau$ (see (3) and (4) in Algorithm 1) to ensure accurate convergence under multiple local updates. It is important to note that neither process (3) nor (4) alone can guarantee accurate convergence, as both processes are essential to eliminating "client drifts".*

Remark 3. *We can explain why EXTRA behaves differently from our approach, FedRecu, under multiple local updates by examining their respective equivalent formulations in this setting. To this end, we define: $X(t) = [x_1^T(t), x_2^T(t), \cdots, x_N^T(t)]^T$, $\nabla f(X(t)) = [\nabla f_1(x_1^T(t)), \nabla f_2(x_2^T(t)), \cdots, \nabla f_N(x_N^T(t))]^T$, $W(t+1) = \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T$ for $t + 1 = \tau k$, and $W(t+1) = \mathbf{I}_N$ for $t + 1 \neq \tau k$.*

Directly applying multiple local updates in EXTRA yields the following matrix form:

$$X(t+1) = \frac{W(t+1)+\mathbf{I}_N}{2}\big(2X(t) - X(t-1)\big) - \alpha\nabla f(X(t)) + \alpha\nabla f(X(t-1)),$$

By defining $Y(t) = \frac{1}{\alpha}\{X(t-1) - X(t) - \alpha\nabla f(X(t-1))\}$*, we can verify that EXTRA is equivalent to the following form:*

$$\begin{cases} Y(t+1) = Y(t) + \dfrac{1}{2\alpha}(\mathbf{I}_N - W(t+1))\big\{X(t) - \alpha\nabla f(X(t-1)) - \alpha Y(t)\big\}, \\ X(t+1) = X(t) - \alpha\nabla f(X(t)) - \alpha Y(t+1). \end{cases} \tag{6}$$

Similarly, by defining $Y(t+1) = W(t+1)\{Y(t) + \nabla f(X(t+1)) - \nabla f(X(t))\}$ *with inital value* $Y(-1) = \nabla f(X(-1))$*, we can have an equivalent form of FedRecu as (see Lemma 2 of Appendix B for the proof of equivalence)*

$$\begin{cases} X(t+1) = W(t+1)\big(X(t) - \alpha Y(t)\big), \\ Y(t+1) = W(t+1)\big(Y(t) + \nabla f(X(t+1)) - \nabla f(X(t))\big). \end{cases} \tag{7}$$

From the equivalent formulation of our algorithm in (7), we observe that our algorithm, FedRecu, applies the averaging matrix $W(t)$ *twice—once in updating the variable* $X(t)$ *and once in updating* $Y(t)$*. In contrast, from the equivalent formulation of EXTRA in (6), one can see that EXTRA applies the averaging matrix* $W(t)$ *only once (to* $Y(t)$*, but not to* $X(t)$*). This double application of* $W(t)$ *in FedRecu enforces a stronger consensus at each iteration, whereas EXTRA's single application yields weaker consensus enforcement.*

Under multiple local updates, where local-model drift naturally arises, maintaining consensus is crucial for effective drift correction and accurate convergence. This difference explains why FedRecu provides strong convergence guarantees in the setting of multiple local updates (see Section 3.3) whereas EXTRA cannot. Numerical experiments in Appendix A.3 further corroborate this observation.

Remark 4. *Based on the definitions in Remark 3, FedRecu can be expressed in the equivalent form given in (7). To provide an intuitive explanation for FedRecu, we will explain that* $Y(t) = [y_1^T(t), y_2^T(t), \cdots, y_N^T(t)]^T$ *is an estimate of the global gradient. Based on the definition of* $W(t)$*, taking the network average gives*

$$\overline{Y}(t+1) = \overline{Y}(t) + \overline{\nabla f}(X(t+1)) - \overline{\nabla f}(X(t)),$$

where $\overline{\nabla f}(X(t)) = \frac{1}{N}\sum_{i=1}^N \nabla f_i(x_i(t))$ *and* $\overline{Y}(t) = \frac{1}{N}\sum_{i=1}^N y_i(t)$*. The initial value setting* $Y(-2) = \nabla f(X(-2))$ *further implies*

$$\overline{Y}(t) = \overline{\nabla f}(X(t)).$$

Therefore, the average of $Y(t)$ *coincides with the true global gradient at every iteration, and the consensus action of* $W(t)$ *ensures that all local* $y_i(t)$ *converge to this average. Hence,* $Y(t)$ *is mathematically an accurate* ***estimate of the global gradient****. At each iteration, the local model parameters (i.e.,* $X(t)$*) are updated based on the* ***global gradient estimate*** *(i.e.,* $Y(t)$*) rather than the local gradients. This mechanism ensures that the local model parameters converge to the* ***optimal solution of the global loss function****, rather than to the optima of the individual local loss functions resulting from non-IID data.*

3.3 Convergence Analysis under Deterministic Gradients

In this subsection, we analyze the convergence of FedRecu under both convex and nonconvex loss functions in the deterministic gradient case. We would like to emphasize that the results apply to **both the IID and non-IID** data cases, as we do not require the local optima of $f_i(x)$ to be identical to the global optima of $f(x)$. Given that the information exchange between clients and the parameter server occurs periodically, we characterize the convergence behavior of the model parameter at iterations when communication is conducted, i.e., $t = k\tau$.

Theorem 1 (Convex and Deterministic Case)**.** *Under Assumption 1 and Assumption 2, if the loss function $f_i(x)$ of client $i \in \mathcal{S}$ is convex over $\mathbb{R}^n$ and the stepsize satisfies $0 < \alpha \leq \frac{8}{13\tau L}$, then for any $i \in \mathcal{S}$, FedRecu guarantees that $f(x_i(K\tau))$ converges to $f(x^*)$ at a rate of $o(1/K)$, i.e.,*

$$\lim_{K\to\infty} K\Big\{f(x_i(K\tau)) - f(x^*)\Big\} = 0.$$

Proof. See Appendix C. □

Theorem 1 demonstrates that FedRecu effectively eliminates "client drifts" and ensures accurate convergence. Notably, we prove that FedRecu achieves a convergence rate of $o(1/K)$ for general convex loss functions, which is a significant improvement over the $O(1/K)$ rate obtained in existing results. In fact, to our knowledge, apart from Liu et al. (2026), FedRecu is among the **first** algorithms to ensure $o(1/K)$ convergence for federated learning under **general convex** loss functions (note that $o(1/K)$ convergence have only been established in the literature under **special convex** conditions—such as gradient difference being uniformly bounded (Jiang et al., 2024), or Hessian difference being uniformly bounded (Kovalev et al., 2022)). In addition, FedRecu supports a much larger stepsize compared with existing federated-learning algorithms tackling "client drifts" (see detailed comparison in Table 2), which, as confirmed in the numerical experiments, enables FedRecu to converge much faster than existing counterpart algorithms.

Under general nonconvex loss functions, FedRecu can also enable accurate convergence:

Theorem 2 (Nonconvex and Deterministic Case)**.** *Under Assumption 1 and Assumption 2, if the stepsize satisfies $0 < \alpha \leq \frac{8}{17\tau L}$, then for any $i \in \mathcal{S}$, the iterates under FedRecu satisfy*

$$\frac{1}{K}\sum_{k=0}^{K-1} \|\nabla f(x_i(k\tau))\|^2 \leq \frac{8}{13\alpha^2 L\tau K}\Big(f(x_i(0)) - f(x^*)\Big).$$

Proof. See Appendix E. □

Theorem 2 shows that even in the nonconvex setting, FedRecu can also avoid "client drifts" and ensure convergence to a desired solution. It is worth noting that the proposed stepsize range $0 < \alpha \leq \frac{8}{17\tau L}$ is significantly larger than those permitted in existing federated-learning algorithms (see Table 2 for detailed comparison).

Remark 5. *We have established the convergence rates of $o(1/K)$ and $O(1/K)$ for FedRecu in the general convex case and the general nonconvex case, respectively. In the special case where the global loss function $f(x)$ is μ-strongly convex or satisfies the μ-PL condition, following the presented proof techniques, the linear convergence of $f(x_i(k\tau)) - f(x^*)$ can be directly obtained with the proposed stepsizes in Theorem 1 and Theorem 2, respectively.*

Remark 6. *Based on the reformulation in Remark 4, one can see that FedRecu is mathematically equivalent to gradient-tracking-based algorithms that incorporate multiple local updates. However, it is crucial to note that **FedRecu offers more than a mere equivalence**. Specifically, it introduces several key advantages over conventional gradient-tracking methods, which we detail below.*

- ***Sharper Convergence Guarantee:** A key theoretical contribution of FedRecu is the establishment of a **monotonicity property** (see Appendix C): $f(x_i(k\tau+\tau)) \leq f(x_i(k\tau))$, $\forall i \in \mathcal{V}$, for general convex loss functions. Leveraging this property, we prove that FedRecu achieves an $o(1/K)$ **convergence rate** for general convex loss functions. This rate constitutes a strict improvement over the standard $O(1/T)$ rate attainable by existing gradient-tracking methods under general convex loss functions. To the best of our knowledge, apart from Liu et al. (2026), FedRecu is among the **first** federated learning algorithms to guarantee $o(1/K)$ convergence under **general convex** loss functions. Previous results establishing such $o(1/K)$ convergence rate must rely on **stronger assumptions**, such as uniformly bounded gradient differences (Jiang et al., 2024) or uniformly bounded Hessian differences (Jiang et al., 2024).*

- ***Memory Efficiency:*** *To quantify memory overhead, we measure the number of n-dimensional vectors that must be stored, where n is the dimension of the model parameter. It is worth noting that we do not consider gradients since they can be computed directly using the model parameters. In the standard gradient-tracking formulation, each client must maintain* ***three n-dimensional vectors****: the current model parameter, the previous model parameter, and an auxiliary gradient-tracking variable. In contrast, the recursive formulation of FedRecu* ***eliminates the explicit gradient-tracking variable****. Consequently, each client in FedRecu needs to store only* ***two n-dimensional vectors****: the current and the previous model parameters. Therefore, this recursive design yields a fundamentally more memory-efficient algorithm by inherently incorporating past gradient information into the local update rule, thereby obviating the need for a separate auxiliary gradient-tracking variable.*
- ***Communication Efficiency:*** *In addition to memory efficiency, FedRecu also improves communication efficiency compared with gradient-tracking-based algorithms. In each communication round, clients in our FedRecu only need to share a single n-dimensional vector (a simple linear combination of model parameters and gradients) with the parameter server. In contrast, gradient-tracking-based algorithms require the transmission of two n-dimensional vectors: the model parameter and the gradient-tracking variable.*

3.4 Convergence Analysis under Stochastic Gradients

In this subsection, we extend our analysis to the more practical setting of stochastic gradients (the mini-batch setting). In this case, the local loss function $f_i(x)$ is determined by

$$f_i(x) = \mathbb{E}_{\xi_i \sim D_i}[f_i(x, \xi_i)], \tag{8}$$

where ξ_i denotes a stochastic data sample drawn from the local distribution D_i of client i. As a result, client i can only access a stochastic estimate $\nabla f_i(x, \xi_i)$ of the true gradient $\nabla f_i(x)$ for any $x \in \mathbb{R}^n$. We use the following standard assumption regarding the stochastic gradient (Karimireddy et al. (2020); Mukherjee et al. (2023); Jhunjhunwala et al. (2023)):

Assumption 3. *The stochastic gradient $\nabla f_i(x, \xi_i)$ is an unbiased estimate of the accurate gradient $\nabla f_i(x)$, with its variance bounded by σ^2. Specifically, we have*

$$\mathbb{E}_{\xi_i \sim D_i}[\nabla f_i(x, \xi_i)] = \nabla f_i(x), \quad \textit{and} \quad \mathbb{E}_{\xi_i \sim D_i}[\|\nabla f_i(x, \xi_i) - \nabla f_i(x)\|^2] \leq \sigma^2,$$

for any $x \in \mathbb{R}^n$ and $i \in \mathcal{S}$.

In the stochastic gradient setting, the exact gradients $\nabla f_i(x_i(t))$ and $\nabla f_i(x_i(t-1))$ of FedRecu should be replaced with their stochastic counterparts $\nabla f_i(x_i(t), \xi_i(t))$ and $\nabla f_i(x_i(t-1), \xi_i(t-1))$, respectively, where $\xi_i(t) \sim D_i$ are samples drawn from the local data distribution at each iteration. Next, we establish the convergence properties of FedRecu in this stochastic setting for both convex and nonconvex loss functions. Again, the results apply to **both the IID and non-IID** data cases, as we do not require the local optima of $f_i(x)$ to be identical to the global optima of $f(x)$.

Theorem 3 (Convex and Stochastic Case)**.** *Under Assumption 1, Assumption 2, and Assumption 3, if the loss function $f_i(x)$ of client $i \in \mathcal{S}$ is convex and the stepsize satisfies $0 < \alpha < \frac{1}{6\tau L}$, then for any $i \in \mathcal{S}$, the iterates under FedRecu satisfy*

$$\mathbb{E}\Big[f\Big(\frac{1}{K}\sum_{k=0}^{K-1} x_i(k\tau)\Big)\Big] - f(x^*) \leq \frac{\mathbb{E}[\|x_i(0) - x^*\|^2]}{(2\alpha\tau - 12\tau^2 L\alpha^2)K} + \frac{34\tau^2\alpha^2}{2\alpha\tau - 12\tau^2 L\alpha^2}\sigma^2.$$

Proof. See Appendix G. □

Theorem 4 (Nonconvex and Stochastic Case)**.** *Under Assumption 1, Assumption 2, and Assumption 3, if the stepsize satisfies $0 < \alpha < \frac{1}{13\tau L}$, then for any $i \in \mathcal{S}$, the iterates under FedRecu satisfy*

$$\frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}\Big[\|\nabla f(x_j(k\tau))\|^2\Big] \leq \frac{\mathbb{E}[f(x_i(0))] - f(x^*)}{(\frac{\alpha\tau}{2} - \frac{13}{2}\tau^2 L\alpha^2)K} + \frac{44\tau^2\alpha^2 L}{\alpha\tau - 13\tau^2 L\alpha^2}\sigma^2.$$

Proof. See Appendix I. □

Theorem 3 and Theorem 4 show that, in the presence of noisy gradients, a constant stepsize can only ensure convergence to a neighborhood of the optimal solution. The size of this neighborhood depends on the local update period τ, the stepsize α, the smoothness constant L, and the variance σ^2 of the stochastic gradients.

Remark 7. *In the stochastic gradient setting, FedRecu can still guarantee accurate convergence by adopting a diminishing stepsize. For instance, following a similar line of reasoning in Theorem 3 and Theorem 4, one can easily obtain that setting* $\alpha = O(1/\sqrt{K})$ *yields*

$$\mathbb{E}\Big[f(\frac{1}{K}\sum_{k=0}^{K-1} x_i(k\tau))\Big] - f(x^*) \leq O(1/\sqrt{K}), \text{ and } \frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\Big[\|\nabla f(x_i(k\tau))\|^2\Big] \leq O(1/\sqrt{K}),$$

for convex and nonconvex loss functions, respectively. However, despite ensuring accurate convergence, a diminishing stepsize slows down convergence compared to the constant stepsize case.

4 Comparisons with Existing Works

In this section, we systematically show that FedRecu has advantages in storage and communication overheads, stepsize, and convergences rates with respect to existing counterpart algorithms.

Table 1: Comparison of the required memory and communicated messages between FedRecu and existing algorithms addressing "client drifts" in federated learning.

Algorithm	Memory Overhead[1]			Communication Overhead[2]	
	strongly convex	convex	nonconvex	$\tau \geq 2$	$\tau = 1$
This work[3]	2	2	3	2	1
Mitra et al. (2021b)[4]	3	4	4	2	2
Karimireddy et al. (2020)	4	5	5	2	2
Mitra et al. (2021a)	3	–	4	2	2
Huang et al. (2023)	–	–	4	2	2
Huang et al. (2024)[4]	–	–	5	2	2
Sun & Wei (2022)	4	–	–	2	2

[1] To quantify memory overhead, we measure the number of n-dimensional variables that must be stored, where n is the dimension of the model parameter. It is worth noting that we do not consider gradients since they can be computed directly using the model parameter.
[2] To quantify communication overhead, we measure the number of n-dimensional variables shared after every τ local updates.
[3] The convergence result of this work for the convex case relies on the last-iterate terms (see Theorem 1). In contrast to the nonconvex setting, no running-averaged iterate needs to be stored.
[4] Note that message compression addressed in the paper is orthogonal to the message count-based communication efficiency discussed here.

4.1 More Efficient Memory and Communication

A significant advantage of FedRecu lies in its memory-efficient design. Existing federated-learning algorithms, such as SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), FedTrack (Mitra et al., 2021a), and Scaffnew (Mishchenko et al., 2022), rely on auxiliary variables, such as control or gradient tracking variables, to address the "client-drift" problem. However, storing and updating these variables incur significant extra overhead in memory consumption. In contrast, FedRecu leverages a recursive mechanism that naturally incorporates both current and past gradient information into the local update rule, eliminating the need for using auxiliary variables and resulting in reduced memory requirement (despite requiring to store the past model parameter).

In addition to memory efficiency, FedRecu also improves communication efficiency compared with existing counterpart algorithms. In each communication round, clients in our FedRecu only need to share a single n-dimensional vector (a simple linear combination of model parameters and gradients, see $v_i(t)$ and $w_i(t)$

in Algorithm 1 for details) with the parameter server. In contrast, existing federated-learning algorithms addressing "client drifts", such as SCAFFOLD, FedLin, and FedTrack, require the transmission of multiple variables. In fact, when $\tau = 1$, FedRecu reduces to

$$x_i(t+1) = \frac{1}{N}\sum_{j=1}^{N}\Big\{2x_j(t) - x_j(t-1) - \alpha\nabla f_j(x_j(t)) + \alpha\nabla f_j(x_j(t-1))\Big\}.$$

In this case, FedRecu only shares one variable between local clients and the parameter server in each communication round, which reduces the communication overhead in SCAFFOLD, FedLin, and FedTrack by a half. Table 1 provides a detailed comparison of FedRecu with existing counterpart algorithms regarding memory and communication requirements.

Table 2: Comparison of allowable stepsizes and obtained convergence rates between FedRecu and existing federated-learning algorithms (under τ local training steps)

Assumption	Algorithm	Stepsize	Convergence Rate	Gradient Setting[1]
Convex	This Work (**ours**)	$8/(13\tau L)$	$o(1/K)$	EG
		$1/(6\tau L)$	$O(1/K)$	SG
	Mitra et al. (2021b)	$1/(10\tau L)$	$O(1/K)$	EG
	Karimireddy et al. (2020)	$1/(81\tau L)$	$O(1/K)$	SG
	Mukherjee et al. (2023)	$1/(20\tau L)$	$O(1/K)$	SG
	Khaled et al. (2020)	$1/(10\tau L)$	$O(1/K)$	SG
	Qu et al. (2021)	$O(1/\sqrt{K})$	$O(1/\sqrt{K})$	SG
Nonconvex	This Work (**ours**)	$8/(17\tau L)$	$O(1/K)$	SG
		$1/(13\tau L)$	$O(1/K)$	EG
	Mitra et al. (2021b)	$1/(26\tau L)$	$O(1/K)$	EG
	Mitra et al. (2021a)	$1/(18\tau L)$	$O(1/K)$	EG
	Karimireddy et al. (2020)	$1/(24\tau L)$	$O(1/K)$	SG
	Allouah et al. (2024)	$1/(16\tau L)$	$O(1/K)$	SG
	Beikmohammadi et al. (2025) Haddadpour & Mahdavi (2019) Yang et al. (2021); Zhu et al. (2021) Yu et al. (2019); Cheng et al. (2024) Wang et al. (2020); Yan et al. (2025) Reisizadeh et al. (2020); Li & Li (2023) Xiang et al. (2024); Huang et al. (2023)	$O(1/\sqrt{K})$	$O(1/\sqrt{K})$	SG
	Kim et al. (2023)	Adaptive	$O(1/\sqrt{K})$	SG

[1] EG denotes the exact gradient setting; SG denotes the stochastic gradient setting.

4.2 Improved Convergence Rates

FedRecu offers significant advantages in terms of convergence rates. Unlike many existing methods that are subject to steady-state optimization errors (see, e.g., Jhunjhunwala et al. (2023); Wang et al. (2020); Cho et al. (2020); Wang et al. (2021)) or achieve a convergence rate of $O(1/K)$ for general convex loss functions (see, e.g., Mitra et al. (2021a;b); Karimireddy et al. (2020); Haddadpour et al. (2019)), FedRecu ensures accurate convergence at a rate of $o(1/K)$ for **general** convex loss functions. This contrasts sharply with existing results, which establish $o(1/K)$ convergence only for **special** classes of convex functions—such as those with uniformly bounded gradient differences (Jiang et al., 2024) or uniformly bounded Hessian differences (Kovalev et al., 2022).

4.3 Larger Stepsizes

FedRecu allows larger constant stepsizes than existing counterpart algorithms that tackle "client drifts" in federated learning. While some prior methods exploit diminishing stepsizes to mitigate "client drifts", this

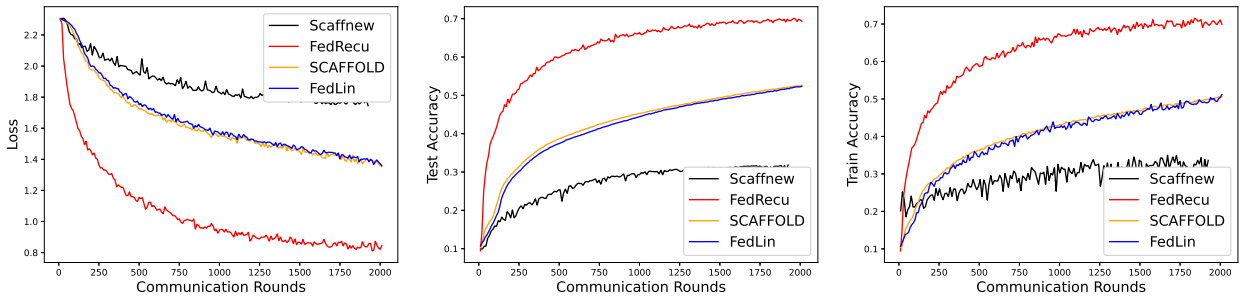
approach inevitably results in slow convergence and is not considered here. The stepsize comparisons are summarized in Table 2. It can be seen that FedRecu's stepsize can be at least 6, 8, 6, and 49 times larger than those used in Khaled et al. (2020), Mitra et al. (2021a), Mitra et al. (2021b), and Karimireddy et al. (2020), respectively.

5 Experiments

We evaluate our proposed algorithm by training a CNN on 10 clients using the benchmark datasets CIFAR-10 and CIFAR-100, respectively [1]. The CNN architecture consists of three convolutional layers with 32, 64, and 128 filters, respectively, each followed by a max-pooling layer. After the final convolutional and pooling layers, the network includes a fully connected layer with 256 units and ReLU activation, a dropout layer with a rate of 0.25 for regularization, and a final dense output layer with 10 units that produces the class logits. In our experiments, we compare the proposed algorithm against existing federated learning methods specifically designed to address client drift, including SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), and Scaffnew (Mishchenko et al., 2022). Following Hsu et al. (2019) and Kim et al. (2023), we generate heterogeneous data distributions across the 10 agents using a Dirichlet distribution, with the heterogeneity parameter β set to 0.1, 1, and 10, respectively. A higher value of β yields a nearly uniform distribution of data across classes for each client, resulting in approximately IID local datasets. In contrast, a lower β leads to highly skewed distributions, where clients tend to specialize in only a few classes.

Figures 1 and 2, report results for $\beta = 1$, which corresponds to a moderately heterogeneous setting. For all results shown in Figure 1 (CIFAR-10) and Figure 2, (CIFAR-100), the stepsizes for FedRecu, SCAFFOLD, FedLin, and Scaffnew are selected according to the guidelines from Theorem 1, Karimireddy et al. (2020), Mitra et al. (2021b), and Mishchenko et al. (2022), respectively, using an estimated smoothness parameter of $L = 2$. For FedRecu, SCAFFOLD, and FedLin, the local training period is set to $\tau = 10$. For Scaffnew, the communication probability is set to $\frac{1}{11}$ to ensure that the total number of communicated messages remains consistent across methods. As shown in the figures, our algorithm achieves faster convergence and higher accuracy on both the CIFAR-10 and CIFAR-100 datasets.

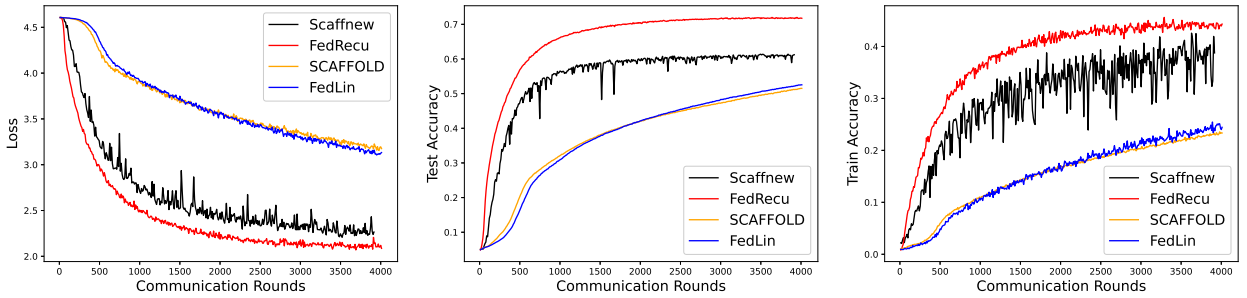
Under the same experimental setup, we also conducted experiments with Dirichlet distribution parameters $\beta = 0.1$ (reflecting a high degree of heterogeneity). The results are presented in Figures 3 and 4. It is evident that FedRecu achieves a substantial speedup compared with the baseline methods. Additional experiments with Dirichlet distribution parameters $\beta = 10$ (reflecting a low degree of heterogeneity) and experiments for the least squares problem are presented in Appendix A, which also confirm the effectiveness of our proposed algorithm.



(a) (b) (c)

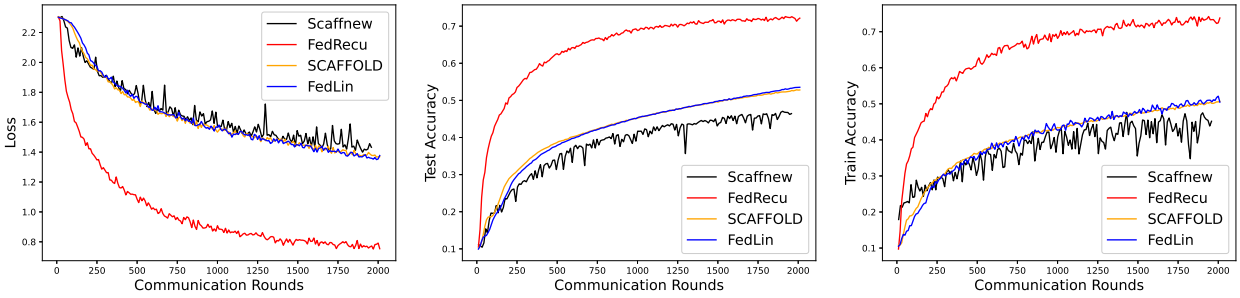
Figure 1: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, FedLin, and Scaffnew—on the CIFAR-10 dataset. Each curve represents the average of six independent runs. The Dirichlet distribution parameter was set to $\beta = 1$, which corresponds to moderate heterogeneity.

[1] Code available at https://anonymous.4open.science/r/fedrecu-E043/README.md



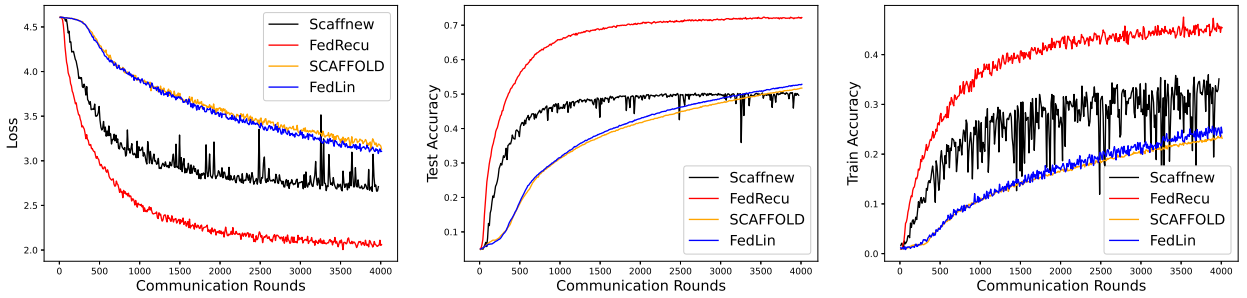
(a) (b) (c)

Figure 2: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, FedLin, and Scaffnew—on the CIFAR-100 dataset. Each curve represents the average of six independent runs. Note that the test accuracy in Figure 2(b) is top-5 accuracy. The Dirichlet distribution parameter was set to $\beta = 1$, which corresponds to moderate heterogeneity.



(a) (b) (c)

Figure 3: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, FedLin, and Scaffnew—on the CIFAR-10 dataset. Each curve represents the average of five independent runs. The Dirichlet distribution parameter was set to $\beta = 0.1$, which corresponds to a relatively high level of heterogeneity.



(a) (b) (c)

Figure 4: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, FedLin, and Scaffnew—on the CIFAR-100 dataset. Each curve represents the average of three independent runs. Note that the test accuracy in Figure 2(b) is top-5 accuracy. The Dirichlet distribution parameter was set to $\beta = 0.1$, which corresponds to a relatively high level of heterogeneity.

6 Conclusion

We have proposed FedRecu, a novel recursion-based algorithm that can address "client drifts" in federated learning. Different from all existing federated-learning algorithms that have to employ auxiliary variables to estimate the global gradient or the amounts of drift from it, the novel recursion-based architecture of our algorithm enables eliminating "client drifts" without introducing any auxiliary variables. This elimination of auxiliary variables enables our algorithm to significantly reduce the communication overhead and memory requirement in combating "client drifts" in federated learning. The novel architecture also enables employing larger constant stepsizes than existing counterpart algorithms with drift correction, resulting in much faster convergence. We provide rigorous convergence analysis of the proposed algorithm under both convex and nonconvex loss functions, in both the deterministic gradient case and the stochastic gradient case. Notably, we prove that FedRecu guarantees an $o(1/K)$ convergence rate under general convex loss functions, improving upon the existing federated learning literature, which achieves this rate only under restricted convex settings with heterogeneity constraints. Numerical experiments further confirm that FedRecu converges faster than existing counterpart algorithms that can tackle "client drifts."

Acknowledgments

The work was supported in part by the National Science Foundation under Grants CCF-2106293, CCF-2215088, CNS-2219487, CCF-2334449, and CNS-2422312.

References

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021.

Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems*, volume 31, pp. 7564–7575, 2018.

Youssef Allouah, Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Rafael Pinot, Geovani Rizk, and Sasha Voitovych. Byzantine-robust federated learning: Impact of client subsampling and local updates. *arXiv preprint arXiv:2402.12780*, 2024.

Ali Beikmohammadi, Sarit Khirirat, and Sindri Magnússon. On the convergence of federated learning algorithms without data similarity. *IEEE Transactions on Big Data*, 11(2):659–668, 2025.

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

Zachary Charles and Jakub Konečný. Convergence and accuracy trade-offs in federated learning and meta-learning. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130, pp. 2575–2583, 2021.

Mingzhe Chen, Zhaohui Yang, Walid Saad, Changchuan Yin, H. Vincent Poor, and Shuguang Cui. A joint learning and communications framework for federated learning over wireless networks. *IEEE Transactions on Wireless Communications*, 20(1):269–283, 2021.

Ziheng Cheng, Xinmeng Huang, Pengfei Wu, and Kun Yuan. Momentum benefits non-iid federated learning simply and provably. *arXiv preprint arXiv:2306.16504*, 2023.

Ziheng Cheng, Xinmeng Huang, Pengfei Wu, and Kun Yuan. Momentum benefits non-iid federated learning simply and provably. In *The Twelfth International Conference on Learning Representations*, 2024.

Yae Jee Cho, Jianyu Wang, and Gauri Joshi. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. *arXiv preprint arXiv:2010.01243*, 2020.

Jian-hui Duan, Wenzhong Li, Derun Zou, Ruichen Li, and Sanglu Lu. Federated learning with data-agnostic distribution fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8074–8083, 2023.

Margalit R Glasgow, Honglin Yuan, and Tengyu Ma. Sharp bounds for federated averaging (local SGD) and continuous perspective. In *International Conference on Artificial Intelligence and Statistics*, pp. 9050–9090, 2022.

Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen. Recovering private text in federated learning of language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 8130–8143, 2022.

Farzin Haddadpour and Mehrdad Mahdavi. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019.

Farzin Haddadpour, Mohammad Mahdi Kamani, Mehrdad Mahdavi, and Viveck Cadambe. Local SGD with periodic averaging: Tighter analysis and adaptive synchronization. In *Advances in Neural Information Processing Systems*, volume 32, pp. 11082–11094, 2019.

Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Minhui Huang, Dewei Zhang, and Kaiyi Ji. Achieving linear speedup in non-IID federated bilevel learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 14039–14059, 2023.

Xinmeng Huang, Ping Li, and Xiaoyun Li. Stochastic controlled averaging for federated learning with communication compression. In *International Conference on Representation Learning*, pp. 27956–27994, 2024.

Divyansh Jhunjhunwala, Shiqiang Wang, and Gauri Joshi. Fedexp: Speeding up federated averaging via extrapolation. *arXiv preprint arXiv:2301.09604*, 2023.

Xiaowen Jiang, Anton Rodomanov, and Sebastian U Stich. Stabilized proximal-point methods for federated optimization. *Advances in Neural Information Processing Systems*, 37:99735–99772, 2024.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 5132–5143, 2020.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Tighter theory for local SGD on identical and heterogeneous data. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108, pp. 4519–4529, 2020.

Latif U. Khan, Shashi Raj Pandey, Nguyen H. Tran, Walid Saad, Zhu Han, Minh N. H. Nguyen, and Choong Seon Hong. Federated learning for edge networks: Resource optimization and incentive mechanism. *IEEE Communications Magazine*, 58(10):88–93, 2020.

Junhyung Lyle Kim, Mohammad Taha Toghani, César A Uribe, and Anastasios Kyrillidis. Adaptive federated learning with auto-tuned clients. *arXiv preprint arXiv:2306.11201*, 2023.

Dmitry Kovalev, Aleksandr Beznosikov, Ekaterina Borodich, Alexander Gasnikov, and Gesualdo Scutari. Optimal gradient sliding and its application to optimal distributed optimization under similarity. *Advances in Neural Information Processing Systems*, 35:33494–33507, 2022.

Ching-Pei Lee and Stephen Wright. First-order algorithms converge faster than $o(1/k)$ on convex problems. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 3754–3762, 2019.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

Xiaoyun Li and Ping Li. Analysis of error feedback in federated non-convex optimization with biased compression: Fast convergence and partial participation. In *International Conference on Machine Learning*, pp. 19638–19688, 2023.

Zexi Li, Tao Lin, Xinyi Shang, and Chao Wu. Revisiting weighted aggregation in federated learning with neural networks. In *International Conference on Machine Learning*, pp. 19767–19788, 2023.

Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020a.

Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020b.

Bill Yuchen Lin, Chaoyang He, Zihang Zeng, Hulin Wang, Yufen Huang, Christophe Dupuy, Rahul Gupta, Mahdi Soltanolkotabi, Xiang Ren, and Salman Avestimehr. Fednlp: Benchmarking federated learning methods for natural language processing tasks. *arXiv preprint arXiv:2104.08815*, 2021.

Jie Liu, Zuang Wang, and Yongqiang Wang. Achieving faster than $O(1/t)$ convergence in general convex federated learning. *Transactions on Machine Learning Research*, 2026.

Ming Liu, Stella Ho, Mengqi Wang, Longxiang Gao, Yuan Jin, and He Zhang. Federated learning meets natural language processing: A survey. *arXiv preprint arXiv:2107.12603*, 2021.

Wei Liu, Li Chen, Yunfei Chen, and Wenyi Zhang. Accelerating federated learning via momentum gradient descent. *IEEE Transactions on Parallel and Distributed Systems*, 31(8):1754–1766, 2020.

Bing Luo, Xiang Li, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. Cost-effective federated learning in mobile edge networks. *IEEE Journal on Selected Areas in Communications*, 39(12):3606–3621, 2021.

Pouya M Ghari and Yanning Shen. Personalized federated learning with mixture of models for adaptive prediction and model fine-tuning. *Advances in Neural Information Processing Systems*, 37:92155–92183, 2024.

Chuan Ma, Jun Li, Ming Ding, Howard H. Yang, Feng Shu, Tony Q. S. Quek, and H. Vincent Poor. On safeguarding privacy and security in the framework of federated learning. *IEEE Network*, 34(4):242–248, 2020.

Grigory Malinovskiy, Dmitry Kovalev, Elnur Gasanov, Laurent Condat, and Peter Richtarik. From local SGD to local fixed-point methods for federated learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 6692–6701, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54, pp. 1273–1282, 2017.

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, and Peter Richtárik. Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally! In *International Conference on Machine Learning*, pp. 15750–15769. PMLR, 2022.

Aritra Mitra, Rayana Jaafar, George J Pappas, and Hamed Hassani. Federated learning with incrementally aggregated gradients. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 775–782, 2021a.

Aritra Mitra, Rayana Jaafar, George J Pappas, and Hamed Hassani. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. *Advances in Neural Information Processing Systems*, 34:14606–14619, 2021b.

Sohom Mukherjee, Nicolas Loizou, and Sebastian U Stich. Locally adaptive federated learning. *arXiv preprint arXiv:2307.06306*, 2023.

Solmaz Niknam, Harpreet S. Dhillon, and Jeffrey H. Reed. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine*, 58(6):46–51, 2020.

Kumar Kshitij Patel, Margalit Glasgow, Ali Zindari, Lingxiao Wang, Sebastian U Stich, Ziheng Cheng, Nirmit Joshi, and Nathan Srebro. The limits and potentials of local sgd for distributed heterogeneous learning with intermittent communication. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 4115–4157, 2024.

Reese Pathak and Martin J Wainwright. Fedsplit: an algorithmic framework for fast federated optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7057–7066, 2020.

Tiancheng Qin, S Rasoul Etesami, and César A Uribe. Faster convergence of local sgd for over-parameterized models. *arXiv preprint arXiv:2201.12719*, 2022.

Zhaonan Qu, Kaixiang Lin, Zhaojian Li, and Jiayu Zhou. Federated learning's blessing: Fedavg has linear speedup. In *International Conference on Learning Representations*, pp. 1–47, 2021.

Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108, pp. 2021–2031, 2020.

Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Zhenyu Sun and Ermin Wei. A communication-efficient algorithm with linear convergence for federated minimax learning. *Advances in Neural Information Processing Systems*, 35:6060–6073, 2022.

Nguyen H. Tran, Wei Bao, Albert Zomaya, Minh N. H. Nguyen, and Choong Seon Hong. Federated learning over wireless networks: Optimization model design and analysis. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pp. 1387–1395, 2019.

Yeshwanth Venkatesha, Youngeun Kim, Leandros Tassiulas, and Priyadarshini Panda. Federated learning with spiking neural networks. *IEEE Transactions on Signal Processing*, 69:6183–6194, 2021.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7611–7623, 2020.

Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. A novel framework for the analysis and design of heterogeneous federated learning. *IEEE Transactions on Signal Processing*, 69:5234–5249, 2021.

Blake Woodworth, Kumar Kshitij Patel, Sebastian Stich, Zhen Dai, Brian Bullins, Brendan Mcmahan, Ohad Shamir, and Nathan Srebro. Is local sgd better than minibatch sgd? In *International Conference on Machine Learning*, pp. 10334–10343, 2020.

Ming Xiang, Stratis Ioannidis, Edmund Yeh, Carlee Joe-Wong, and Lili Su. Efficient federated learning against heterogeneous and non-stationary client unavailability. *Advances in Neural Information Processing Systems*, 37:104281–104328, 2024.

Jing Xu, Sen Wang, Liwei Wang, and Andrew Chi-Chih Yao. Fedcm: Federated learning with client-level momentum. *arXiv preprint arXiv:2106.10874*, 2021.

Wenjing Yan, Kai Zhang, Xiaolu Wang, and Xuanyu Cao. Problem-parameter-free federated learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Haibo Yang, Minghong Fang, and Jia Liu. Achieving linear speedup with partial worker participation in non-IID federated learning. *arXiv preprint arXiv:2101.11203*, 2021.

Howard H Yang, Zuozhu Liu, Tony QS Quek, and H Vincent Poor. Scheduling policies for federated learning in wireless networks. *IEEE Transactions on Communications*, 68(1):317–333, 2019.

Zhengjie Yang, Wei Bao, Dong Yuan, Nguyen H. Tran, and Albert Y. Zomaya. Federated learning with nesterov accelerated gradient. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):4863–4873, 2022.

Rui Ye, Rui Ge, Xinyu Zhu, Jingyi Chai, Yaxin Du, Yang Liu, Yanfeng Wang, and Siheng Chen. Fedllm-bench: Realistic benchmarks for federated learning of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 111106–111130, 2024.

Hao Yu, Sen Yang, and Shenghuo Zhu. 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*, volume 33, pp. 5693–5700, 2019.

Honglin Yuan and Tengyu Ma. Federated accelerated stochastic gradient descent. *Advances in Neural Information Processing Systems*, 33:5332–5344, 2020.

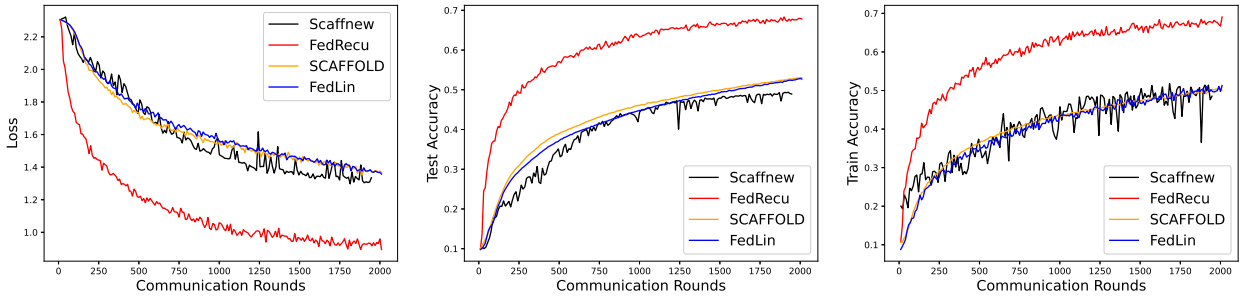
Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International conference on machine learning*, pp. 7252–7261, 2019.

Ligeng Zhu, Hongzhou Lin, Yao Lu, Yujun Lin, and Song Han. Delayed gradient averaging: Tolerate the communication latency for federated learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29995–30007, 2021.

A Additional Numerical Experiments

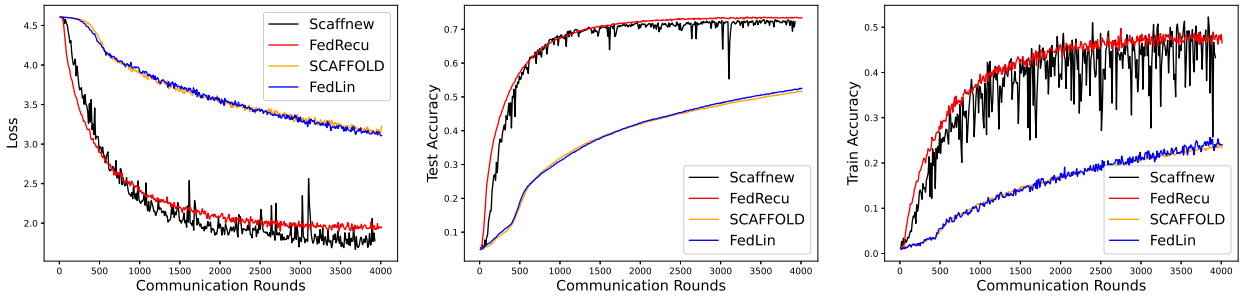
A.1 Additional CNN Training Results with A Different Non-IID Level

Additional CNN training experiments on the CIFAR-10 and CIFAR-100 datasets are presented in Figures 5 and 6, respectively, using a non-IID data distribution characterized by a Dirichlet distribution with heterogeneity parameter $\beta = 10$. The other setup is the same as that presented in Section 5.



(a) (b) (c)

Figure 5: Comparison of FedRecu with state-of-the-art federated learning algorithms SCAFFOLD, FedLin, and Scaffnew using their prescribed stepsizes on the CIFAR-10 dataset. Each curve represents the average of five independent runs. Each curve represents the average of three independent runs. Note that the test accuracy in Figure 5(b) is top-5 accuracy. The Dirichlet distribution parameter was set to $\beta = 10$, which corresponds to a relatively low level of heterogeneity.

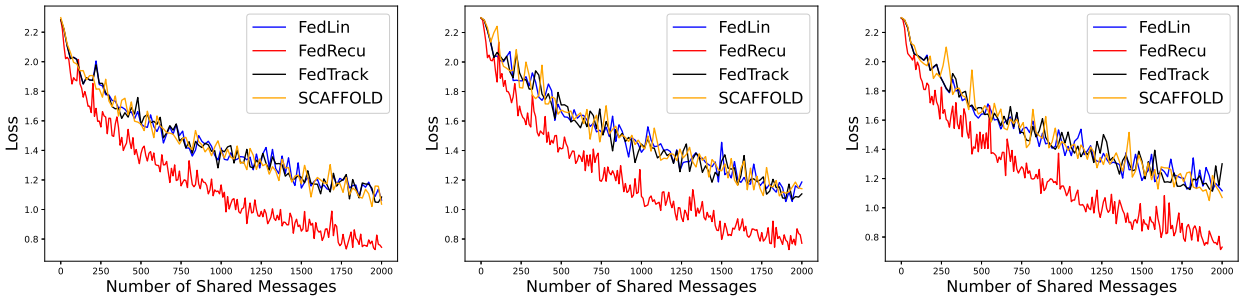


(a) (b) (c)

Figure 6: Comparison of FedRecu with state-of-the-art federated learning algorithms SCAFFOLD, FedLin, and Scaffnew using their prescribed stepsizes on the CIFAR-100 dataset. Each curve represents the average of three independent runs. Note that the test accuracy in Figure 6(b) is top-5 accuracy. The Dirichlet distribution parameter was set to $\beta = 10$, which corresponds to a relatively low level of heterogeneity.

A.2 Comparison of FedRecu, SCAFFOLD, FedTrack, and FedLin Using Their Best-Found Stepsizes for CNN Training

Additional CNN training experiments on the CIFAR-10 dataset are presented in Figure 7, using a non-IID data distribution characterized by a Dirichlet distribution with different heterogeneity parameters $\beta = 0.1, 1, 10$. In Figure 7, we compared the convergence of FedRecu with SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), and FedTrack (Mitra et al., 2021a) using the best-found stepsize for each algorithm. These experiments show that our algorithm can achieve faster convergence than the competing methods. These results confirm the effectiveness of the proposed algorithm.

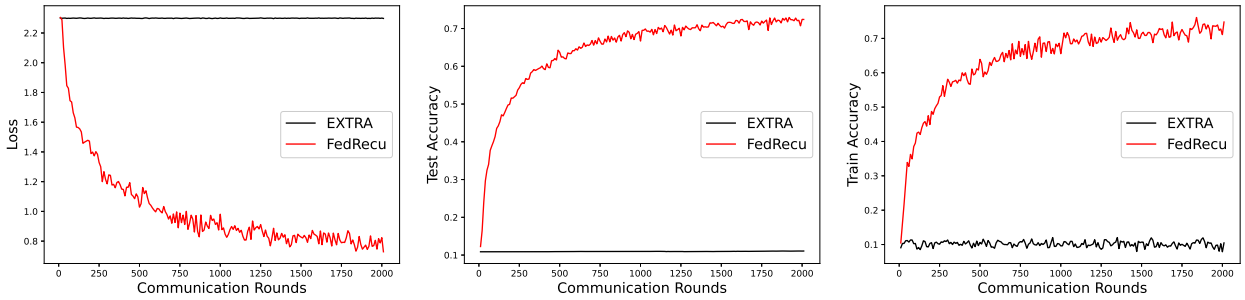


(a) (b) (c)

Figure 7: Comparison of FedRecu with state-of-the-art federated learning algorithms SCAFFOLD, FedLin, and FedTrack under their respective best-found stepsizes on the CIFAR-10 dataset. Each curve represents the average of six independent runs. To study different heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 0.1, 1, 10$ for Figures (a), (b), (c), respectively. The results demonstrate that our proposed FedRecu achieves a significant speedup compared to these baseline methods.

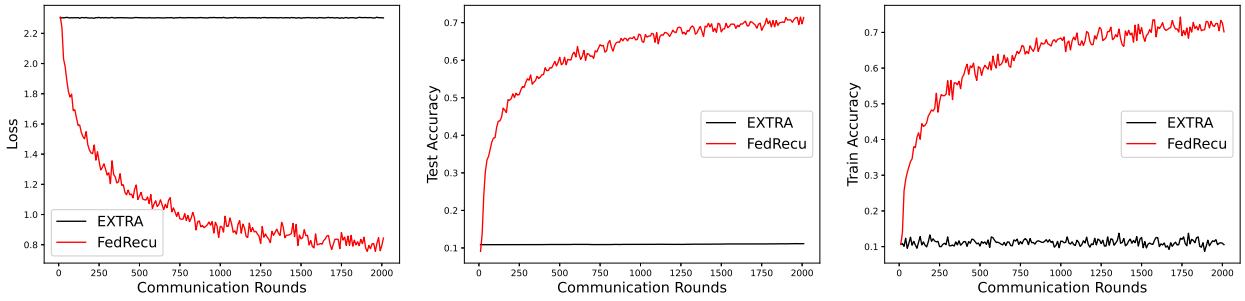
A.3 Comparison of FedRecu and EXTRA for CNN Training

Additional comparison between FedRecu and EXTRA (Shi et al., 2015) on the CIFAR-10 dataset are presented in Figures 8, 9, and 10, respectively, using non-IID data distributions generated by Dirichlet partitions with heterogeneity parameters $\beta = 0.1, 1, 10$, respectively. For both FedRecu and EXTRA (Shi et al., 2015), the local training period is set to $\tau = 10$ and the stepsize to $\alpha = 0.1$. As shown in Figures 8, 9, and 10, FedRecu achieves convergence, whereas EXTRA fails to converge. These numerical results further support the statement in Remark 3 that FedRecu allows a larger stepsize than EXTRA.



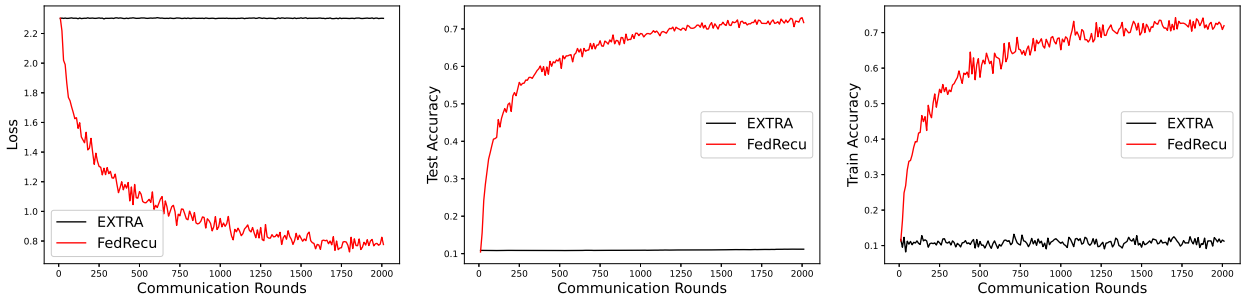
(a) (b) (c)

Figure 8: Comparison of FedRecu with EXTRA on the CIFAR-10 dataset. Each curve represents the average of six independent runs. To induce greater heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 0.1$.



(a) (b) (c)

Figure 9: Comparison of FedRecu with EXTRA on the CIFAR-10 dataset. Each curve represents the average of six independent runs. To induce greater heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 1$.



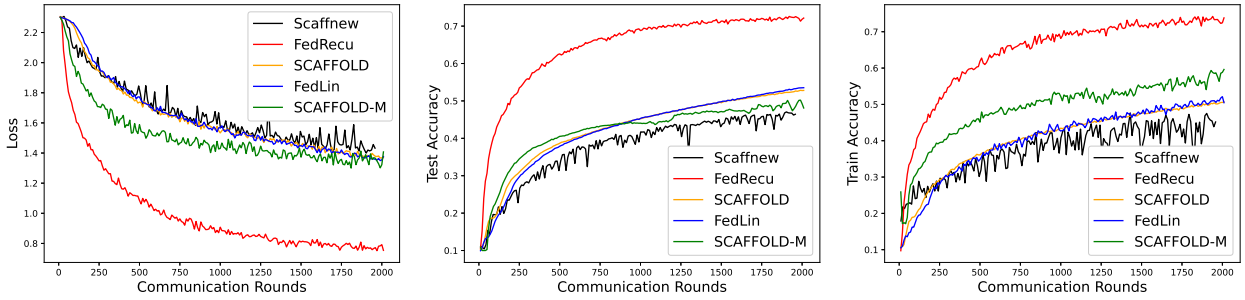
(a) (b) (c)

Figure 10: Comparison of FedRecu with EXTRA on the CIFAR-10 dataset. Each curve represents the average of six independent runs. To induce greater heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 10$.

A.4 Comparison of FedRecu, SCAFFOLD, SCAFFOLD-M, Scaffnew, and FedLin for CNN Training

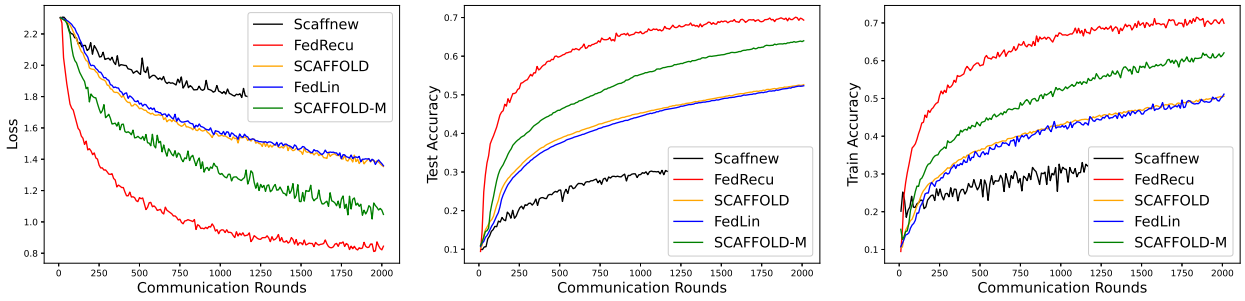
Additional CNN training experiments on the CIFAR-10 dataset are presented in Figures 11, 12, and 13, respectively, using non-IID data distributions generated by Dirichlet partitions with heterogeneity parameters $\beta = 0.1, 1, 10$, respectively. The stepsizes for FedRecu, SCAFFOLD, SCAFFOLD-M, FedLin, and Scaffnew are selected according to the guidelines from Theorem 1, Karimireddy et al. (2020), Cheng et al. (2023), Mitra et al. (2021b), and Mishchenko et al. (2022), respectively, using an estimated smoothness parameter of $L = 2$. For FedRecu, SCAFFOLD, SCAFFOLD-M, and FedLin, the local training period is set to $\tau = 10$.

For Scaffnew, the communication probability is set to $\frac{1}{11}$ to ensure that the total number of communicated messages remains consistent across methods. These experiments in Figures 11, 12, and 13, show that our algorithm can achieve faster convergence than the competing methods.



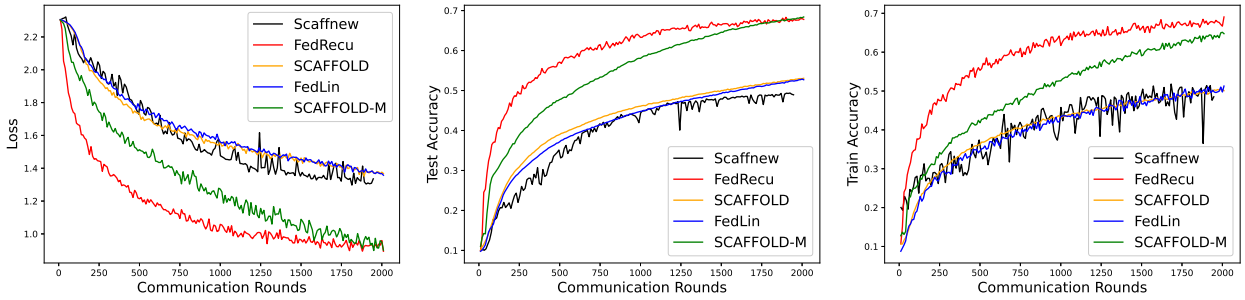
(a) (b) (c)

Figure 11: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, SCAFFOLD-M, FedLin, and Scaffnew—on the CIFAR-10 dataset. Each curve represents the average of six independent runs. Note that the test accuracy in Figure 11(b) is top-5 accuracy. To induce greater heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 0.1$.



(a) (b) (c)

Figure 12: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, SCAFFOLD-M, FedLin, and Scaffnew—on the CIFAR-10 dataset. Each curve represents the average of six independent runs. Note that the test accuracy in Figure 12(b) is top-5 accuracy. To induce greater heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 1$.



(a) (b) (c)

Figure 13: Comparison of FedRecu with state-of-the-art federated learning algorithms—SCAFFOLD, SCAFFOLD-M, FedLin, and Scaffnew—on the CIFAR-10 dataset. Each curve represents the average of six independent runs. Note that the test accuracy in Figure 13(b) is top-5 accuracy. To induce greater heterogeneity in data distribution, the Dirichlet distribution parameter was set to $\beta = 10$.

A.5 Additional evaluation results using least squares regression

We also used the following least squares regression problem to evaluate the convergence performance of the proposed algorithm[2]:

$$\min_{x\in\mathbb{R}^n} f(x) = \min_{x\in\mathbb{R}^n} \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|A_i x - b_i\|^2, \tag{9}$$

where $A_i \in \mathbb{R}^{50\times 10}$, $b_i \in \mathbb{R}^{50}$, $x \in \mathbb{R}^{10}$, and n is set to 20. We consider $[A_i]_{jk}$ and $[b_i]_j$ generated from $[0, 1]$ randomly for $1 \le j \le 50$ and $1 \le k \le 10$.

We compare our algorithm with existing federated-learning algorithms that can tackle "client drifts," including SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), and FedTrack (Mitra et al., 2021a). The stepsizes for FedRecu, SCAFFOLD, FedLin, and FedTrack are selected based on the guidelines provided in Theorem 1, Karimireddy et al. (2020), Mitra et al. (2021b), and Mitra et al. (2021a), respectively. We use the convergence error $f(x(k\tau)) - f(x^*)$, where $x(k\tau) = \frac{1}{N}\sum_{i=1}^{N} x_i(k\tau)$, to quantify the learning accuracy of each algorithm. We implement all algorithms using accurate gradients for fairness. Figure 14 illustrates the convergence errors of all algorithms under different local training periods $\tau = 4, 8, 12, 16$, respectively. These numerical results clearly confirm that FedRecu achieves much faster convergence than SCAFFOLD, FedLin, and FedTrack across all tested settings despite its reduced overhead in storage.

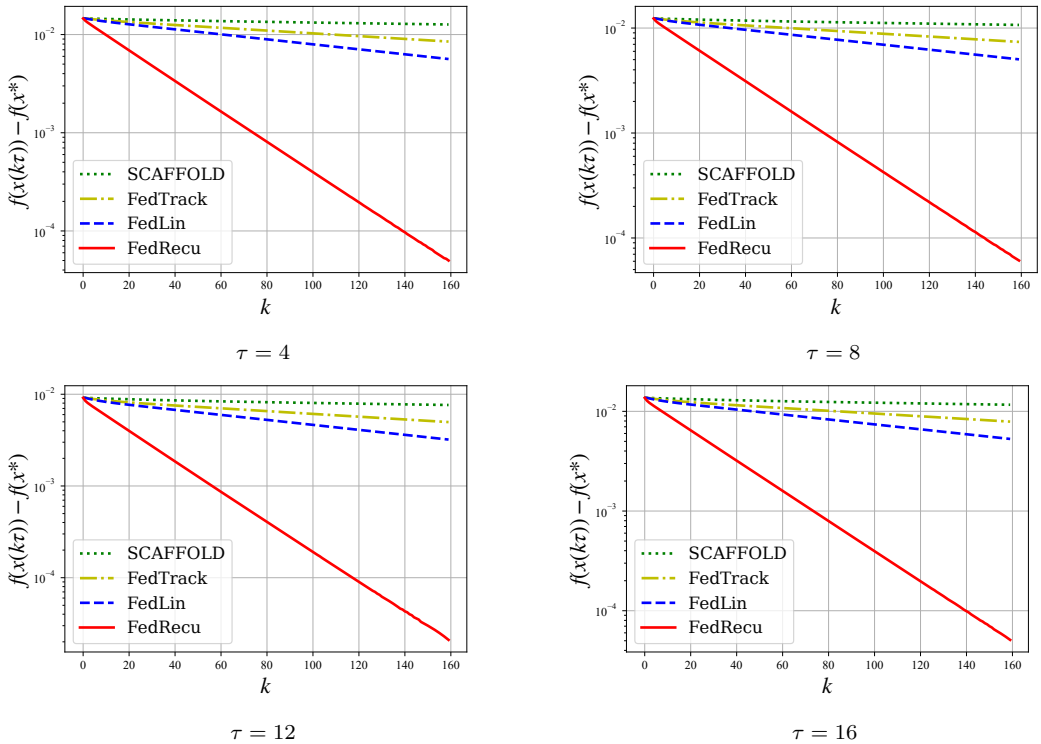


Figure 14: Comparisons of FedRecu with SCAFFOLD (Karimireddy et al., 2020), FedLin (Mitra et al., 2021b), and FedTrack (Mitra et al., 2021a) under different local training periods τ.

B Supporting Lemmas For the Proof of Theorem 1

Lemma 1 (Lee & Wright (2019)). *Let $\{\Delta(t)\}$ be a nonnegative sequence satisfying the following conditions:*

(1) $\{\Delta(t)\}$ is monotonically decreasing;

[2]Code available at https://anonymous.4open.science/r/fedrecu-E043/README.md

(2) $\{\Delta(t)\}$ *is summable, that is,* $\sum_{k=0}^{\infty}\Delta(k)<\infty$.

Then, we have $\Delta(t)=o(1/t)$*, i.e.,* $\lim_{t\to\infty}t\Delta(t)=0$.

Lemma 2. *Algorithm 1 can be equivalently expressed in the following matrix form:*

$$\begin{cases} x_i(k\tau+j)=x_i(k\tau+j-1)-\alpha y_i(k\tau+j-1),\\ y_i(k\tau+j)=y_i(k\tau+j-1)+\nabla f_i(x_i(k\tau+j))-\nabla f_i(x_i(k\tau+j-1)). \end{cases} \tag{10}$$

for any $j=1,2,\cdots,\tau-1$.

Proof. For the convenience of expression, we define $X(t)=[x_1^{\mathbf{T}}(t),x_2^{\mathbf{T}}(t),\cdots,x_N^{\mathbf{T}}(t)]^{\mathbf{T}}$, $\nabla f(t)=[f_1(x_1^{\mathbf{T}}(t)),f_2(x_2^{\mathbf{T}}(t)),\cdots,f_N(x_N^{\mathbf{T}}(t))]^{\mathbf{T}}$, and

$$W(t+1)=\begin{cases}\frac{1}{N}\mathbf{1}_N\mathbf{1}_N^{\mathbf{T}}, & t+1=\tau k,\\ \mathbf{I}, & t+1\neq\tau k.\end{cases} \tag{11}$$

Thus, Algorithm 1 can be rewritten as the following matrix form:

$$\begin{aligned} X(t+1)=&2W(t+1)X(t)-W(t+1)W(t)X(t-1)-\alpha W(t+1)W(t)\nabla f(t)\\ &+\alpha W(t+1)W(t)\nabla f(t-1). \end{aligned} \tag{12}$$

Rearranging the terms of (12), we arrive at

$$\begin{aligned} \frac{1}{\alpha}\Big\{W(t+1)X(t)-X(t+1)\Big\}=&\frac{1}{\alpha}W(t+1)\Big\{W(t)X(t-1)-X(t)\Big\}\\ &+W(t+1)\Big\{W(t)\{\nabla f(t)-\nabla f(t-1)\}\Big\}. \end{aligned} \tag{13}$$

Defining

$$Z(t)=\frac{1}{\alpha}\Big\{W(t+1)X(t)-X(t+1)\Big\},$$

we can obtain

$$Z(t)=W(t+1)Z(t-1)+W(t+1)\Big\{W(t)\{\nabla f(t)-\nabla f(t-1)\}\Big\}, \tag{14}$$

based on the structure of (13).

From (11) and (12), we can construct $Y(t)$ satisfying

$$Z(t)=W(t+1)Y(t).$$

Thus, from (13) and (14), we have

$$W(t+1)Y(t)=W(t+1)W(t)Y(t-1)+W(t+1)\Big\{W(t)\{\nabla f(t)-\nabla f(t-1)\}\Big\}.$$

From the definition (11) of $W(t)$, we have

$$\begin{aligned} X(t)=&W(t)X(t-1)-\alpha W(t)Y(t-1),\\ Y(t)=&W(t)Y(t-1)+W(t)\{\nabla f(t)-\nabla f(t-1)\}, \end{aligned}$$

where $Y(t)=[y_1^{\mathbf{T}}(t),y_2^{\mathbf{T}}(t),\cdots,y_N^{\mathbf{T}}(t)]^{\mathbf{T}}$, which completes the proof. □

Lemma 3 (Mitra et al. (2021b))**.** *Suppose* $f_i(x)$ *is L-smooth and convex. Then, for any* $0\leq\alpha\leq\frac{1}{L}$*, we have*

$$\|y-x-\alpha(\nabla f_i(y)-\nabla f_i(x))\|\leq\|y-x\|$$

for any $x,y\in\mathbb{R}^n$.

Lemma 4. *For any $k \geq 0$, we have*

$$y_i(k\tau) = \nabla f(x_i(k\tau)) \tag{15}$$

for any $i \in \mathcal{S}$.

Proof. We use mathematical induction to prove Lemma 4.

It is clear that the relation holds for $k = 0$.

Next, we assume that the relation holds at time instant k, i.e.,

$$y_i(k\tau) = \nabla f(x_i(k\tau)) = \frac{1}{N}\sum_{j=1}^{N} \nabla f_j(x_j(k\tau)), \tag{16}$$

and prove that the relation also hold at time instant $k+1$.

Using Lemma 2, we can obtain the following relation based on (16):

$$y_i(k\tau+1) = y_i(k\tau) + \nabla f_i(x_i(k\tau+1)) - \nabla f_i(x_i(k\tau)). \tag{17}$$

Similarly, for any $j = 1, 2, \cdots, \tau-1$, we can obtain

$$y_i(k\tau+j) = y_i(k\tau) + \nabla f_i(x_i(k\tau+j)) - \nabla f_i(x_i(k\tau)).$$

Thus, we have

$$y_i(k\tau+\tau-1) = y_i(k\tau) + \nabla f_i(x_i(k\tau+\tau-1)) - \nabla f_i(x_i(k\tau)). \tag{18}$$

Using Lemma 2 leads to

$$y_i(k\tau+\tau) = \frac{1}{N}\sum_{j=1}^{N} \Big\{ y_j(k\tau+\tau-1) + \nabla f_j(x_j(k\tau+\tau)) - \nabla f_j(x_j(k\tau+\tau-1)) \Big\}.$$

Therefore, we have

$$\begin{aligned} y_i(k\tau+\tau) &= \frac{1}{N}\sum_{j=1}^{N} \Big\{ y_j(k\tau) + \nabla f_j(x_j(k\tau+\tau)) - \nabla f_j(x_j(k\tau)) \Big\} \\ &= \frac{1}{N}\sum_{j=1}^{N} \nabla f_j(x_j(k\tau+\tau)), \end{aligned}$$

where the first and second equalities follow from (16) and (18), respectively. Namely, the relation in the lemma statement also holds at time instant $k+1$, which completes the proof. □

Lemma 5. *For $0 \leq j < \tau$ and $0 < \alpha L \leq 1$, we have*

$$\|x_i(k\tau+j) - x_i(k\tau)\| \leq j\alpha \|y_i(x_i(k\tau))\|.$$

Proof. From Lemma 2, we obtain

$$\begin{aligned} &\|x_i(k\tau+j+1) - x_i(k\tau)\| \\ =&\|x_i(k\tau+j) - x_i(k\tau) - \alpha\{y_i(k\tau) + \nabla f_i(x_i(k\tau+j)) - \nabla f_i(x_i(k\tau))\}\|. \end{aligned}$$

From the above equation and Lemma 3, we have

$$\|x_i(k\tau+j+1) - x_i(k\tau)\| \leq \|x_i(k\tau+j) - x_i(k\tau)\| + \alpha\|y_i(k\tau)\|,$$

which further implies

$$\|x_i(k\tau+j) - x_i(k\tau)\| \leq j\alpha\|y_i(k\tau)\|.$$

for any $0 \leq j < \tau$. □

C Proof of Theorem 1

From Lemma 2, we have

$$x_i(k\tau+\tau) = \frac{1}{N}\sum_{i=1}^{N}\Big\{x_i(k\tau+\tau-1) - \alpha y_i(k\tau+\tau-1)\Big\}, \tag{19}$$

$$y_i(k\tau+\tau) = \frac{1}{N}\sum_{i=1}^{N}\Big\{y_i(k\tau+\tau-1) + \nabla f_i(x_i(k\tau+\tau)) - \nabla f_i(x_i(k\tau+\tau-1))\Big\}. \tag{20}$$

Thus, (10) implies

$$x_i(k\tau+\tau-1) = x_i(k\tau) - \alpha\sum_{h=0}^{\tau-2}\nabla y_i(k\tau+h).$$

From the above equation and (19), we have

$$x_i(k\tau+k) = \frac{1}{N}\sum_{j=1}^{N}\Big\{x_j(k\tau) - \alpha\sum_{h=0}^{\tau-1} y_j(k\tau+h)\Big\}, \tag{21}$$

which, in combination with (20) yields

$$x_i(k\tau+k) = \frac{1}{N}\sum_{j=1}^{N}\Big\{x_j(k\tau) - \alpha\sum_{h=0}^{\tau-1}\{y_j(k\tau) + \nabla f_j(x_j(k\tau+h)) - \nabla f_j(x_j(k\tau))\}\Big\}. \tag{22}$$

Lemma 4 and (22) imply

$$x_i(k\tau+k) = x_i(k\tau) - \frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)). \tag{23}$$

From (23), we have

$$\begin{aligned}
&\|x_i(k\tau+k) - x^*\|^2 - \|x_i(k\tau) - x^*\|^2 \\
=& -2\langle\frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)), x_i(k\tau) - x^*\rangle + \|\frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h))\|^2.
\end{aligned}$$

Then, using Assumption 1 and the convex property of $f_i(x)$, we can obtain

$$\begin{aligned}
&-2\langle\frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)), x_i(k\tau) - x^*\rangle \\
\le&\frac{2\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{f_j(x^*) - f_j(x_j(k\tau+h))\Big\} \\
&+\frac{2\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{f_j(x_j(k\tau+h)) - f_j(x_j(k\tau)) + \frac{L}{2}\|x_j(k\tau+h) - x_j(k\tau)\|^2\Big\},
\end{aligned}$$

which further implies

$$\begin{aligned}
&-2\langle\frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)),x_i(k\tau)-x^*\rangle\\
\leq&\frac{2\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{f_j(x^*)-f_j(x_j(k\tau+h))\Big\}+\frac{2\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{f_j(x_j(k\tau+h))-f_j(x_j(k\tau))\Big\}\\
&+\frac{\alpha L}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\|x_i(k\tau+h)-x_i(k\tau)\|^2.
\end{aligned}$$

Combining Lemma 4 and Lemma 5, we arrive at

$$\begin{aligned}
&-2\langle\frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)),x_i(k\tau)-x^*\rangle\\
\leq&2\alpha\tau\Big\{f(x^*)-f(x_i(k\tau))\Big\}+L\alpha^3\Big\{\sum_{h=0}^{\tau-1}h^2\Big\}\|\nabla f(x_j(k\tau))\|^2.
\end{aligned}$$

which further implies the following relationship based on Assumption 1:

$$\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h))\|\leq\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}L\|x_j(k\tau+h)-x_j(k\tau)\|+\tau\|\nabla f(x_j(k\tau))\|.$$

The above inequality, Lemma 5, and the condition $0<\alpha\tau L\leq 1$ imply

$$\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h))\|\leq\frac{3\tau-1}{2}\|\nabla f(x_j(k\tau))\|. \tag{24}$$

Thus, we can obtain

$$\|x_i(k\tau+k)-x^*\|^2-\|x_i(k\tau)-x^*\|^2\leq 2\alpha\tau\{f(x^*)-f(x_i(k\tau))\}+A_1\|\nabla f(x_j(k\tau))\|^2, \tag{25}$$

where $A_1=L\alpha^3\{\sum_{h=0}^{\tau-1}h^2\}+(\frac{3\tau-1}{2})^2$.

From (23) and Assumption 1, we have

$$\begin{aligned}
&f(x_i(k\tau+\tau))\\
\leq&f(x_i(k\tau))-\alpha\langle\nabla f(x_i(k\tau)),\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h))\rangle\\
&+\frac{\alpha^2L}{2}\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h))\|^2.
\end{aligned}$$

From 24, Lemma 4, and Lemma 5, we have

$$\begin{aligned}
&f(x_i(k\tau+\tau))\\
\leq&f(x_i(k\tau))-\alpha\tau\|\nabla f(x_i(k\tau))\|^2+\frac{\alpha^2L(3\tau-1)^2}{8}\|\nabla f(x_j(k\tau))\|^2\\
&+\frac{\alpha^2L\tau(\tau-1)}{2}\|\nabla f(x_i(k\tau))\|^2,
\end{aligned}$$

which further implies

$$f(x_i(k\tau+\tau))\leq f(x_i(k\tau))+\Big\{\frac{\alpha^2L(3\tau-1)^2}{8}+\frac{\alpha^2L\tau(\tau-1)}{2}-\alpha\tau\Big\}\|\nabla f(x_i(k\tau))\|^2.$$

Thus, if $0 < \alpha < \frac{8}{13L\tau}$ holds, we have

$$\frac{\alpha^2 L(3\tau-1)^2}{8} + \frac{\alpha^2 L\tau(\tau-1)}{2} - \alpha\tau < 0.$$

Combining the preceding two relations yields that there exists $\gamma > 0$ such that

$$\gamma\|\nabla f(x_i(k\tau))\|^2 \leq f(x_i(k\tau+\tau)) - f(x_i(k\tau)) \tag{26}$$

holds under $0 < \alpha \leq \frac{8}{13L\tau}$.

From (25), we have

$$\begin{aligned} 2\alpha\tau\{f(x_i(k\tau)) - f(x^*)\} \leq & \|x_i(k\tau) - x^*\|^2 - \|x_i(k\tau+k) - x^*\|^2 \\ & + \frac{A_1}{\gamma}\{f(x_i(k\tau+\tau)) - f(x_i(k\tau))\} \end{aligned}$$

for any $i \in \mathcal{S}$. Thus, we obtain

$$\sum_{k=1}^{\infty}\{f(x_i(k\tau)) - f(x^*)\} < \infty. \tag{27}$$

From (26), (27), and Lemma 1, we have

$$\lim_{k\to\infty} k\{f(x_i(k\tau)) - f(x^*)\} = 0,$$

which completes the proof.

D Supporting Lemmas for the Proof of Theorem 2

Lemma 6. *For $0 < \alpha \leq \frac{1}{2\tau L}$, we have*

$$\|x_i(k\tau+j) - x_i(k\tau)\| \leq 2j\alpha\|y_i(x_i(k\tau))\| \tag{28}$$

for any $0 \leq j < \tau$.

Proof. From (31), we can obtain

$$\begin{aligned} & \|x_i(k\tau+j+1) - x_i(k\tau)\| \\ =& \|x_i(k\tau+j) - x_i(k\tau) - \alpha\{y_i(k\tau) + \nabla f_i(x_i(k\tau+j)) - \nabla f_i(x_i(k\tau))\}\|. \end{aligned} \tag{29}$$

Combining (29) and Assumption 1 implies

$$\|x_i(k\tau+j+1) - x_i(k\tau)\| \leq (1+\alpha L)\|x_i(k\tau+j) - x_i(k\tau)\| + \alpha\|y_i(k\tau)\|. \tag{30}$$

Next we use mathematical induction to prove the lemma.

We first assume that (28) holds for $0 \leq j \leq \tau - 1$:

$$\|x_i(k\tau+j) - x_i(k\tau)\| \leq 2j\alpha\|y_i(k\tau)\|$$

for any $i \in \mathcal{S}$ and $0 \leq j \leq \tau - 1$. Then, we will prove that (28) also holds for $j+1$.

Combining the above inequality and (30) implies

$$\|x_i(k\tau+j+1) - x_i(k\tau)\| \leq (2j + 2\alpha Lj + 1)\alpha\|y_i(k\tau)\|.$$

If a stepsize satisfies $0 < \alpha \leq \frac{1}{2\tau L}$, we have

$$\|x_i(k\tau+j+1) - x_i(k\tau)\| \leq (2j+2)\alpha\|y_i(k\tau)\|,$$

which completes the proof. □

E Proof of Theorem 2

From Appendix C, for any $j = 1, 2, \cdots, \tau - 1$, we have

$$\begin{cases} x_i(k\tau + j) = x_i(k\tau + j - 1) - \alpha y_i(k\tau + j - 1), \\ y_i(k\tau + j) = y_i(k\tau + j - 1) + \nabla f_i(x_i(k\tau + j)) - \nabla f_i(x_i(k\tau + j - 1)) \\ y_i(k\tau + j) = y_i(k\tau) + \nabla f_i(x_i(k\tau + j)) - \nabla f_i(x_i(k\tau)). \end{cases} \tag{31}$$

Moreover, from Lemma 4 we have

$$y_i(k\tau) = \nabla f(x_i(k\tau)) \tag{32}$$

for any $i \in \mathcal{S}$.

Combining (31) and (32) implies

$$x_i(k\tau + k) = x_i(k\tau) - \frac{\alpha}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \nabla f_j(x_j(k\tau + h)) \tag{33}$$

for any $i \in \mathcal{S}$.

From Assumption 1, we have

$$\|\frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \nabla f_j(x_j(k\tau + h))\| \leq \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} L\|x_j(k\tau + h) - x_j(k\tau)\| + \tau\|\nabla f(x_j(k\tau))\|.$$

From (32), Lemma 6, and $0 < \alpha \leq \frac{1}{2\tau L}$, we have

$$\|\frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \nabla f_j(x_j(k\tau + h))\| \leq \frac{3\tau - 1}{2} \|\nabla f(x_j(k\tau))\|. \tag{34}$$

Combining Assumption 1 and (33) implies

$$\begin{aligned} f(x_i(k\tau + \tau)) \leq & f(x_i(k\tau)) - \alpha \langle \nabla f(x_i(k\tau)), \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \nabla f_j(x_j(k\tau + h)) \rangle \\ & + \frac{\alpha^2 L}{2} \|\frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \nabla f_j(x_j(k\tau + h))\|^2. \end{aligned}$$

From the above inequality and Asumption 1, we arrive at

$$\begin{aligned} f(x_i(k\tau + \tau)) \leq & f(x_i(k\tau)) - \alpha\tau \|\nabla f(x_i(k\tau))\|^2 + \frac{\alpha^2 L(3\tau - 1)^2}{8} \|\nabla f(x_j(k\tau))\|^2 \\ & + \alpha^2 L \|\nabla f(x_i(k\tau))\|^2 \sum_{h=0}^{\tau-1} 2h. \end{aligned}$$

The above inequality and (34) imply

$$f(x_i(k\tau + \tau)) \leq f(x_i(k\tau)) + \Big\{ \frac{\alpha^2 L(3\tau - 1)^2}{8} + \alpha^2 L\tau(\tau - 1) - \alpha\tau \Big\} \|\nabla f(x_i(k\tau))\|^2.$$

If the stepsize satisfies $0 < \alpha \leq \frac{8}{17L\tau}$, we can obtain

$$\frac{\alpha^2 L(3\tau)^2}{8} + \alpha^2 L\tau(\tau) - \alpha\tau \leq 0,$$

i.e.,

$$\alpha\tau - \frac{\alpha^2 L(3\tau-1)^2}{8} - \alpha^2 L\tau(\tau-1) \geq \gamma > 0,$$

where $\gamma = \frac{13\alpha^2 L\tau}{8}$.

Thus, combining the preceding three relations yields

$$\gamma\|\nabla f(x_i(k\tau))\|^2 \leq f(x_i(k\tau)) - f(x_i(k\tau+\tau)),$$

and hence

$$\frac{1}{K}\sum_{k=0}^{K-1}\|\nabla f(x_i(k\tau))\|^2 \leq \frac{f(x_i(0)) - f(x_i(K\tau))}{\gamma K} \leq \frac{f(x_i(0)) - f(x^*)}{\gamma K},$$

for any $i \in \mathcal{S}$.

F Supporting Lemmas for the Proof of Theorem 3

Lemma 7. *Under Assumption 1 and Assumption 2, if the loss function $f_i(x)$ of client $i \in \mathcal{S}$ is convex, we have*

$$\mathbb{E}\Big[\|x_i(k\tau+h) - x_i(k\tau)\|^2\Big] \leq 12\tau^2 L\alpha^2 \mathbb{E}\Big[f(x_i(k\tau)) - f(x^*)\Big] + 27\tau\alpha^2\sigma^2$$

for $0 \leq h < \tau$ and $i \in \mathcal{S}$.

Proof. From (39) and (40), we have

$$x_i(k\tau+j+1) = x_i(k\tau+j) - \alpha\Big\{\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - g_i(x_i(k\tau)) + g_j(x_j(k\tau+j))\Big\},$$

i.e.,

$$\begin{aligned}
&x_i(k\tau+j+1) - x_i(k\tau)\\
=&x_i(k\tau+j) - x_i(k\tau) - \alpha\Big\{\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) - \nabla f_i(x_i(k\tau)) + \nabla f_j(x_j(k\tau+j))\Big\}\\
&- \alpha\Big\{\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) + \nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau))\\
&\quad + g_j(x_j(k\tau+j)) - \nabla f_j(x_j(k\tau+j))\Big\}.
\end{aligned}$$

Further using Assumption 3 yields

$$\begin{aligned}
&\mathbb{E}\Big[\|x_i(k\tau+j+1) - x_i(k\tau)\|^2\Big]\\
=&\mathbb{E}\Big[\Big\|x_i(k\tau+j) - x_i(k\tau) - \alpha\Big(\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) - \nabla f_i(x_i(k\tau)) + \nabla f_j(x_j(k\tau+j))\Big)\Big\|^2\Big]\\
&+ \alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) + \nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau))\\
&\quad + g_j(x_j(k\tau+j)) - \nabla f_j(x_j(k\tau+j))\Big\|^2\Big].
\end{aligned} \tag{35}$$

For the first term of the right hand side of (35), using the inequality

$$\|a+b\|^2 \leq (1+\epsilon)\|a\|^2 + (1+\frac{1}{\epsilon})\|b\|^2$$

and Lemma 3 yields

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|x_i(k\tau+j) - x_i(k\tau) - \alpha\Big(\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) - \nabla f_i(x_i(k\tau)) + \nabla f_j(x_j(k\tau+j))\Big)\Big\|^2\Big] \\
\leq &(1+\frac{1}{\epsilon})\mathbb{E}[\|x_i(k\tau+j) - x_i(k\tau)\|^2] + (1+\epsilon)\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2], \qquad (36)
\end{aligned}$$

for any $\epsilon > 0$ under a stepsize satisfying $0 < \alpha L \leq 1$.

For the second term of the right hand side of (35), we have

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) + \nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau)) \\
&\qquad + g_j(x_j(k\tau+j)) - \nabla f_j(x_j(k\tau+j))\Big\|^2\Big] \\
\leq &\frac{3}{N}\sum_{j=1}^{N}\mathbb{E}[\|g_j(x_j(k\tau)) - \nabla f_j(x_j(k\tau))\|^2\} + 3\mathbb{E}[\|\nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau))\|^2] \\
&+ 3\mathbb{E}[\|g_j(x_j(k\tau+j)) - \nabla f_j(x_j(k\tau+j))\|^2], \qquad (37)
\end{aligned}$$

for any $i \in \mathcal{S}$.

Using Assumption 3 and (37) leads to

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) + \nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau)) \\
&\qquad + g_j(x_j(k\tau+j)) - \nabla f_j(x_j(k\tau+j))\Big\|^2\Big] \leq 9\sigma^2, \qquad (38)
\end{aligned}$$

for any $i \in \mathcal{S}$.

Combining (35), (36), and (38), we can obtain

$$\begin{aligned}
&\mathbb{E}[\|x_i(k\tau+j+1) - x_i(k\tau)\|^2] \\
\leq &(1+\frac{1}{\epsilon})\mathbb{E}[\|x_i(k\tau+j) - x_i(k\tau)\|^2] + (1+\epsilon)\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2] + 9\alpha^2\sigma^2.
\end{aligned}$$

Moreover, for any $0 \leq j < \tau$, we can obtain

$$\mathbb{E}\Big[\|x_i(k\tau+j) - x_i(k\tau)\|^2\Big] \leq \Big\{(1+\epsilon)\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2] + 9\alpha^2\sigma^2\Big\}\frac{(1+\frac{1}{\epsilon})^j - 1}{(1+\frac{1}{\epsilon}) - 1}.$$

Selecting $\epsilon = \tau$ implies

$$\mathbb{E}\Big[\|x_i(k\tau+j) - x_i(k\tau)\|^2\Big] \leq 12\tau^2 L\alpha^2\mathbb{E}[f(x_j(k\tau)) - f(x^*)] + 27\tau\alpha^2\sigma^2,$$

since we have $\|\nabla f(x_j(k\tau))\|^2 \leq 2L(f(x_j(k\tau)) - f(x^*))$ from Assumption 1 and the convex property of $f_i(x)$ for any $i \in \mathcal{S}$. The proof of Lemma 7 is complete. □

G Proof of Theorem 3

For the convenience of expression, we use $g_i(x_i(t))$ to denote $\nabla f_i(x_i(t), \xi_i(t))$ for any $i \in \mathcal{S}$ and $t \geq 0$.

From Appendix C, for any $j = 1, 2, \cdots, \tau - 1$, we have

$$\begin{cases} x_i(k\tau + j) = x_i(k\tau + j - 1) - \alpha y_i(k\tau + j - 1), \\ y_i(k\tau + j) = y_i(k\tau + j - 1) + g_i(x_i(k\tau + j)) - g_i(x_i(k\tau + j - 1)) \\ y_i(k\tau + j) = y_i(k\tau) + g_i(x_i(k\tau + j)) - g_i(x_i(k\tau)). \end{cases} \tag{39}$$

Moreover, similar to the derivation of Lemma 4, we have

$$y_i(k\tau) = \frac{1}{N} \sum_{j=1}^{N} g_j(x_j(k\tau)) \tag{40}$$

for any $i \in \mathcal{S}$.

Combining (39) and (40), we have

$$x_i(k\tau + k) = x_i(k\tau) - \frac{\alpha}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} g_j(x_j(k\tau + h))$$

and further

$$\begin{aligned} &\|x_i(k\tau + k) - x^*\|^2 - \|x_i(k\tau) - x^*\|^2 \\ =& -2\alpha \langle \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} g_j(x_j(k\tau + h)), x_i(k\tau) - x^* \rangle + \alpha^2 \| \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} g_j(x_j(k\tau + h)) \|^2. \end{aligned} \tag{41}$$

For the term $-2\alpha \langle \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} g_j(x_j(k\tau + h)), x_i(k\tau) - x^* \rangle$, we have

$$\begin{aligned} &-2\alpha \mathbb{E}\Big[\langle \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} g_j(x_j(k\tau + h)), x_i(k\tau) - x^* \rangle \Big] \\ =& \frac{2\alpha}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \mathbb{E}\Big[\langle x^* - x_j(k\tau + h), \nabla f_j(x_j(k\tau + h)) \rangle \Big] \\ &+ \frac{2\alpha}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \mathbb{E}\Big[\langle x_j(k\tau + h) - x_i(k\tau), \nabla f_j(x_j(k\tau + h)) \rangle \Big] \end{aligned}$$

From Assumption 1 and the convexity of $f_i(x)$, we can obtain

$$\begin{aligned} &-2\alpha \mathbb{E}\Big[\langle \frac{1}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} g_j(x_j(k\tau + h)), x_i(k\tau) - x^* \rangle \Big] \\ \leq& \frac{2\alpha}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \mathbb{E}\Big[f_j(x^*) - f_j(x_j(k\tau + h)) \Big] \\ &+ \frac{2\alpha}{N} \sum_{j=1}^{N} \sum_{h=0}^{\tau-1} \mathbb{E}\Big[f_j(x_j(k\tau + h)) - f_j(x_i(k\tau)) + \frac{L}{2} \|x_j(k\tau + h) - x_i(k\tau)\|^2 \Big], \end{aligned}$$

which further implies

$$\begin{aligned}
&-2\alpha\mathbb{E}\Big[\langle\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h)),x_i(k\tau)-x^*\rangle\Big]\\
\leq&2\alpha\tau\mathbb{E}\Big[f(x^*)-f(x_i(k\tau))\Big]+\frac{\alpha L}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}\Big[\|x_j(k\tau+h)-x_i(k\tau)\|^2\Big].
\end{aligned}$$

From Lemma 7 we have

$$\mathbb{E}\Big[\|x_i(k\tau+h)-x_i(k\tau)\|^2\Big]\leq 12\tau^2L\alpha^2\mathbb{E}\Big[f(x_j(k\tau))-f(x^*)\Big]+27\tau\alpha^2\sigma^2$$

for $1\leq h<\tau$.

Combining the preceding two inequalities leads to

$$\begin{aligned}
&-2\alpha\mathbb{E}\Big[\langle\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h)),x_i(k\tau)-x^*\rangle\Big]\\
\leq&2\alpha\tau\mathbb{E}\Big[f(x^*)-f(x_i(k\tau))\Big]+12\tau^3L^2\alpha^3\mathbb{E}\Big[f(x_j(k\tau))-f(x^*)\Big]+27\tau^2L\alpha^3\sigma^2.
\end{aligned}$$

If the stepsize satisfies $0<6\tau\alpha L\leq 1$, we have

$$\begin{aligned}
&-2\alpha\mathbb{E}\Big[\langle\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h)),x_i(k\tau)-x^*\rangle\Big]\\
\leq&2\alpha\tau\mathbb{E}\Big[f(x^*)-f(x_i(k\tau))\Big]+2\tau^2L\alpha^2\mathbb{E}\Big[f(x_j(k\tau))-f(x^*)\Big]+9\tau^2\alpha^2\sigma^2.
\end{aligned}\tag{42}$$

For the term $\alpha^2\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\|^2$ in (41), we have

$$\begin{aligned}
&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\\
\leq&2\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2+2\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\Big\|^2.
\end{aligned}\tag{43}$$

Using the inequality $\|a+b+c\|^2\leq 3\|a\|^2+3\|b\|^2+3\|c\|^2$, we have

$$\begin{aligned}
&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2\\
\leq&\frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|\nabla f_j(x_j(k\tau+h))-\nabla f_j(x_j(k\tau))\Big\|^2+\frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|g_j(x_j(k\tau+h))-\nabla f_j(x_j(k\tau+h))\Big\|^2\\
&+\frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|\nabla f_j(x_j(k\tau))-g_j(x_j(k\tau))\Big\|^2.
\end{aligned}\tag{44}$$

Using Assumption 1, we have the following inequality from (44):

$$\begin{aligned}
&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2\\
\leq&\frac{3\tau L^2\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|x_j(k\tau+h)-x_j(k\tau)\Big\|^2+\frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|g_j(x_j(k\tau+h))-\nabla f_j(x_j(k\tau+h))\Big\|^2\\
&+\frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|\nabla f_j(x_j(k\tau))-g_j(x_j(k\tau))\Big\|^2. \qquad (45)
\end{aligned}$$

Combining (45) and Assumption 3, we arrive at

$$\begin{aligned}
&2\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2\Big]\\
\leq&\frac{6\tau\alpha^2L^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}\Big[\Big\|x_j(k\tau+h)-x_j(k\tau)\Big\|^2\Big]+12\alpha^2\tau^2\sigma^2. \qquad (46)
\end{aligned}$$

Lemma 7 and (46) imply

$$\begin{aligned}
&2\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2\Big]\\
\leq&72\tau^4L^3\alpha^4\mathbb{E}[f(x_j(k\tau))-f(x^*)]+162\tau^3L^2\alpha^4\sigma^2+12\alpha^2\tau^2\sigma^2. \qquad (47)
\end{aligned}$$

For the term $\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\|^2$ in (43), we have

$$2\alpha^2\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\|^2\leq\frac{4\alpha^2\tau^2}{N}\sum_{j=1}^{N}\|g_j(x_j(k\tau))-\nabla f_j(x_j(k\tau))\|^2+4\alpha^2\tau^2\|\nabla f(x_j(k\tau))\|^2.$$

Thus, from Lemma 7 and Assumption 3, we have

$$2\alpha^2\mathbb{E}[\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\|^2]\leq4\alpha^2\tau^2\sigma^2+8\alpha^2\tau^2L\mathbb{E}[f(x_j(k\tau))-f(x^*)]. \qquad (48)$$

Combining (43), (47), and (48), we have

$$\begin{aligned}
&\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\Big]\\
\leq&(72\tau^4L^3\alpha^4+8\alpha^2\tau^2L)\mathbb{E}[f(x_j(k\tau))-f(x^*)]+162\tau^3L^2\alpha^4\sigma^2+16\alpha^2\tau^2\sigma^2.
\end{aligned}$$

If the stepsize satisfies $0<6\tau\alpha L\leq1$, then we have

$$\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\Big]\leq10\tau^2L\alpha^2\mathbb{E}[f(x_j(k\tau))-f(x^*)]+25\alpha^2\tau^2\sigma^2. \qquad (49)$$

Combining (41), (42), and (49), we have

$$\mathbb{E}\Big[\|x_i(k\tau+k)-x^*\|^2\Big]-\mathbb{E}\Big[\|x_i(k\tau)-x^*\|^2\Big]\leq(2\alpha\tau-12\tau^2L\alpha^2)\mathbb{E}\Big[f(x^*)-f(x_i(k\tau))\Big]+34\tau^2\alpha^2\sigma^2.$$

If the stepsize satisfies $0 < \alpha < \frac{1}{6\tau L}$, we have

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\Big[f(x_i(k\tau)) - f(x^*)\Big] \leq \frac{\mathbb{E}[\|x_i(0) - x^*\|^2]}{(2\alpha\tau - 12\tau^2 L\alpha^2)K} + A\sigma^2,$$

where

$$A = \frac{34\tau^2\alpha^2}{2\alpha\tau - 12\tau^2 L\alpha^2}.$$

Thus, we have

$$\mathbb{E}\Big[f(\frac{1}{K}\sum_{k=0}^{K-1}x_i(k\tau))\Big] - f(x^*) \leq \frac{\mathbb{E}[\|x_i(0) - x^*\|^2]}{(2\alpha\tau - 12\tau^2 L\alpha^2)K} + A\sigma^2,$$

for any $i \in \mathcal{S}$, which completes the proof.

H Supporting Lemmas for the Proof of Theorem 4

Lemma 8. *Under Assumption 1, if the stepsize satisfies $0 < \alpha \leq \frac{1}{\tau L}$, we have*

$$\mathbb{E}[\|x_i(k\tau + h) - x_i(k\tau)\|^2] \leq 18\tau^2\alpha^2\mathbb{E}[\|\nabla f(x_i(k\tau))\|^2] + 81\tau\alpha^2\sigma^2,$$

for any $1 \leq h < \tau$ and $i \in \mathcal{S}$.

Proof. From (56) and (57), we have

$$x_i(k\tau + j + 1) = x_i(k\tau + j) - \alpha\Big\{\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - g_i(x_i(k\tau)) + g_j(x_j(k\tau + j))\Big\}.$$

The above equation implies

$$\begin{aligned}
&x_i(k\tau + j + 1) - x_i(k\tau)\\
=&x_i(k\tau + j) - x_i(k\tau) - \alpha\Big\{\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) - \nabla f_i(x_i(k\tau)) + \nabla f_j(x_j(k\tau + j))\Big\}\\
&- \alpha\Big\{\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) + \nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau))\\
&\quad + g_j(x_j(k\tau + j)) - \nabla f_j(x_j(k\tau + j))\Big\}.
\end{aligned}$$

From Assumption 3, we obtain

$$\begin{aligned}
&\mathbb{E}\Big[\|x_i(k\tau + j + 1) - x_i(k\tau)\|^2\Big]\\
=&\mathbb{E}\Big[\Big\|x_i(k\tau + j) - x_i(k\tau) - \alpha\Big(\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) - \nabla f_i(x_i(k\tau)) + \nabla f_j(x_j(k\tau + j))\Big)\Big\|^2\Big]\\
&+ \alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau)) + \nabla f_i(x_i(k\tau)) - g_i(x_i(k\tau))\\
&\quad + g_j(x_j(k\tau + j)) - \nabla f_j(x_j(k\tau + j))\Big\|^2\Big].
\end{aligned} \tag{50}$$

For the first term of (50), we have

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|x_i(k\tau+j)-x_i(k\tau)-\alpha\Big(\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau))-\nabla f_i(x_i(k\tau))+\nabla f_i(x_i(k\tau+j))\Big)\Big\|^2\Big]\\
\leq&(1+\frac{1}{\tau})\mathbb{E}\Big[\Big\|x_i(k\tau+j)-x_i(k\tau)-\alpha\Big(\nabla f_i(x_i(k\tau+j))-\nabla f_i(x_i(k\tau))\Big)\Big\|^2\Big]\\
&+(1+\tau)\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau))\Big\|^2\Big].
\end{aligned}\tag{51}$$

Assumption 1 and (51) imply

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|x_i(k\tau+j)-x_i(k\tau)-\alpha\Big(\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau))-\nabla f_i(x_i(k\tau))+\nabla f_i(x_i(k\tau+j))\Big)\Big\|^2\Big]\\
\leq&(1+\alpha L)^2(1+\frac{1}{\tau})\mathbb{E}\Big[\|x_i(k\tau+j)-x_i(k\tau)\|^2\Big]+(1+\tau)\alpha^2\mathbb{E}\Big[\|\nabla f(x_j(k\tau))\|^2\Big].
\end{aligned}\tag{52}$$

For the second term of (50), from Assumption 3, we have

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}g_j(x_j(k\tau))-\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau))+\nabla f_i(x_i(k\tau))-g_i(x_i(k\tau))\\
&\qquad+g_j(x_j(k\tau+j))-\nabla f_j(x_j(k\tau+j))\Big\|^2\Big]\\
\leq&\frac{3}{N}\sum_{j=1}^{N}\mathbb{E}\Big[\|g_j(x_j(k\tau))-\nabla f_j(x_j(k\tau))\|^2\Big]+3\mathbb{E}\Big[\|\nabla f_i(x_i(k\tau))-g_i(x_i(k\tau))\|^2\Big]\\
&+3\mathbb{E}\Big[\|g_j(x_j(k\tau+j))-\nabla f_j(x_j(k\tau+j))\|^2\Big\}.
\end{aligned}$$

Combining the above inequality and Assumption 3 implies

$$\begin{aligned}
&\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}g_j(x_j(k\tau))-\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j(k\tau))+\nabla f_i(x_i(k\tau))-g_i(x_i(k\tau))\\
&\qquad+g_j(x_j(k\tau+j))-\nabla f_j(x_j(k\tau+j))\Big\|^2\Big]\leq 9\sigma^2.
\end{aligned}\tag{53}$$

Combining (50), (52), and (53) leads to

$$\begin{aligned}
&\mathbb{E}\Big[\|x_i(k\tau+j+1)-x_i(k\tau)\|^2\Big]\\
\leq&(1+\alpha L)^2(1+\frac{1}{\tau})\mathbb{E}\Big[\|x_i(k\tau+j)-x_i(k\tau)\|^2\Big]+(1+\tau)\alpha^2\mathbb{E}\Big[\|\nabla f(x_j(k\tau))\|^2\Big]+9\alpha^2\sigma^2.
\end{aligned}$$

Because the stepsize satisfies $0<\alpha\leq\frac{1}{\tau L}$, we have

$$\mathbb{E}\Big[\|x_i(k\tau+j)-x_i(k\tau)\|^2\Big]\leq\Big\{(1+\tau)\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2]+9\alpha^2\sigma^2\Big\}\sum_{h=0}^{j-1}\Big((1+\frac{1}{\tau})^3\Big)^h,$$

which further implies

$$\mathbb{E}\Big[\|x_i(k\tau+j)-x_i(k\tau)\|^2\Big]\leq\Big\{(1+\tau)\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2]+9\alpha^2\sigma^2\Big\}\frac{(1+\frac{1}{\tau})^{3j}-1}{(1+\frac{1}{\tau})^3-1},\tag{54}$$

for any $0 \leq j < \tau$ and $i \in \mathcal{S}$.

In addition, we have

$$(1+\frac{1}{\tau})^{3j} - 1 \leq 3^3, \qquad (1+\frac{1}{\tau})^3 - 1 \geq \frac{3}{\tau}, \tag{55}$$

for any $0 \leq j < \tau$.

Therefore, combining (54) and (55) leads to

$$\mathbb{E}\Big[\|x_i(k\tau+j) - x_i(k\tau)\|^2\Big] \leq 18\tau^2\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2] + 81\tau\alpha^2\sigma^2,$$

which completes the proof. □

I Proof of Theorem 4

For the convenience of expression, we use $g_i(x_i(t))$ to denote $\nabla f_i(x_i(t), \xi_i(t))$ for any $i \in \mathcal{S}$ and $t \geq 0$. From Appendix C, for any $j = 1, 2, \cdots, \tau - 1$, we have

$$\begin{cases} x_i(k\tau+j) = x_i(k\tau+j-1) - \alpha y_i(k\tau+j-1), \\ y_i(k\tau+j) = y_i(k\tau+j-1) + g_i(x_i(k\tau+j)) - g_i(x_i(k\tau+j-1)) \\ y_i(k\tau+j) = y_i(k\tau) + g_i(x_i(k\tau+j)) - g_i(x_i(k\tau)). \end{cases} \tag{56}$$

Moreover, similar to the derivation of Lemma 4, we have

$$y_i(k\tau) = \frac{1}{N}\sum_{j=1}^{N} g_j(x_j(k\tau)) \tag{57}$$

for any $i \in \mathcal{S}$.

From (56) and (57), we have

$$x_i(k\tau+k) = x_i(k\tau) - \frac{\alpha}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h)),$$

for any $i \in \mathcal{S}$. Thus, using Assumption 1, we arrive at

$$\begin{aligned} &f(x_i(k\tau+\tau)) \\ \leq& f(x_i(k\tau)) - \alpha\langle\nabla f(x_i(k\tau)), \frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau))\rangle + \frac{\alpha^2 L}{2}\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h))\|^2 \\ &- \alpha\langle\nabla f(x_i(k\tau)), \frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)) - \frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau))\rangle \\ &- \alpha\langle\nabla f(x_i(k\tau)), \frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h)) - \frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h))\rangle. \end{aligned} \tag{58}$$

Taking the expectation of both sides of (58) leads to

$$\begin{aligned} &\mathbb{E}[f(x_i(k\tau+\tau))] \\ \leq& \mathbb{E}[f(x_i(k\tau))] - \alpha\tau\mathbb{E}[\|\nabla f(x_i(k\tau))\|^2] + \frac{\alpha^2 L}{2}\mathbb{E}\Big[\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h))\|^2\Big] \\ &+ \alpha\mathbb{E}\Big[\|\nabla f(x_i(k\tau))\|\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau+h)) - \frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\nabla f_j(x_j(k\tau))\|\Big]. \end{aligned} \tag{59}$$

Using the inequality $2ab \leq a^2 + b^2$, the relation in (59), and Assumption 3, we have

$$\begin{aligned}
&\mathbb{E}[f(x_i(k\tau+\tau))] \\
\leq&\mathbb{E}[f(x_i(k\tau))] - \frac{\alpha\tau}{2}\mathbb{E}\Big[\|\nabla f(x_i(k\tau))\|^2\Big] + \frac{\alpha^2 L}{2}\mathbb{E}\Big[\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h))\|^2\Big] \\
&+ \frac{\alpha L^2}{2N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}\Big[\|x_j(k\tau+h) - x_j(k\tau)\|^2\Big]. \qquad (60)
\end{aligned}$$

For the term $\frac{\alpha L^2}{2N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}[\|x_j(k\tau+h) - x_j(k\tau)\|^2]$ on the right hand side of the preceding inequality, from Lemma 8, we have

$$\frac{\alpha L^2}{2N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}\Big[\|x_j(k\tau+h) - x_j(k\tau)\|^2\Big] \leq \frac{\alpha L^2}{2N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{18\tau^2\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2] + 81\tau\alpha^2\sigma^2\Big\}. \qquad (61)$$

The stepsize condition $0 < \alpha \leq \frac{1}{6\tau L}$ and (61) imply

$$\frac{\alpha L^2}{2N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}\Big[\|x_j(k\tau+h) - x_j(k\tau)\|^2\Big] \leq 3\tau^2\alpha^2 L\mathbb{E}\Big[\|\nabla f(x_j(k\tau))\|^2\Big] + 7\tau^2\alpha^2 L\sigma^2. \qquad (62)$$

For the term $\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h))\|^2$ in (60), we have

$$\begin{aligned}
&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau+h))\Big\|^2 \\
\leq&2\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h)) - g_j(x_j(k\tau))\Big\}\Big\|^2 + 2\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1} g_j(x_j(k\tau))\Big\|^2. \qquad (63)
\end{aligned}$$

For the term $\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\{g_j(x_j(k\tau+h)) - g_j(x_j(k\tau))\}\|^2$ in (60), we have

$$\begin{aligned}
&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h)) - g_j(x_j(k\tau))\Big\}\Big\|^2 \\
\leq&\frac{\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|\Big(\nabla f_j(x_j(k\tau+h)) - \nabla f_j(x_j(k\tau))\Big) + \Big(g_j(x_j(k\tau+h)) - \nabla f_j(x_j(k\tau+h))\Big) \\
&+ \Big(\nabla f_j(x_j(k\tau)) - g_j(x_j(k\tau))\Big)\Big\|^2,
\end{aligned}$$

which further implies

$$\begin{aligned}
&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h)) - g_j(x_j(k\tau))\Big\}\Big\|^2 \\
\leq&\frac{3\tau L^2\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|x_j(k\tau+h) - x_j(k\tau)\Big\|^2 + \frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|\nabla f_j(x_j(k\tau)) - g_j(x_j(k\tau))\Big\|^2 \\
&+ \frac{3\tau\alpha^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\|g_j(x_j(k\tau+h)) - \nabla f_j(x_j(k\tau+h))\Big\|^2. \qquad (64)
\end{aligned}$$

From (64) and Assumption 3, we have

$$\begin{aligned}&2\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2\Big]\\ \leq&\frac{6\tau\alpha^2L^2}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\mathbb{E}\Big[\Big\|x_j(k\tau+h)-x_j(k\tau)\Big\|^2\Big]+12\alpha^2\tau^2\sigma^2.\end{aligned}$$

Combining Lemma 8 and the above inequality yields

$$\begin{aligned}&2\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2\Big]\\ \leq&108\tau^4L^2\alpha^4\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2]+486\tau^3L^2\alpha^4\sigma^2+12\alpha^2\tau^2\sigma^2.\end{aligned}\tag{65}$$

For the term $\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\Big\|^2$ in (63), we have

$$\begin{aligned}&2\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\Big\|^2\Big]\\ \leq&4\alpha^2\tau^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\Big\{g_j(x_j(k\tau))-\nabla f_j(x_j(k\tau))\Big\}\Big\|^2\Big]+4\alpha^2\tau^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\Big\{\nabla f_j(x_j(k\tau))\Big\}\Big\|^2\Big].\end{aligned}\tag{66}$$

From (66) and Assumption 3, we have

$$2\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\Big\|^2\Big]\leq4\alpha^2\tau^2\sigma^2+4\alpha^2\tau^2\mathbb{E}[\|\nabla f(x_i(k\tau))\|^2],\tag{67}$$

for any $i\in\mathcal{S}$.

Using the inequality $\|a+b\|^2\leq2\|a\|^2+2\|b\|^2$, we have

$$\begin{aligned}&\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\\ \leq&2\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}\Big\{g_j(x_j(k\tau+h))-g_j(x_j(k\tau))\Big\}\Big\|^2+2\alpha^2\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau))\Big\|^2.\end{aligned}$$

Combining (63), (65), and (67) leads to

$$\begin{aligned}&\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\Big]\\ \leq&108\tau^4L^2\alpha^4\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2]+486\tau^3L^2\alpha^4\sigma^2+16\alpha^2\tau^2\sigma^2+4\alpha^2\tau^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2].\end{aligned}$$

Plugging the stepsize condition $0<6\tau\alpha L\leq1$ into the preceding inequality leads to

$$\alpha^2\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\Big]\leq7\tau^2\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2]+30\tau^2\alpha^2\sigma^2,$$

i.e.,

$$\frac{L\alpha^2}{2}\mathbb{E}\Big[\Big\|\frac{1}{N}\sum_{j=1}^{N}\sum_{h=0}^{\tau-1}g_j(x_j(k\tau+h))\Big\|^2\Big]\leq\frac{7}{2}\tau^2L\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2]+15\tau^2\alpha^2L\sigma^2.\tag{68}$$

From (60), (62), and (68), we have

$$\begin{aligned}\mathbb{E}[f(x_i(k\tau+\tau))] \leq & \mathbb{E}[f(x_i(k\tau))] - \frac{\alpha\tau}{2}\mathbb{E}[\|\nabla f(x_i(k\tau))\|^2] + \frac{13}{2}\tau^2 L\alpha^2\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2] \\ & + 22\tau^2\alpha^2 L\sigma^2,\end{aligned}$$

i.e.,

$$\Big(\frac{\alpha\tau}{2} - \frac{13}{2}\tau^2 L\alpha^2\Big)\mathbb{E}[\|\nabla f(x_j(k\tau))\|^2] \leq \mathbb{E}[f(x_i(k\tau))] - \mathbb{E}[f(x_i(k\tau+\tau))] + 22\tau^2\alpha^2 L\sigma^2.$$

Under a stepsize satisfing $0 < \alpha < \frac{1}{13\tau L}$, we have

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(x_i(k\tau))\|^2] \leq \frac{\mathbb{E}[f(x_i(0))] - f(x^*)}{(\frac{\alpha\tau}{2} - \frac{13}{2}\tau^2 L\alpha^2)K} + \frac{44\tau^2\alpha^2 L}{\alpha\tau - 13\tau^2 L\alpha^2}\sigma^2$$

for any $i \in \mathcal{S}$, which completes the proof.