Non-Convex Federated Optimization under Cost-Aware Client Selection

Xiaowen Jiang
Saarland University & CISPA*
xiaowen.jiang@cispa.de

Anton Rodomanov
CISPA*
anton.rodomanov@cispa.de

Sebastian U. Stich
CISPA*
stich@cispa.de

Abstract

Different federated optimization algorithms typically employ distinct client-selection strategies: some methods communicate only with a randomly sampled subset of clients at each round, while others need to periodically communicate with all clients or use a hybrid scheme that combines both strategies. However, existing metrics for comparing optimization methods typically do not distinguish between these strategies, which often incur different communication costs in practice. To address this disparity, we introduce a simple and natural model of federated optimization that quantifies communication and local computation complexities. This new model allows for several commonly used client-selection strategies and explicitly associates each with a distinct cost. Within this setting, we propose a new algorithm that achieves the best-known communication and local complexities among existing federated optimization methods for non-convex optimization. This algorithm is based on the inexact composite gradient method with a carefully constructed gradient estimator and a special procedure for solving the auxiliary subproblem at each iteration. The gradient estimator is based on SAGA, a popular variance-reduced gradient estimator. We first derive a new variance bound for it, showing that SAGA can exploit functional similarity. We then introduce the Recursive-Gradient technique as a general way to potentially improve the error bound of a given conditionally unbiased gradient estimator, including both SAGA and SVRG. By applying this technique to SAGA, we obtain a new estimator, RG-SAGA, which has an improved error bound compared to the original one.

1 Introduction

Motivation. Federated Learning (FL) is a distributed training paradigm in which a central server coordinates model updates across multiple remote clients—such as mobile devices or hospitals—without requiring access to their local data [1, 2]. This framework enables collaborative learning on decentralized data, but introduces new algorithmic challenges due to the distributed nature of optimization.

A key issue in FL is the high cost of communication between the clients and the server. Clients may be intermittently available [3] and connected over slow or unreliable networks. These constraints make it critical to design optimization algorithms that minimize communication costs, particularly in settings with partial client participation.

Various federated optimization algorithms have been proposed to address communication efficiency, each often relying on distinct client-selection strategies. Some methods communicate only with a randomly sampled subset of clients at each round, while others need to select the set of participating clients more carefully or employ hybrid schemes that combine both strategies. While prior works [4, 5, 6, 7, 8, 9] introduced a few models for federated optimization, they do not account for the varying costs of each client-selection strategy, which can in practice differ due to factors such as client reliability, device heterogeneity, and network conditions. Consequently, existing metrics such as the number of communication rounds are not entirely fair for comparing methods in such scenario.

*CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

For instance, optimization methods based on SARAH [10, 11] have been shown to be communication-efficient in finding an approximate stationary point [12, 13]. This efficiency arises from the method's ability to exploit dissimilarity (δ) between local and global objectives. In many practical scenarios—such as statistical or semi-supervised learning [14, 15, 13]—δ is often small, leading to substantial theoretical gains in communication cost. However, SARAH-based methods require periodic full synchronization with all clients in order to compute full gradients. This can be impractical in real-world large-scale federated systems, where clients may be intermittently unavailable due to energy constraints, network issues, or user behavior.

In contrast to SARAH, methods such as SAG [16] and SAGA [17] are naturally better suited to the partial participation setting in FL. These methods update the model by sampling a small subset of clients at each round and using locally stored gradients. As a result, they avoid the need for periodic full synchronization, which makes them more compatible with federated systems where only a fraction of clients may be available at any given time. Despite this advantage, the existing communication complexity of such methods depends on the individual smoothness constant $L_{\max}$ [18, 19, 20], which can be significantly larger than the dissimilarity constant δ. Consequently, it remains unclear whether such methods are more communication-efficient than SARAH-based methods, since they rely on fundamentally different client-selection strategies with different constant dependencies.

Contributions. In this work, we aim to develop optimization algorithms that are efficient in both communication and local computation in the setting where client-selection strategies incur different costs. Our main contributions are as follows:

- We propose a new model formalizing the concept of federated optimization algorithms and defining information-based notions of communication and local complexities. This model associates the non-uniform costs with different client-selection strategies, enabling fair comparisons across optimization algorithms. (Section 2.1)
- Within our new model, we propose a new gradient method that achieves the best communication and local complexities among existing first-order methods for non-convex optimization. This method is based on the inexact composite gradient method (I-CGM) with a carefully constructed gradient estimator and a special procedure for solving auxiliary subproblem at each iteration. (Section 6)
- Specifically, we first study the convergence of I-CGM for arbitrary gradient estimators and present an efficient technique for solving the auxiliary subproblem. Our technique is based on running the classical composite gradient method locally for a random number of iterations following a geometrical distribution with a carefully chosen parameter. (Section 3)
- We then analyze the SAGA estimator and establish a new variance bound for it that only depends on δ without requiring individual smoothness, improving upon previous results showing that SAGA can exploit functional similarity. We also study SVRG as another example that can be incorporated into I-CGM. (Section 4)
- Finally, we introduce the *Recursive-Gradient (RG) technique* as a general way to potentially improve the error bound for a given conditionally unbiased gradient estimator, including both SAGA and SVRG. Applying this technique to SAGA and SVRG, we obtain new RG-SAGA and RG-SVRG gradient estimators with better error bounds compared to the original ones. (Section 5)

We discuss our results in detail in the context of related work in Appendix A and summarize them in Table 1.

2 Problem Formulation

We consider the following federated optimization problem:

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

where each $f_i\colon \mathbb{R}^d \to \mathbb{R}$ is a differentiable function which can be directly accessed only by client i.

Notation. We abbreviate $[n] := \{1, 2, \ldots, n\}$. For a finite set A and an integer $1 \le m \le |A|$, $\binom{A}{m}$ denotes the power set comprised of all m-element subsets of A. $\|\cdot\|$ denotes the standard Euclidean

Table 1: Summary of efficiency guarantees (in BigO-notation) for finding an ε-stationary point. I-CGM-RG-SAGA achieves the best communication and local complexities. For the precise description of the problem classes, notations, as well as the discussions of the methods, see Section A.

Method	Communication complexity	Assumption	Local complexity	VR type
Centralized GD	$C_A n_m \frac{L_f F^0}{\varepsilon^2}$	FS	$n_m \frac{L_f F^0}{\varepsilon^2}$	None
FedRed [21]	$C_A n_m \frac{\Delta_1 F^0}{\varepsilon^2}$	2.1 ; 2.3	$\frac{L_1 F^0}{\varepsilon^2} + n_m \frac{\Delta_1 F^0}{\varepsilon^2}$	None
FedAvg [1]	$C_R\big(\frac{\zeta_m^2 F^0}{\varepsilon^4} + \frac{\sqrt{L_{\max}}\zeta}{\varepsilon^3} + \frac{L_{\max} F^0}{\varepsilon^2}\big)$	IS , BGD	$\frac{\zeta_m^2 F^0}{\varepsilon^4} + \frac{\sqrt{L_{\max}}\zeta}{\varepsilon^3} + \frac{L_{\max} F^0}{\varepsilon^2}$	None
FedDyn [22]	$C_A n_m + C_R n_m \frac{L_{\max} F^0}{\varepsilon^2}$	IS	unknown	None
MimeMVR [15]	$C_A\big(\frac{\zeta_m^2 F^0}{\varepsilon^2} + \frac{\zeta_m \Delta_{\max} F^0}{\varepsilon^3} + \frac{\Delta_{\max} F^0}{\varepsilon^2}\big)$	IS, BGD, SD	$\frac{L_{\max}\zeta_m^2 F^0}{\Delta_{\max}\varepsilon^2} + \frac{\zeta_m L_{\max} F^0}{\varepsilon^3} + \frac{L_{\max} F^0}{\varepsilon^2}$	None
CE-LGD [6]	$C_R\big(\frac{\zeta_m^2 F^0}{\varepsilon^2} + \frac{\zeta_m \Delta_{\max} F^0}{\sqrt{m}\varepsilon^3} + \frac{\Delta_{\max} F^0}{\varepsilon^2}\big)$	IS, BGD, SD	$\frac{L_{\max}\zeta_m^2 F^0}{\Delta_{\max}\varepsilon^2} + \frac{\zeta_m L_{\max} F^0}{\sqrt{m}\varepsilon^3} + \frac{L_{\max} F^0}{\varepsilon^2}$	None
Scaffold [20]	$C_A n_m + C_A \frac{n_m^{2/3} L_{\max} F^0}{\varepsilon^2}$	IS	$n_m + \frac{n_m^{2/3} L_{\max} F^0}{\varepsilon^2}$	SAG [16]
SABER-full [12]	$C_A n_m + C_A \frac{(\Delta_{\max} + \sqrt{n_m}\delta_m) F^0}{\varepsilon^2}$	SD	unknown	PAGE [11]
SABER-partial [12]	$C_A n_m + C_R \frac{\zeta_m^2}{\varepsilon^2} \frac{\Delta_{\max} F^0}{\varepsilon^2}$	SD, BGD	unknown	SARAH [10]
I-CGM-RG-SVRG (ours)	$C_A n_m + \frac{(C_R\Delta_1 + \sqrt{C_A C_R n_m}\delta_m) F^0}{\varepsilon^2}$	2.1,2.2 ; 2.3	$n_m + \frac{(L_1 + \Delta_1 + \sqrt{\frac{C_A}{C_R} n_m}\delta_m) F^0}{\varepsilon^2}$	RG-SVRG [23]
I-CGM-RG-SAGA (ours)	$C_A n_m + C_R \frac{(\Delta_1 + \sqrt{n_m}\delta_m) F^0}{\varepsilon^2}$	2.1,2.2 ; 2.3	$n_m + \frac{(L_1 + \Delta_1 + \sqrt{n_m}\delta_m) F^0}{\varepsilon^2}$	RG-SAGA [17]

norm in $\mathbb{R}^d$. We use $\mathbb{E}[\cdot]$ to denote the standard (full) expectation. We write $\mathbb{E}_\xi[\cdot]$ for the expectation taken w.r.t. ξ. We assume that the objective function in problem (1) is bounded from below and denote its infimum by $f^\star$. We denote $F^0 := f(\mathbf{x}^0) - f^\star$ where $\mathbf{x}^0$ is the initial point.

2.1 Federated Optimization Algorithms and their Complexity

Federated Optimization Algorithm. We consider the standard federated optimization setting with a central server and n clients. The server is the main entity that implements the optimization algorithm, but cannot directly access any of the local functions $(f_i)_{i=1}^n$. Instead, it interacts with problem (1) through communications with the clients, allowing certain information to be exchanged between them. Each client $i \in [n]$ has access to the information provided by the server and can interact with its own local function f_i, but only through the oracle O_{f_i}. An oracle is a standard notion in optimization [24, 25], which is a procedure that takes as input a point and returns certain information about the function at this point. The most commonly used oracle is the first-order oracle, which returns the function value and its gradient. In general, an oracle can be stochastic; however, in this work, we mainly consider the standard deterministic first-order oracle $\mathrm{O}_{\mathrm{FO}_i}(\mathbf{x}) := (f_i(\mathbf{x}), \nabla f_i(\mathbf{x}))$. Throughout the paper, we assume that the server can communicate with up to $m \in [n]$ clients simultaneously in parallel. We formalize optimization algorithms in this setting as follows.

Given the oracles $\mathrm{O}_{f_1}, \ldots, \mathrm{O}_{f_n}$, a *federated optimization algorithm* for a problem class $\mathcal{F}$ is a procedure that proceeds across *communication rounds*. A *problem class* is the collection of all problems of form (1) satisfying certain assumptions. (We will introduce a specific problem class considered in this work in Section 2.2.) At the beginning, the server and each client $i \in [n]$ initialize the empty information sets $\mathcal{I}^0$ and $\mathcal{H}_i^0$, respectively. At each round $r \geq 0$, the server chooses a subset of clients $S_r \subseteq [n]$ with at most m elements (to be discussed later). The server then communicates with the clients in S_r, providing each client $i \in S_r$ with certain information $\bar{\mathcal{I}}_i^r$ constructed from the server's information set $\mathcal{I}^r$. Then it specifies a certain method $\mathcal{M}_i^r$ (often called a *local method*) for each client $i \in S_r$ to run locally. The method $\mathcal{M}_i^r$ starts with the initial information $(\bar{\mathcal{I}}_i^r, \mathcal{H}_i^r)$, and iteratively queries the oracle O_{f_i}, obtaining a response $\mathcal{R}_i^r$, which is then sent back to the server. (The details of this procedure are discussed in the next paragraph.) The server collects the output responses and updates the information set $\mathcal{I}^{r+1} = (\mathcal{I}^r, (\mathcal{R}_i^r)_{i \in S_r})$. At each round $r \geq 0$, the server also performs a termination test based on the current information set $\mathcal{I}^r$. After the algorithm terminates at a certain round $R \geq 0$, the server then constructs and outputs an approximate solution $\hat{\mathbf{x}}^R$ to problem (1) based on $\mathcal{I}^R$ using a certain rule specified by the algorithm. To summarize, a federated optimization algorithm is a collection of rules prescribing what to do at each communication round r: 1) how to select clients, 2) how to compute the information $\bar{\mathcal{I}}_i^r$ that is sent to each selected client, 3) which local method each selected client runs, 4) when to terminate, and 5) how to form the approximate solution. We allow each of these rules to be randomized. See Figure 1 for an illustration summarizing the procedures described above.

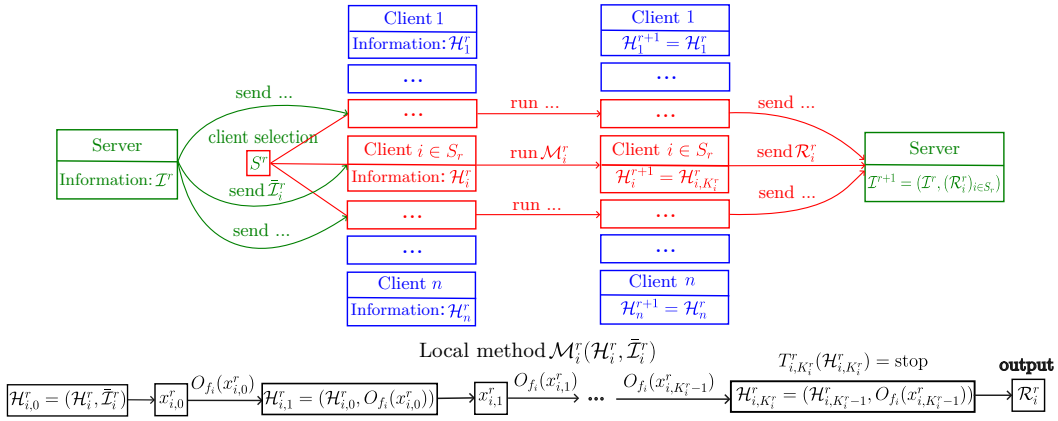

Figure 1: Illustration of the sequence of procedures performed by a federated optimization algorithm at each communication round r. Each client $i \in S_r$ can make different number of local steps.

At the beginning of each round r, each selected client $i \in S_r$ receives the information $\bar{\mathcal{I}}_i^r$ from the server. Using this new information, it enriches its information set $\mathcal{H}_{i,0}^r := (\mathcal{H}_i^r, \bar{\mathcal{I}}_i^r)$ and runs the specified method $\mathcal{M}_i^r$. At each step $k \geq 0$, this method first computes a point $\mathbf{x}_{i,k}^r$ based on $\mathcal{H}_{i,k}^r$, queries the oracle at this point, and then updates its local information: $\mathcal{H}_{i,k+1}^r = (\mathcal{H}_{i,k}^r, \mathrm{O}_{f_i}(\mathbf{x}_{i,k}^r))$. At the beginning of each step k, the method also performs a termination test $T_{i,k}^r(\mathcal{H}_{i,k}^r)$; Once this test is satisfied at a certain step K_i^r, the method terminates and constructs the output $\mathcal{R}_i^r$ from the final information $\mathcal{H}_{i,K_i^r}^r$. The information sets are then updated as $\mathcal{H}_i^{r+1} := \mathcal{H}_{i,K_i^r}^r$, and remain the same ($\mathcal{H}_i^{r+1} := \mathcal{H}_i^r$) for each non-selected client $i \notin S_r$. To summarize, a local method $\mathcal{M}_i^r$ is a collection of 3 rules: 1) how to compute the next point at each step, 2) when to terminate, and 3) how to form the result. We allow each of these rules to be randomized (resulting in a *randomized* local method); if all the rules are deterministic, the local method is called *deterministic*.

Note that the above definition of a distributed optimization algorithm is rather general and only constrains how the algorithm accesses information about the optimization problem. In particular, we do not impose any restrictions on the arithmetic or memory complexity of each step of the algorithm, nor on the size of the data transmitted between the server and the clients. This general definition is sufficient to introduce the two *information-based* notions of complexity that we focus on in this work: communication and local complexities (defined below). In practice, however, both memory storage and information usage should be implemented efficiently. Typically, the accumulated information sets maintained by the clients and the server, as well as the information exchanged between them, are simply a collection of a few vectors and scalars.

Client-Selection Strategies. We next introduce three commonly used client-selection strategies and associate them with different costs. The distinction among them lies in how the set S_r is selected.

- *Arbitrary Client Selection Strategy* (A-CSS): The set S_r can be chosen in any way from $\binom{[n]}{m}$. We define the cost of this operation as C_A.
- *Random Client Selection Strategy* (R-CSS): The set S_r is sampled uniformly at random from $\binom{[n]}{m}$. We define the cost of this operation as C_R.
- *Delegated Client Selection Strategy* (D-CSS): The set S_r is chosen to be S_D, where S_D is a fixed set of so-called delegate clients (to be discussed later) with $|S_D| \leq m$. We define the cost of this operation as 1.

Clearly, A-CSS is the most powerful among the three strategies, as the other two could be easily implemented in terms of A-CSS. Further, this strategy allows the server to collect information from any subset of clients. This flexibility enables the implementation of *full synchronization*, where the algorithm needs certain information to be collected from all clients. This feature appears in many algorithms, the most basic example being the usual gradient descent (GD). Specifically, if an algorithm requires computing the full gradient $\nabla f(\mathbf{x})$ at a point $\mathbf{x}$, the server can split $[n]$ into m disjoint sets and repeatedly use A-CSS to make $\lceil \frac{n}{m} \rceil$ sequential communications with each set of

clients, sending to each client the point $\mathbf{x}$ and asking it to compute and return the gradient $\nabla f_i(\mathbf{x})$ (this corresponds to the simplest one-step local first-order method $\mathcal{M}_i^r$).

However, when clients are unreliable or slow to respond, using A-CSS can become costly. In cross-device settings, it is often more efficient to communicate only with a randomly sampled subset of clients at each communication round—a strategy commonly known as partial client participation [1], which is modeled by the R-CSS. Therefore, we treat R-CSS as a cheaper strategy compared with A-CSS. Unlike A-CSS, the full-gradient computation cannot be directly implemented with R-CSS. (But it can be recovered with high probability by using R-CSS multiple times [26].)

In addition to the previous two strategies, there are scenarios where there exist so-called *delegate* clients that are always reliable and efficient both in communication and performing local computations. Sometimes, it is sufficient—or even preferable—to interact with these clients. With D-CSS, the server can always query information about the specific functions in the delegate set. In this work, we focus on the setting where there is one delegate client (number 1), i.e., $S_D = \{1\}$.

Based on the properties of each strategy discussed above, we assume that the above costs satisfy the following natural relations:

$$\boxed{1 \leq C_R \leq C_A}\,.$$

Communication-and Local Complexities. Consider a federated optimization algorithm $\mathcal{A}$ for solving a problem f from the problem class $\mathcal{F}$. Let R be the (possibly random) number of communication rounds made by $\mathcal{A}$ on f and let $\hat{\mathbf{x}}^R$ be the corresponding output of $\mathcal{A}$. We define the *accuracy* of the algorithm $\mathcal{A}$ at the problem f as:

$$\boxed{\mathrm{Accur}(\mathcal{A}, f) = \mathbb{E}[\|\nabla f(\hat{\mathbf{x}}^R)\|^2]}\,.$$

Further, let N_A, N_R, and N_D be the (possibly random) total number of times that the client-selection strategies A-CSS, R-CSS, and D-CSS are used by $\mathcal{A}$ during the R communication rounds, respectively. We define the *communication complexity* of $\mathcal{A}$ on f as:

$$\boxed{N_f = \mathbb{E}[C_A N_A + C_R N_R + N_D]}\,,$$

and the *local complexity* of $\mathcal{A}$ on f as:

$$\boxed{K_f = \mathbb{E}\left[\sum_{r=0}^{R-1} K_r\right]}\,,$$

where $K_r := \max_{i \in S_r} K_i^r$ and $K_i^r \geq 0$ is the number of queries to the oracle O_{f_i} by the client i at round r. We next define the worst-case complexities of $\mathcal{A}$ over the entire problem class $\mathcal{F}$. The communication complexity of $\mathcal{A}$ for solving the problem class $\mathcal{F}$ up to ε accuracy is defined as:

$$\boxed{N_{\mathcal{F}}(\varepsilon) = \sup_{f \in \mathcal{F}} \left\{ N_f \middle|\, \mathrm{Accur}(\mathcal{A}, f) \leq \varepsilon^2 \right\}}\,,$$

and the corresponding local complexity is defined as:

$$\boxed{K_{\mathcal{F}}(\varepsilon) = \sup_{f \in \mathcal{F}} \left\{ K_f \middle|\, \mathrm{Accur}(\mathcal{A}, f) \leq \varepsilon^2 \right\}}\,.$$

If there exists some $f \in \mathcal{F}$ such that $\mathcal{A}$ fails to reach $\mathrm{Accur}(\mathcal{A}, f) \leq \varepsilon^2$, then both complexities $N_{\mathcal{F}}(\varepsilon)$ and $K_{\mathcal{F}}(\varepsilon)$ are defined as $+\infty$.

After fixing the desired accuracy ε, we consider only federated optimization algorithms that can achieve $\mathrm{Accur}(\mathcal{A}, f) \leq \varepsilon^2$ for all $f \in \mathcal{F}$. Among these algorithms, we say that the one with smaller communication complexity $N_{\mathcal{F}}(\varepsilon)$ is more efficient. If two algorithms have the same communication complexity, the one with lower local complexity $K_{\mathcal{F}}(\varepsilon)$ is preferable.

2.2 Problem Class

We study optimization problem (1) in which the client objectives exhibit an underlying similarity structure. Specifically, we use the following two assumptions that relax standard smoothness assumptions. The first quantifies the deviation between the delegate function f_1 and f. For an index $i \in [n]$, we use $h_i := f - f_i$ to denote the difference function.

Assumption 2.1. There exists $\Delta_1 > 0$ such that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have:

$$\|\nabla h_1(\mathbf{x}) - \nabla h_1(\mathbf{y})\| \leq \Delta_1 \|\mathbf{x} - \mathbf{y}\| . \tag{2}$$

Alternatively, one may define a uniform dissimilarity constant $\Delta_{\max}$ [20, 21] such that for any $i \in [n]$, it holds that $\|\nabla h_i(\mathbf{x}) - \nabla h_i(\mathbf{y})\| \leq \Delta_{\max} \|\mathbf{x} - \mathbf{y}\|$. In this work, we focus on Δ_1 since it can be much smaller than $\Delta_{\max}$.

The second assumption characterizes the average dissimilarity among all local functions.

Assumption 2.2 ([13, 27, 28, 21, 29])**.** There exists $\delta > 0$ such that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have:

$$\frac{1}{n} \sum_{i=1}^{n} \|\nabla h_i(\mathbf{x}) - \nabla h_i(\mathbf{y})\|^2 \leq \delta^2 \|\mathbf{x} - \mathbf{y}\|^2 . \tag{3}$$

The left-hand side of (3) is equal to $\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|^2 - \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|^2$, which can be interpreted as the variance of $\nabla f_i(x) - \nabla f_i(y)$ where i is selected uniformly at random. If each f_i has $L_{\max}$-Lipschitz gradient, then we have $\Delta_1 \leq 2L_{\max}$ and $\delta \leq L_{\max}$. Therefore, both conditions are weaker than assuming each f_i is Lipschitz-smooth. We refer to discussions in [27] for more properties and details.

The previous two quantities δ and Δ_1 will only affect the communication complexity of our algorithms, while the local complexity additionally depends on L_1 which is defined as follows.

Assumption 2.3. There exists $L_1 > 0$ such that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have:

$$\|\nabla f_1(\mathbf{x}) - \nabla f_1(\mathbf{y})\| \leq L_1 \|\mathbf{x} - \mathbf{y}\| . \tag{4}$$

3 Inexact Composite Gradient Method

3.1 Inexact Composite Gradient Method

We first introduce the Inexact Composite Gradient Method (I-CGM), which serves as the backbone of our approach. Consider the composite reformulation of the problem 1: $f = f_1 + [f - f_1] = f_1 + h_1$. Let $\lambda > 0$ and $\mathbf{x}^0 \in \mathbb{R}^d$ be the initial point. At each iteration $t \geq 0$, I-CGM computes an approximation of the gradient $\mathbf{g}^t \approx \nabla f(\mathbf{x}^t)$ and defines the next iterate as:

$$\boxed{\mathbf{x}^{t+1} \approx \arg\min_{\mathbf{x} \in \mathbb{R}^d} \Big\{ F_t(\mathbf{x}) := f_1(\mathbf{x}) + h_1(\mathbf{x}^t) + \langle \mathbf{g}^t - \nabla f_1(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t \rangle + \frac{\lambda}{2}\|\mathbf{x} - \mathbf{x}^t\|^2 \Big\},} \tag{I-CGM}$$

where both the inaccuracy in solving the subproblem and the approximation error (defined below) are assumed to be sufficiently small (to be specified later):

$$F_t(\mathbf{x}^{t+1}) \leq F_t(\mathbf{x}^t), \quad e_t := \|\nabla F_t(\mathbf{x}^{t+1})\|, \quad \hat{\Sigma}_t^2 := \left\|\mathbf{g}^t - \nabla f(\mathbf{x}^t)\right\|^2 . \tag{5}$$

In the following statement, we provide the general convergence guarantee for I-CGM. The proof can be found in Section C.1 in the Appendix.

Theorem 3.1. *Let I-CGM be applied to Problem* (1)*. Suppose Assumption 2.1 and condition* (5) *are satisfied. Let $\lambda > \Delta_1$. Then for any $T \geq 1$, we have:*

$$\begin{aligned} &\sum_{t=1}^{T} \|\nabla f(\mathbf{x}^t)\|^2 + (\lambda + \Delta_1)^2 \sum_{t=1}^{T} \|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2 \\ &\quad \leq \frac{12(\lambda + \Delta_1)^2}{\lambda - \Delta_1} F^0 + \Big(\frac{12(\lambda + \Delta_1)^2}{(\lambda - \Delta_1)^2} + 4 \Big) \sum_{t=0}^{T-1} \hat{\Sigma}_t^2 + 4 \sum_{t=0}^{T-1} e_t^2 . \end{aligned}$$

We see that each subproblem can be solved inexactly without affecting the convergence rate (up to absolute constants), provided that the error term $\sum_{t=0}^{T-1} e_t^2$ is of the same order as the first two terms on the right-hand side. Moreover, if the approximation errors $\sum_{t=0}^{T-1} \hat{\Sigma}_t^2$ can also be bounded by the first two terms on the left-hand side, then the convergence of the gradient norm is guaranteed. If there exists randomness either in solving the subproblems or in constructing the estimators, then these conditions are required to hold in expectation. Specifically, we obtain the following corollary.

Corollary 3.2. *Following the same settings as in Theorem 3.1. If the inaccuracies in solving the subproblems satisfy:*

$$F_t(\mathbf{x}^{t+1}) \leq F_t(\mathbf{x}^t), \quad \sum_{t=0}^{T-1} \mathbb{E}[e_t^2] \leq \frac{(\lambda+\Delta_1)^2}{\lambda-\Delta_1} F^0 + \sum_{t=0}^{T-1} \Sigma_t^2 \ , \tag{6}$$

and the approximation errors satisfy:

$$\Big(\frac{12(\lambda+\Delta_1)^2}{(\lambda-\Delta_1)^2} + 8\Big) \sum_{t=0}^{T-1} \Sigma_t^2 \leq \frac{1}{2} \sum_{t=1}^{T} G_t^2 + (\lambda+\Delta_1)^2 \sum_{t=1}^{T} \chi_t^2 \ , \tag{7}$$

then for any $T \geq 1$, we have:

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \frac{32(\lambda+\Delta_1)^2}{\lambda-\Delta_1} \frac{F^0}{T} \ .$$

where $G_t^2 := \mathbb{E}[\|\nabla f(\mathbf{x}^t)\|^2]$, $\chi_t^2 := \mathbb{E}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2]$, $\Sigma_t^2 := \mathbb{E}[\|\mathbf{g}^t - \nabla f(\mathbf{x}^t)\|^2]$, and $\bar{\mathbf{x}}^T$ is uniformly sampled from $(\mathbf{x}^t)_{t=1}^T$.

When $\mathbf{g}^t$ is the exact gradient $\nabla f(\mathbf{x}^t)$ for all $t \geq 0$, then I-CGM is reduced to CGM that is widely used for solving Problem (1), particularly because of its ability to exploit functional similarity and reduce communication costs [30, 21, 28, 13, 27, 12, 31]. Indeed, if $\lambda \simeq \Delta_1$ and the accuracy condition (6) is satisfied, then $\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \varepsilon^2$ after $T = \mathcal{O}(\frac{\Delta_1 F^0}{\varepsilon^2})$ iterations. In contrast, the iteration complexity of Gradient Descent depends on L_f which can be larger than Δ_1 (2.1) when f_1 is similar to f. However, CGM has sub-optimal communication complexity in terms of n. Indeed, let us assume, for simplicity, that $m = 1$. Then each iteration involves: 1) computing the full gradient $\nabla f(\mathbf{x}^t)$, which requires n sequential communication rounds using A-CSS, and 2) an additional round using D-CSS for solving the subproblem. Consequently, the total number of communication rounds with A-CSS and D-CSS is $N_A = nT$ and $N_D = T$, respectively. The communication complexity of CGM is thus: $C_A N_A + N_D = C_A nT + T = \mathcal{O}(C_A nT) = \mathcal{O}(C_A \frac{n\Delta_1 F^0}{\varepsilon^2})$. This linear dependency on n can be prohibitive in large-scale federated learning settings and is worse than the complexity of stochastic methods such as ProxSARAH [32], SpiderBoost [33], and PAGE [11], each of them achieving: $\mathcal{O}(C_A \frac{\sqrt{n}\bar{L}F^0}{\varepsilon^2})$, although they rely on a slightly different assumption of average smoothness [2]. Moreover, the dependence on $\frac{C_A}{\varepsilon^2}$ can become significantly large in scenarios where using A-CSS is costly.

3.2 Solving Auxiliary Subproblems

In this section, we assume that f_1 is L_1-smooth and study how to achieve the accuracy condition (6). Recall that each subproblem F_t consists of a smooth function $\phi(\mathbf{x}) = f_1(\mathbf{x})$ and a quadratic regularizer $\psi_t(\mathbf{x}) = \langle \mathbf{g}^t - \nabla f_1(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t \rangle + \frac{\lambda}{2} \|\mathbf{x} - \mathbf{x}^t\|^2$. Let us solve it using the standard composite gradient method (CGM), which proceeds as follows: For $k = 0, 1, ..., K_t - 1$,

$$\boxed{\begin{aligned} \mathbf{y}_{k+1}^t &= \arg\min_{\mathbf{y} \in \mathbb{R}^d} \Big\{ \phi(\mathbf{y}_k^t) + \big\langle \nabla \phi(\mathbf{y}_k^t), \mathbf{y} - \mathbf{y}_k^t \big\rangle + \frac{L_1}{2} \big\|\mathbf{y} - \mathbf{y}_k^t\big\|^2 + \psi_t(\mathbf{y}) \Big\} \\ &= \frac{1}{\lambda + L_1} \big(L_1 \mathbf{y}_k^t + \lambda \mathbf{x}^t + \nabla f_1(\mathbf{x}^t) - \mathbf{g}^t - \nabla f_1(\mathbf{y}_k^t) \big) \ . \end{aligned}} \tag{8}$$

Each CGM step monotonically decreases the function value of F_t (see Lemma C.2). Therefore, we can initialize $\mathbf{y}_0^t = \mathbf{x}^t$ and choose $\mathbf{x}^{t+1}$ to be a certain iterate of $(\mathbf{y}_k)_{k=0}^K$. Then the condition on $F_t(\mathbf{x}^{t+1}) \leq F_t(\mathbf{x}^t)$ is satisfied. We next study the number of local steps K_t required to achieve the second inequality in condition (6).

3.2.1 Fixed Number of Local Steps

Let $K_t \equiv K \geq 1$ be a constant number and let $\mathbf{x}^{t+1}$ be the iterate with the minimum gradient norm of F_t among $\{\mathbf{y}_k^t\}_{k=1}^K$. We use the notation $\mathbf{x}^{t+1} = \mathrm{CGM}_{\mathrm{const}}(\lambda, K, \mathbf{x}^t, \mathbf{g}^t)$ for this process.

[2] $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, it holds that $\frac{1}{n} \sum_{i=1}^n \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|^2 \leq \bar{L}^2 \|\mathbf{x} - \mathbf{y}\|^2$ and we have $\delta \leq \bar{L}$.

The goal is to upper bound $\sum_{t=0}^{T-1} e_t^2$ where $e_t := \|\nabla F_t(\mathbf{x}^{t+1})\|$. For each $t \geq 0$, we have: $e_t^2 \lesssim \frac{L_1(F_t(\mathbf{y}_0^t)-F_t(\mathbf{y}_K^t))}{K} \lesssim \frac{L_1(f(\mathbf{x}^t)-f(\mathbf{y}_K^t)+\frac{1}{\lambda}\hat{\Sigma}_t^2}{K}$ (see Lemma C.2 and C.1). However, since $\mathbf{y}_K^t$ and $\mathbf{x}^t$ are not necessarily the same, we cannot telescope $f(\mathbf{x}^t) - f(\mathbf{y}_K^t)$ when we sum up e_t^2. Instead, the "best" we can do is to upper bound $f(\mathbf{x}^t) - f(\mathbf{y}_K^t)$ by $f(\mathbf{x}^t) - f^\star$. Then by further upper-bounding the summation of $\sum_{t=0}^{T-1}[f(\mathbf{x}^t) - f^\star]$ in terms of F^0 and $\hat{\Sigma}_t^2$, it can be shown that we need $K \simeq \frac{L_1 T}{\lambda}$ local steps to achieve the desired accuracy condition (6). The proof can be found in Section C.2.1.

Lemma 3.3. *Consider I-CGM with* $\mathbf{x}^{t+1} = \mathrm{CGM}_{\mathrm{const}}(\lambda, K, \mathbf{x}^t, \mathbf{g}^t)$ *under Assumption 2.1 and 2.3. Let* $T \geq 1$ *be the fixed number in condition* (6)*. Then by choosing* $\lambda > \Delta_1$ *and* $K = K_T := \lceil \frac{8L_1 T}{\lambda - \Delta_1} \rceil$*, the accuracy condition* (6) *is satisfied.*

3.2.2 Random Number of Local Steps.

We now allow the number of local steps K_t to follow a geometric distribution—a common technique used to derive last-iterate recurrences [34]. When applied to solve the subproblems in I-CGM, this approach yields an algorithm that is efficient in local computation.

Let us consider CGM (8) with $K_t = \hat{K}_t + 1$ iterations where $\hat{K}_t \sim \mathrm{Geom}(p)$, that is $\mathbb{P}(\hat{K}_t = k) = (1-p)^k p$ for each $k \in \{0, 1, 2, ...\}$. The solution is set to be $\mathbf{x}^{t+1} = \mathbf{y}_{K_t}$. We use the notation $\mathbf{x}^{t+1} = \mathrm{CGM}_{\mathrm{rand}}(\lambda, \hat{K}_t, \mathbf{x}^t, \mathbf{g}^t)$ for this process.

In contrast to the convergence rate of using a deterministic K, we can now show that $\mathbb{E}_{\hat{K}_t}[e_t^2] \lesssim \frac{L_1^2 p}{L_1+\lambda}\mathbb{E}_{\hat{K}_t}[F_t(\mathbf{x}^t) - F_t(\mathbf{x}^{t+1})]$. Using $\mathbb{E}_{\hat{K}_t}[F_t(\mathbf{x}^t) - F_t(\mathbf{x}^{t+1})] \lesssim \mathbb{E}_{\hat{K}_t}[f(\mathbf{x}^t) - f(\mathbf{x}^{t+1}) + \frac{1}{\lambda}\hat{\Sigma}_t^2]$, we get the telescoping term $\mathbb{E}[f(\mathbf{x}^t) - f(\mathbf{x}^{t+1})]$ after passing to the full expectation, which allows to improve the total amount of local computations.

Lemma 3.4. *Consider I-CGM with* $\mathbf{x}^{t+1} = \mathrm{CGM}_{\mathrm{rand}}(\lambda, \hat{K}_t, \mathbf{x}^t, \mathbf{g}^t)$ *where* $\hat{K}_t \sim \mathrm{Geom}(p)$ *under Assumption 2.1 and 2.3. Let* $T \geq 1$ *be the fixed number in condition* (6)*. Then by choosing* $\lambda > \Delta_1$ *and* $p = \frac{\lambda - \Delta_1}{8(L_1+\lambda)} < 1$*, the accuracy condition* (6) *is satisfied.*

To achieve the accuracy condition (6), the number of local first-order oracle queries required by using the random $\hat{K}_t$ at each iteration t in expectation is $\mathbb{E}_{\hat{K}_t}[K_t] = \frac{1}{p} \simeq \frac{L_1}{\lambda}$, which improves upon the previous result of $\frac{L_1 T}{\lambda}$ obtained by using a fixed number of K.

So far, we have studied how to solve the subproblems of I-CGM such that the accuracy condition (6) is satisfied. We now turn to constructing the gradient estimator $\mathbf{g}^t$ that has the desired approximation error (7). Meanwhile, we aim to improve both the dependence on n and C_A in the communication complexity of CGM. The main strategy is to design a gradient estimator whose approximation error depends only on the similarity constant δ while avoiding periodic full synchronizations.

4 Basic Application Examples: SAGA + SVRG

In this section, we present two algorithms that maintain an approximation of the gradient, $\mathbf{G}^t \approx \nabla f(\mathbf{x}_t)$ for $t \geq 0$. Each algorithm starts with an initial point $\mathbf{x}^0$. Then at each iteration $t \geq 0$, $\mathbf{G}^t$ is computed first, after which the next iterate $\mathbf{x}^{t+1}$ is computed. In what follows, for a set $S \in \binom{[n]}{m}$ and $m \in [n]$, we use $f_S := \frac{1}{m}\sum_{i \in S} f_i$ to denote the average function over this set.

For convenience of presentation, we use the following notations throughout the rest of the paper:

$$n_m := \frac{n}{m}, \quad q_m := \frac{n-m}{n-1}, \quad \text{and} \quad \delta_m^2 := \frac{q_m}{m}\delta^2 \,. \tag{9}$$

4.1 SAGA Estimator

SAGA estimator is a variance-reduction technique based on incremental gradient updates, originally designed for centralized finite-sum minimization [17]. In this section, we adapt this estimator to the federated optimization scenario and study its properties.

The SAGA estimator defines:

$$\boxed{\mathbf{G}^0 = \nabla f(\mathbf{x}^0), \quad \mathbf{G}^1 = \nabla f(\mathbf{x}^1), \quad \mathbf{G}^t = \mathbf{b}_{S_t}^t - \mathbf{b}_{S_t}^{t-1} + \mathbf{b}^{t-1},\ t \geq 2\,,} \tag{SAGA}$$

where $S_t \in \binom{[n]}{m}$ is uniformly sampled at random without replacement, $\mathbf{b}_{S_t}^t := \frac{1}{m}\sum_{i\in S_t}\mathbf{b}_i^t$, $\mathbf{b}_{S_t}^{t-1} := \frac{1}{m}\sum_{i\in S_t}\mathbf{b}_i^{t-1}$, $\mathbf{b}^t := \frac{1}{n}\sum_{i=1}^n \mathbf{b}_i^t$, and for any $i \in [n]$, $\mathbf{b}_i^t$ is recurrently defined as:

$$\mathbf{b}_i^0 = \nabla f_i(\mathbf{x}^0), \quad \mathbf{b}_i^1 = \nabla f_i(\mathbf{x}^1), \quad \mathbf{b}_i^t = \begin{cases} \nabla f_i(\mathbf{x}^t) & \text{if } i \in S_t, \\ \mathbf{b}_i^{t-1} & \text{otherwise,} \end{cases}, \quad t \geq 2 \, .$$

We have the following recurrence for $\mathbf{b}^t$ (the derivation can be found in Lemma C.3):

$$\mathbf{b}^t = \mathbf{b}^{t-1} + \frac{1}{n_m}[\nabla f_{S_t}(\mathbf{x}^t) - \mathbf{b}_{S_t}^{t-1}], \quad t \geq 2 \, . \tag{10}$$

Implementation. At the beginning, when $t = 0$ and 1, each client $i = 1, \ldots, n$ computes $\nabla f_i(\mathbf{x}^t)$ and initializes $\mathbf{b}_i^t$ and sends the result to the server; the server then aggregates these results computing $\nabla f(\mathbf{x}^t)$ to initialize $\mathbf{G}^t$ and $\mathbf{b}^t$. This requires two full synchronizations ($2\lceil n_m \rceil$ communications rounds using A-CSS). At each iteration $t \geq 2$, the server contacts the randomly selected set of clients S_t using R-CSS and sends $\mathbf{x}^t$ to them. Each client $i \in S_t$ computes $\mathbf{b}_i^t = \nabla f_i(\mathbf{x}^t)$ and sends $\mathbf{b}_i^t - \mathbf{b}_i^{t-1}$ back to the server. The server then updates $\mathbf{b}^t$ according to (10) and constructs the gradient estimator $\mathbf{G}^t$ using the stored $\mathbf{b}^{t-1}$ according to (SAGA).

Each client i thus needs to store a single vector $\mathbf{b}_i^t$. On the server side, only the aggregated vector $\mathbf{b}^t$ and the iterate $\mathbf{x}^t$ need to be maintained. The memory overhead of the SAGA estimator is thus very small in the federated learning setting, similarly to the SAG estimator [16] used in SCAFFOLD [20].

Properties of SAGA. It is not difficult to show that $\mathbf{G}^t$ is a conditionally unbiased estimator of $\nabla f(\mathbf{x}^t)$, namely, $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$. We next present a new variance bound for SAGA that is controlled by the constant δ. The proof can be found in Section C.3.1.

Lemma 4.1. *Consider the* SAGA *estimator* (SAGA) *under Assumption 2.2. Then for any $t \geq 2$, $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$ and for any $T \geq 1$, we have :*

$$\sum_{t=0}^{T} \sigma_t^2 \leq \frac{2n_m q_m}{m} G_1^2 + \frac{n_m - 1 + \sqrt{n_m^2 - n_m}}{(n-1)} \sum_{t=2}^{T-1} G_t^2 + 4n_m^2\delta_m^2 \sum_{t=2}^{T} \chi_t^2 \, ,$$

where $\sigma_t^2 := \mathbb{E}_{S_{[t]}}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2]$, $G_t^2 = \mathbb{E}_{S_{[t-1]}}[\|\nabla f(\mathbf{x}^t)\|^2]$, $\chi_t^2 := \mathbb{E}_{S_{[t-1]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2]$, and $S_{[t]} := (S_2, ..., S_t)$.

Note that this variance bound depends on $G_1^2, \ldots, G_{T-1}^2$ and $\chi_2^2, \ldots, \chi_T^2$, which aligns with the terms on the right-hand side of the desired error bound (7). However, the coefficient in front of $\mathbf{G}_1^2$ in this bound can be larger than 1, whereas (7) requires it to be strictly less than 1. Consequently, the requirement is not met and we cannot directly incorporate the SAGA estimator into I-CGM by setting $\mathbf{g}^t = \mathbf{G}^t$. We will show in Section 5 that this error bound can be significantly improved by using the recursive gradient estimation technique.

Remark 4.2. Instead of computing the exact gradients $\nabla f(x^0)$ and $\nabla f(\mathbf{x}^1)$ at the beginning which requires full synchronizations, it is possible to start with an approximation $\mathbf{G}^0 \approx \nabla f(\mathbf{x}^0)$. This requires only one communication round using R-CSS. The resulting communication-complexity estimate will now additionally depend on the inexactness of the initial approximation but this strategy often works well in practice.

4.2 SVRG Estimator

Another possible choice of the gradient estimator is the SVRG estimator [35]. There are different variants of SVRG, and here we consider the so-called loopless-SVRG estimator [23] for simplicity.

The SVRG estimator defines:

$$\boxed{\mathbf{G}^0 = \nabla f(\mathbf{x}^0), \quad \mathbf{G}^t = \nabla f_{S_t}(\mathbf{x}^t) + \nabla f(\mathbf{w}^t) - \nabla f_{S_t}(\mathbf{w}^t), \; t \geq 1 \, ,} \tag{SVRG}$$

where $S_t \in \binom{[n]}{m}$ is uniformly sampled at random without replacement,

$$\mathbf{w}^0 = \mathbf{x}^0, \quad \mathbf{w}^t = \begin{cases} \mathbf{x}^t & \text{if } \omega_t = 1, \\ \mathbf{w}^{t-1} & \text{otherwise,} \end{cases} \quad t \geq 1 \, ,$$

and ω_t is a Bernoulli random variable with parameter p_B, i.e., $P(\omega_t = 1) = p_B \in (0, 1)$.

Implementation. At the beginning when $t = 0$, each client $i = 1, \ldots, n$ computes $\nabla f_i(\mathbf{x}^0)$ and sends the result to the server; the server then aggregates these results, computing $\nabla f(\mathbf{x}^0)$ to initialize $\mathbf{G}^0$. This requires one full synchronization. At each iteration $t \geq 1$, the server uses R-CSS and sends $\mathbf{w}^t$ and $\mathbf{x}^t$ to the clients in S_t and then receives the gradient difference $\nabla f_i(\mathbf{x}^t) - \nabla f_i(\mathbf{w}^t)$ from them. If $\omega_t = 1$, the server computes the new gradient $\nabla f(\mathbf{w}^t)$ performing one full synchronization and stores it in memory; otherwise, it continues with $\nabla f(\mathbf{w}^t) = \nabla f(\mathbf{w}^{t-1})$ which is already stored in memory. In total, the server needs to maintain two points $\mathbf{x}^t$ and $\mathbf{w}^t$ and one vector $\nabla f(\mathbf{w}^t)$ and the clients are so-called stateless.

Properties: It is not difficult to show that the SVRG estimator $\mathbf{G}^t$ is a conditionally unbiased estimator of $\nabla f(\mathbf{x}^t)$, namely, $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$. Moreover, the variance is controlled by δ. The proof can be found in Section C.3.2.

Lemma 4.3. *Consider the* SVRG *estimator* (SVRG) *under Assumption 2.2. Then for any* $t \geq 1$, $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$ *and for any* $T \geq 1$, *we have:* $\sum_{t=0}^{T} \sigma_t^2 \leq \frac{4\delta_m^2}{p_B^2} \sum_{t=1}^{T} \chi_t^2$, *where* $\sigma_t^2 := \mathbb{E}_{S_t, \omega_{[t]}}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2]$, $\chi_t^2 := \mathbb{E}_{\omega_{[t-1]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2]$, *and* $\omega_{[t]} := (\omega_1, ..., \omega_t)$.

We can incorporate the SVRG estimator into I-CGM by setting $\mathbf{g}^t = \mathbf{G}^t$. This requires setting $p_B \simeq \frac{1}{n_m}$ and $\lambda \simeq \Delta_1 + n_m \delta_m$ to achieve the error condition (7). The resulting communication complexity of the method is $\mathcal{O}(C_A n_m + \frac{(C_R \Delta_1 + C_A n_m \delta_m) F^0}{\varepsilon^2})$, which still has a linear dependence on n_m. (See Theorem C.4 with the proof that the reader can inspect if interested). Note that unbiasedness is not needed to incorporate SVRG directly into I-CGM. However, it becomes necessary later for the recursive gradient technique, which we discuss in the next section.

5 Recursive Gradient Estimator + Examples (SAGA and SVRG)

In this section, we present a general formular of the recursive gradient estimator that can potentially improve the error bound for a given conditionally unbiased gradient estimator $\mathbf{G}^t \approx \nabla f(\mathbf{x}^t)$. Formally, we consider the following setting.

Assumption 5.1. For any $t \geq 0$, it holds that: 1) S_t is independent of $\mathbf{x}^0, \ldots, \mathbf{x}^{t+1}, \mathbf{G}^0, \ldots, \mathbf{G}^{t-1}$; 2) $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$.

The recursive gradient estimator (RG) defines:

$$\boxed{\mathbf{g}^0 = \nabla f(\mathbf{x}^0), \quad \mathbf{g}^{t+1} = (1-\beta)\mathbf{g}^t + \beta \mathbf{G}^t + \nabla f_{S_t}(\mathbf{x}^{t+1}) - \nabla f_{S_t}(\mathbf{x}^t),\ t \geq 0\ ,} \qquad \text{(RG)}$$

where $\beta \in (0, 1]$ and $S_t \in \binom{[n]}{m}$ is uniformly sampled at random without replacement.

Note that the indexing here differs from the previous ones. The algorithm starts with an initial point $\mathbf{x}^0$. At each iteration $t \geq 0$, the estimator $\mathbf{g}^t \approx \nabla f(\mathbf{x}^t)$ is computed first and it depends only on $\mathbf{G}^{t-1}$ and S_{t-1}. After that, the next iterate $\mathbf{x}^{t+1}$ is computed. Therefore, $\mathbf{x}^{t+1}$ is independent from S_t while previously it was dependent on it (if we use the SAGA/SVRG estimator).

Inspired by previous works, the expression of $\mathbf{g}^t$ incorporates both recursive gradient update and momentum [36, 37]. This expression unifies several existing methods: When $\beta = 0$, the estimator reduces to the SARAH update rule [10]. When $\mathbf{G}^t$ is replaced with the SAGA estimator, then $\mathbf{g}^t$ recovers the structure of ZEROSARAH [19]. When $\mathbf{G}^t = \nabla f_{S_t}(\mathbf{x}^{t+1})$ and $\nabla f_{S_t}(\mathbf{x}^{t+1}) - \nabla f_{S_t}(\mathbf{x}^t)$ is multiplied by $1 - \beta$, then it becomes STORM [38]. In our formulation, $\mathbf{G}^t$ is a general similarity-aware estimator of $\nabla f(\mathbf{x}^t)$ that satisfies Assumption 5.1, allowing us to flexibly instantiate the framework with various variance-reduction techniques.

For instance, we can combine RG with SAGA or SVRG. We refer to the resulting estimators as RG-SAGA and RG-SVRG. Note that S_t in the formulas for SAGA and SVRG is exactly the same random index set that is used in the RG – they share the same randomness for the sake of efficiency.

Implementation. At the beginning, each client $i = 1, \ldots, n$ computes $\nabla f_i(\mathbf{x}^0)$ and sends the result to the server; the server then aggregates these results, computing $\nabla f(\mathbf{x}^0)$ to initialize $\mathbf{g}^0$. This requires one full synchronization. Then $\mathbf{x}^1$ is computed based on $\mathbf{g}^0$. At each iteration $t \geq 0$, the server uses R-CSS which generates a random subset S_t. For RG-SAGA, the server sends $\mathbf{x}^{t+1}$, $\mathbf{x}^t$

to the clients in S_t. Each client $i \in S_t$ updates $\mathbf{b}_i^t = \nabla f_i(\mathbf{x}^t)$ and sends $\nabla f_i(\mathbf{x}^{t+1})$ along with $\mathbf{b}_i^t - \mathbf{b}_i^{t-1}$ (when $t \geq 2$) or $\mathbf{b}_i^t$ (when $t = 1$) to the server. For RG-SVRG, the server sends $\mathbf{x}^{t+1}$, $\mathbf{x}^t$ and $\mathbf{w}^t$ to the clients which then return the gradients evaluated at these three points. If $\omega_t = 1$, the server additionally computes the new gradient $\nabla f(\mathbf{w}^t)$ performing one full synchronization. After receiving all the vectors, the server can compute $\nabla f_{S_t}(\mathbf{x}^{t+1})$, $\nabla f_{S_t}(\mathbf{x}^t)$, $\mathbf{G}^t$ and $\mathbf{g}^{t+1}$.

For RG-SAGA, each client i needs to store a single vector $\mathbf{b}_i^t$ and the server needs to maintain two points $\mathbf{x}^{t+1}$ and $\mathbf{x}^t$, and two vectors $\mathbf{b}^t$ and $\mathbf{g}^t$. For RG-SVRG, clients are stateless and the server is required to maintain three points $\mathbf{x}^{t+1}$, $\mathbf{x}^t$, $\mathbf{w}^t$ and one vector $\nabla f(\mathbf{w}^t)$.

Lemma 5.2 (Error bound for RG)**.** *Consider the RG estimator* (RG) *under Assumptions 5.1 and 2.2. Then for any $T \geq 1$, we have:*

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{2\beta}{2-\beta} \sum_{t=0}^{T-1} \sigma_t^2 + \frac{2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \ .$$

where $\Sigma_t^2 := \mathbb{E}_{S_{[t-1]}}[\|\mathbf{g}^t - \nabla f(\mathbf{x}^t)\|^2]$, $\sigma_t^2 := \mathbb{E}_{S_{[t]}}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2]$, $\chi_t^2 := \mathbb{E}_{S_{[t-2]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2]$, *and* $S_{[t]} := (S_0, \ldots, S_t)$.

The proof can be found in Section C.4.1. We next show that the error bound of both SAGA and SVRG can be improved by combining them with RG and adjusting the parameter β. For instance, by combining Lemma 5.2 and Lemma 4.1, we obtain the following result for RG-SAGA.

Corollary 5.3. *Consider the RG-SAGA estimator under Assumptions 5.1 and 2.2. Then for any $T \geq 1$, it holds that:*

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{4\beta n_m q_m}{(2-\beta)m} G_1^2 + \frac{2\beta(n_m - 1 + \sqrt{n_m^2 - n_m})}{(2-\beta)(n-1)} \sum_{t=2}^{T-1} G_t^2 + \frac{8\beta^2 n_m^2 \delta_m^2 + 2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \ ,$$

where $\Sigma_t^2 := \mathbb{E}_{S_{[t-1]}}[\|\mathbf{g}^t - \nabla f(\mathbf{x}^t)\|^2]$, $G_t^2 := \mathbb{E}_{S_{[t-2]}}[\|\nabla f(\mathbf{x}^t)\|^2]$, $\chi_t^2 := \mathbb{E}_{S_{[t-2]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2]$ *and* $S_{[t]} := (S_0, \ldots, S_t)$.

By choosing $\beta \simeq \frac{1}{n_m}$, we get $\sum_{t=0}^{T} \Sigma_t^2 \lesssim \frac{q_m}{m} G_1^2 + \frac{1}{n} \sum_{t=2}^{T-1} G_t^2 + n_m \delta_m^2 \sum_{t=1}^{T} \chi_t^2$. Compared with the original variance bound for SAGA (Lemma 4.1), the bound with RG achieves an improvement by a factor of n_m.

The error bound for the SVRG estimator can be improved in a similar way.

Corollary 5.4. *Consider the RG-SVRG estimator under Assumptions 5.1 and 2.2. Then for any $T \geq 1$, it holds that:*

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{8\beta^2 \delta_m^2 / p_B^2 + 2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \ ,$$

where $\Sigma_t^2 := \mathbb{E}_{S_{[t-1]}, \omega_{[t-1]}}[\|\mathbf{g}^t - \nabla f(\mathbf{x}^t)\|^2]$, $\chi_t^2 := \mathbb{E}_{S_{[t-2]}, \omega_{[t-2]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2]$, $S_{[t]} := (S_0, \ldots, S_t)$ *and* $\omega_{[t]} := (\omega_1, \ldots, \omega_t)$.

Compared with the original variance bound for SVRG (Lemma 4.3), the new bound achieves an improvement by a factor of $1/p_B$ by choosing $\beta \simeq p_B$.

We can now incorporate both enhanced estimators into I-CGM. It can be shown that the iterates $\{\mathbf{x}^t\}_{t=0}^{\infty}$ generated by I-CGM-RG-SAGA or I-CGM-RG-SVRG and the corresponding sequence $\{\mathbf{G}^t\}_{t=0}^{\infty}$ satisfy Assumption 5.1. (See Lemma C.5 and C.6).

6 Communication and Local Complexity of I-CGM-RG

We are ready to establish the complexity of I-CGM equipped with the RG-SAGA and RG-SVRG estimator. We first present the result for RG-SAGA. The proof can be found in Section C.5.1.

Theorem 6.1 (I-CGM-RG-SAGA)**.** *Let I-CGM be applied to Problem 1 under Assumptions 2.1, 2.2 and 2.3, where $\mathbf{x}^{t+1} = \mathrm{CGM}_{\mathrm{rand}}(\lambda, \hat{K}_t, \mathbf{x}^t, \mathbf{g}^t)$ with $\hat{K}_t \sim \mathrm{Geom}(p)$ and $\mathbf{g}^t$ is generated by the RG-SAGA estimator. Then by choosing $\lambda = 3\Delta_1 + 113\sqrt{n_m}\delta_m$, $\beta = \frac{1}{112 n_m}$ and $p = \frac{\lambda - \Delta_1}{8(L_1 + \lambda)}$, after $T = \left\lceil \frac{(256(\Delta_1 + 38\sqrt{n_m}\delta_m)F^0}{\varepsilon^2} \right\rceil$ iterations, we have $\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \varepsilon^2$, where $\bar{\mathbf{x}}^T$ is is uniformly sampled*

from $(\mathbf{x}^t)_{t=1}^T$. *The communication complexity is at most* $2C_A\lceil n_m\rceil+(C_R+1)\lceil\frac{(256(\Delta_1+38\sqrt{n_m}\delta_m)F^0}{\varepsilon^2}\rceil$ *and the local complexity is bounded by* $14+2\lceil n_m\rceil+\frac{512(7\Delta_1+283\sqrt{n_m}\delta_m+2L_1)F^0}{\varepsilon^2}+\frac{4L_1}{\Delta_1+28\sqrt{n_m}\delta_m}$.

The communication complexity of I-CGM-RG-SAGA is of order $C_An_m+C_R\frac{\Delta_1+(\sqrt{n_m}\delta_m)F^0}{\varepsilon^2}$ and the local complexity is of order $n_m+\frac{(\Delta_1+\sqrt{n_m}\delta_m+L_1)F^0}{\varepsilon^2}$ when $\frac{(\Delta_1+\sqrt{n_m}\delta_m)F^0}{\varepsilon^2}\gtrsim 1$. The n_m term comes from n_m sequential rounds with A-CSS for computing the full gradients in the beginning.

Comparison: RG-SVRG Estimator. The communication complexity of I-CGM-RG-SVRG is $\mathcal{O}\big(C_An_m+\frac{(C_R\Delta_1+\sqrt{C_AC_Rn_m}\delta_m)F^0}{\varepsilon^2}\big)$, where C_A also affects the term involving ε (see Appendix C.6 for details and the result of the local complexity).

7 Numerical experiments

In this section, we verify the theory of the proposed methods in numerical experiments. We set $C_A=C_R=1$ in the definition of communication complexity for all the experiments. We choose this case to demonstrate that even when A-CSS and R-CSS are equally cheap, our proposed methods already outperform several commonly used algorithms. (The study of the scenario when $C_A>C_R$ can be found in Appendix E.1.1.)

Quadratic minimization with log-sum penalty. Consider the problem of $f(\mathbf{x})=\frac{1}{n}\sum_{i=1}^n f_i(\mathbf{x})$ with $f_i(\mathbf{x}):=\frac{1}{b}\sum_{j=1}^b\frac{1}{2}\langle\mathbf{A}_{i,j}(\mathbf{x}-\mathbf{b}_{i,j}),\mathbf{x}-\mathbf{b}_{i,j}\rangle+\sum_{k=1}^d\log\big(1+\alpha|\mathbf{x}_k|\big)$, where $\alpha>0$, $\mathbf{b}_{i,j}\in\mathbb{R}^d$, $\mathbf{A}_{i,j}\in\mathbb{R}^{d\times d}$ is a diagonal matrix, and $\cdot_k$ is an indexing operation of a vector. We set $\alpha=10$, $b=5$, $n=100$ and $d=1000$. Each coordinate of $\mathbf{b}_{i,j}$ is uniformly sampled from $[0,10]$. To generate $\mathbf{A}_{i,j}$, we first sample a diagonal matrix $\bar{\mathbf{A}}$ with entries uniformly distributed in $[0,110]$, and then add bn diagonal noise matrices whose entries are sampled from $[0,18]$. Each resulting $\mathbf{A}_{i,j}$ is clipped to the interval $[1,100]$ on the diagonal, and some eigenvalues are further set close to zero. Consequently, the dataset satisfies $\mathbf{0}\preceq\mathbf{A}_{i,j}\preceq 100\mathbf{I}$ for any i,j, with $\Delta_1\approx\delta\approx 5$ and $L_{\max}\approx 100$. We set $m=\sqrt{n}$. For I-CGM-RG, we set $p=\frac{\delta}{L}$, $\lambda=\frac{\sqrt{n}}{m}\delta+\Delta_1$, $\eta=2L_{\max}$ and $\beta=\frac{m}{n}$. We compare two proposed algorithms against SCAFFOLD [20], FEDAVG [1] (with sampling), SABER-FULL [12] (with PAGE), SABER-PARTIAL [12] (only compute full gradient once) and GD (running directly on f). For SVRG-based methods, the expected number of communication rounds at each iteration is roughly $\frac{m}{n}n+m$, which is twice as large as other methods. From Figure 2, we observe that: 1) I-CGM-RG-SAGA is the most efficient in both communication and local computation. 2) SCAFFOLD cannot fully exploit δ-similarity as its local complexity is comparable to GD (the theoretical local complexity of both methods depends on $L_{\max}$). Finally, I-CGM-RG-SAGA with different initialization strategies can be found in Figure E.1.

Logistic regression with nonconvex regularizer. We now experiment with the binary classification task on two real-world LIBSVM datasets [39]. We use the standard regularized logistic loss: $f(\mathbf{x})=\frac{1}{n}\sum_{i=1}^n f_i(\mathbf{x})$ with $f_i(\mathbf{x}):=\frac{n}{M}\sum_{j=1}^{m_i}\log(1+\exp(-y_{i,j}\langle\mathbf{a}_{i,j},\mathbf{x}\rangle))+\alpha\sum_{k=1}^d\frac{[\mathbf{x}]_k^2}{1+[\mathbf{x}]_k^2}$ where $\alpha>0$, $(\mathbf{a}_{i,j},y_{i,j})\in\mathbb{R}^{d+1}$ are feature and labels and $M:=\sum_{i=1}^n m_i$ is the total number of data points. We use $m=1$ and $n=10$. We plot the local L_1 and δ by computing $\big\|\nabla f_1(\mathbf{x}^t)-\nabla f_1(\mathbf{x}^{t+1})\big\|/\big\|\mathbf{x}^t-\mathbf{x}^{t+1}\big\|$ and $\sqrt{\frac{1}{n}\sum_{i=1}^n\|\nabla h_i(\mathbf{x}^t)-\nabla h_i(\mathbf{x}^{t+1})\|^2/\|\mathbf{x}^t-\mathbf{x}^{t+1}\|^2}$ along the iterates of I-CGM-RG-SAGA. We observe that δ is much smaller than L_1 for the mushrooms dataset, while being comparable for the duke dataset. However, for both cases, I-CGM-RG-SAGA remains the most efficient in communication complexity.

Deep learning tasks. We defer the study of neural network training to Appendix E, where more experiments and details can be found.

8 Conclusion

We introduced a new simple model for comparing federated optimization algorithms, where different client-selection strategies are associated with non-uniform costs. Within this model, we developed a new family of algorithm based on inexact composite gradient method with recursive gradient estimator.

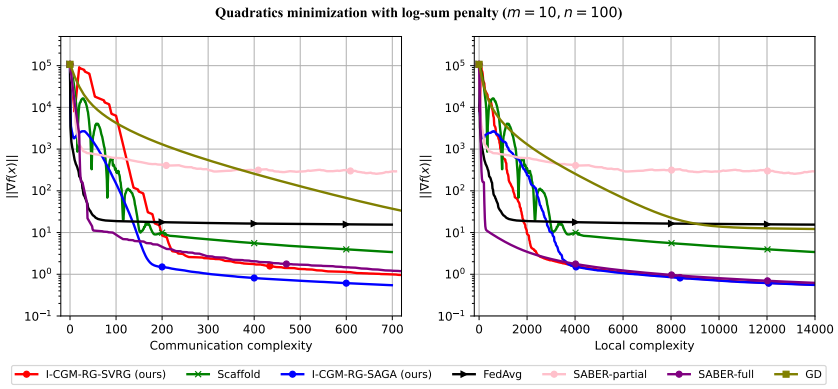

Figure 2: Comparisons of different algorithms for solving the quadratic minimization problems with non-convex log-sum penalty.

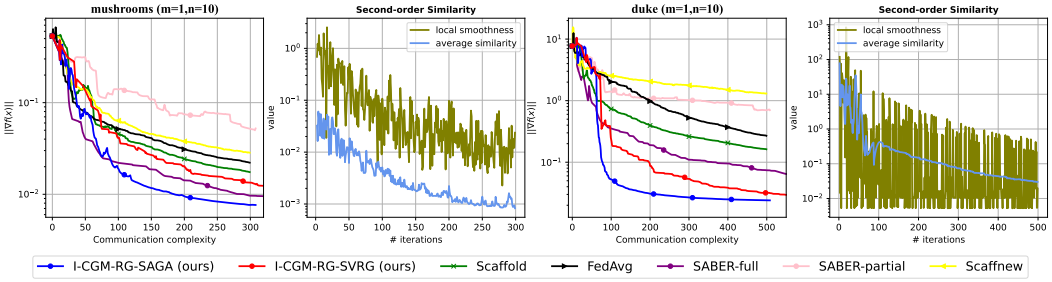

Figure 3: Comparisons of different algorithms on two LIBSVM datasets using logistic loss with non-convex regularizer.

This design enables us to exploit functional similarity among clients while supporting partial client participation—a key requirement in practical FL systems. It is efficient when full synchronizations (requiring sequential communications with all clients) are costly compared to client sampling. The key technical contribution of this work is a new variance bound for the SAGA estimator, which depends on the functional dissimilarity constant δ rather than individual smoothness. This allows the SAGA-based variant of I-CGM-RG to outperform the previously best-known communication complexity of SARAH-based methods.

Limitations and future extensions. 1) In this work, we have assumed that there exists one delegate client that is reliable for communication. If we modify the setting and remove this delegate client, then we can still guarantee similar complexity with minor modifications. Specifically, instead of fixing the index 1 in I-CGM, we can sample $i_t \in [n]$ uniformly at random and define the updates as $\mathbf{x}^{t+1} \approx \arg\min_{\mathbf{x}\in\mathbb{R}^d} \left\{ F_t(\mathbf{x}) := f_{i_t}(\mathbf{x}) + h_{i_t}(\mathbf{x}^t) + \langle \mathbf{g}^t - \nabla f_{i_t}(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t \rangle + \frac{\lambda}{2} \|\mathbf{x} - \mathbf{x}^t\|^2 \right\}$. This variant uses R-CSS instead of D-CSS at each iteration. To ensure the convergence rate of $\frac{\lambda F^0}{T}$, we need to choose $\lambda \simeq \Delta_{\max}$ [21], where $\Delta_{\max} \lesssim \Delta_1$ is defined in A.3. However, suppose there exists more than one delegate client, then it is interesting to check if we can further improve the current complexity. 2) We have shown that the variance of the SAGA estimator is bounded by the function dissimilarity constant δ. An interesting question is whether something similar can be done for another closely related popular gradient estimator, SAG [16], used in Scaffold [20]. It turns out that the answer is negative (see Section D). 3) Our analysis focuses on the deterministic first-order oracle $\mathrm{O}_{f_i} = \mathrm{O}_{\mathrm{FO}_i}$. It is interesting to develop efficient algorithms with stochastic, zero-order, or higher-order oracles. 4) The current model does not impose constraints on the size of information that is transmitted between the server and clients. A promising direction is to incorporate communication compression and study how such constraints affect the algorithm design and overall complexity.

References

[1] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[2] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and Open Problems in Federated Learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[3] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated Learning: Strategies for Improving Communication Efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[4] Blake E Woodworth, Jialei Wang, Adam Smith, Brendan McMahan, and Nati Srebro. Graph Oracle Models, Lower Bounds, and Gaps for Parallel Stochastic Optimization. *Advances in neural information processing systems*, 31, 2018.

[5] Janne H Korhonen and Dan Alistarh. Towards Tight Communication Lower Bounds for Distributed Optimisation. *Advances in Neural Information Processing Systems*, 34:7254–7266, 2021.

[6] Kumar Kshitij Patel, Lingxiao Wang, Blake E Woodworth, Brian Bullins, and Nati Srebro. Towards Optimal Communication Complexity in Distributed Non-Convex Optimization. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[7] Yuchen Zhang, John Duchi, Michael I Jordan, and Martin J Wainwright. Information-Theoretic Lower Bounds for Distributed Statistical Estimation with Communication Constraints. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[8] Peter Davies, Vijaykrishna Gurunathan, Niusha Moshrefi, Saleh Ashkboos, and Dan Alistarh. New Bounds for Distributed Mean Estimation and Variance Reduction. *arXiv preprint arXiv:2002.09268*, 2020.

[9] Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal Convergence Rates for Convex Distributed Optimization in Networks. *Journal of Machine Learning Research*, 20(159):1–31, 2019.

[10] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A Novel Method for Machine Learning Problems using Stochastic Recursive Gradient. In *International conference on machine learning*, pages 2613–2621. PMLR, 2017.

[11] Zhize Li, Hongyan Bao, Xiangliang Zhang, and Peter Richtárik. PAGE: A Simple and Optimal Probabilistic Gradient Estimator for Nonconvex Optimization. In *International conference on machine learning*, pages 6286–6295. PMLR, 2021.

[12] Konstantin Mishchenko, Rui Li, Hongxiang Fan, and Stylianos Venieris. Federated Learning Under Second-Order Data Heterogeneity, 2024.

[13] Ahmed Khaled and Chi Jin. Faster Federated Optimization under Second-Order Similarity. In *The Eleventh International Conference on Learning Representations*, 2023.

[14] El Mahdi Chayti and Sai Praneeth Karimireddy. Optimization with Access to Auxiliary Information. *arXiv preprint arXiv:2206.00395*, 2022.

[15] Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. Breaking the Centralized Barrier for Cross-Device Federated Learning. In *Advances in Neural Information Processing Systems*, 2021.

[16] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing Finite Sums with the Stochastic Average Gradient. *Mathematical Programming*, 162:83–112, 2017.

[17] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A Fast Incremental Gradient Method with Support for Non-Strongly Convex Composite Objectives. *Advances in neural information processing systems*, 27, 2014.

[18] Sashank J Reddi, Suvrit Sra, Barnabás Póczos, and Alex Smola. Fast Incremental Method for Nonconvex Optimization. *arXiv preprint arXiv:1603.06159*, 2016.

[19] Zhize Li, Slavomír Hanzely, and Peter Richtárik. ZeroSARAH: Efficient Nonconvex Finite-Sum Optimization with Zero Full Gradient Computation. *arXiv preprint arXiv:2103.01447*, 2021.

[20] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *International conference on machine learning*, pages 5132–5143. PMLR, 2020.

[21] Xiaowen Jiang, Anton Rodomanov, and Sebastian U Stich. Federated Optimization with Doubly Regularized Drift Correction. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 21912–21945. PMLR, 21–27 Jul 2024.

[22] 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.

[23] Dmitry Kovalev, Samuel Horváth, and Peter Richtárik. Don't Jump Through Hoops and Remove Those Loops: SVRG and Katyusha are Better Without the Outer Loop. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, volume 117 of *Proceedings of Machine Learning Research*, pages 451–467. PMLR, 2020.

[24] A. S. Nemirovsky. Information-Based Complexity of Convex Programming. 1994.

[25] A. S. Nemirovsky and D. B. Yudin. Problem Complexity and Method Efficiency in Optimization. 1983.

[26] Yossi Arjevani, Amit Daniely, Stefanie Jegelka, and Hongzhou Lin. On the complexity of minimizing convex finite sums without using the indices of the individual functions. *arXiv preprint arXiv:2002.03273*, 2020.

[27] Xiaowen Jiang, Anton Rodomanov, and Sebastian U Stich. Stabilized Proximal-Point Methods for Federated Optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[28] Dachao Lin, Yuze Han, Haishan Ye, and Zhihua Zhang. Stochastic Distributed Optimization under Average Second-Order Similarity: Algorithms and Analysis. *Advances in Neural Information Processing Systems*, 36, 2024.

[29] Yuki Takezawa, Xiaowen Jiang, Anton Rodomanov, and Sebastian U Stich. Exploiting Similarity for Computation and Communication-Efficient Decentralized Optimization. 2025.

[30] Hadrien Hendrikx, Lin Xiao, Sebastien Bubeck, Francis Bach, and Laurent Massoulie. Statistically Preconditioned Accelerated Gradient Method for Distributed Optimization. In *International conference on machine learning*, pages 4203–4227. PMLR, 2020.

[31] 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.

[32] Nhan H Pham, Lam M Nguyen, Dzung T Phan, and Quoc Tran-Dinh. ProxSARAH: An Efficient Algorithmic Framework for Stochastic Composite Nonconvex Optimization. *Journal of Machine Learning Research*, 21(110):1–48, 2020.

[33] Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. SpiderBoost and Momentum: Faster Variance Reduction Algorithms. *Advances in Neural Information Processing Systems*, 32, 2019.

[34] Zeyuan Allen-Zhu. Katyusha x: Practical Momentum Method for Stochastic Sum-of-Nonconvex Optimization. *arXiv preprint arXiv:1802.03866*, 2018.

[35] Rie Johnson and Tong Zhang. Accelerating Stochastic Gradient Descent using Predictive Variance Reduction. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[36] El Mahdi Chayti, Nikita Doikov, and Martin Jaggi. Improving Stochastic Cubic Newton with Momentum. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 1441–1449. PMLR, 03–05 May 2025.

[37] Yuan Gao, Anton Rodomanov, and Sebastian U. Stich. Non-Convex Stochastic Composite Optimization with Polyak Momentum. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[38] Ashok Cutkosky and Francesco Orabona. Momentum-Based Variance Reduction in Non-Convex SGD. *Advances in neural information processing systems*, 32, 2019.

[39] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[40] Mark Braverman, Ankit Garg, Tengyu Ma, Huy L Nguyen, and David P Woodruff. Communication Lower Bounds for Statistical Estimation Problems via a Distributed Data Processing Inequality. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 1011–1020, 2016.

[41] Ankit Garg, Tengyu Ma, and Huy L Nguyen. On Communication Cost of Distributed Statistical Estimation and Dimensionality. *Advances in Neural Information Processing Systems*, 27, 2014.

[42] Yossi Arjevani and Ohad Shamir. Communication Complexity of Distributed Convex Learning and Optimization. *Advances in neural information processing systems*, 28, 2015.

[43] Jason D. Lee, Qihang Lin, Tengyu Ma, and Tianbao Yang. Distributed Stochastic Variance Reduced Gradient Methods by Sampling Extra Data with Replacement. *Journal of Machine Learning Research*, 18(122):1–43, 2017.

[44] Blake Woodworth. The Minimax Complexity of Distributed Optimization. *arXiv preprint arXiv:2109.00534*, 2021.

[45] Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-Efficient Distributed Optimization using an Approximate Newton-Type Method. In *International conference on machine learning*, pages 1000–1008. PMLR, 2014.

[46] Yuan Gao, Yuki Takezawa, and Sebastian U Stich. A Bias Correction Mechanism for Distributed Asynchronous Optimization. *Transactions on Machine Learning Research*, 2025.

[47] Yurii Nesterov. *Lectures on Convex Optimization*. Springer Publishing Company, Incorporated, 2nd edition, 2018.

[48] Zeyuan Allen-Zhu. Katyusha x: Practical Momentum Method for Stochastic Sum-of-Nonconvex Optimization. *arXiv preprint arXiv:1802.03866*, 2018.

[49] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. EMNIST: Extending MNIST to Handwritten Letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[50] Yida Yin, Zhiqiu Xu, Zhiyuan Li, Trevor Darrell, and Zhuang Liu. A Coefficient Makes SVRG Effective. In *The Thirteenth International Conference on Learning Representations*, 2025.

[51] 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*, pages 15750–15769. PMLR, 2022.

[52] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR-10 (Canadian Institute for Advanced Research).

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

Appendix

Contents

A Related Work

A.1 Formalization of Federated Optimization Algorithms and Their Complexity

Several prior works have proposed oracle models and complexity metrics for distributed and federated optimization. These works typically consider solving the same problem as in (1), where each of the n clients or workers only has access to its own local function. The primary distinctions among these models lie in how communication and computation are formalized. One line of work focuses on the settings where each worker can compute arbitrary information about its local objective, but only a limited number of bits is allowed to transmitted during each communication round [40, 41, 7]. In this setting, complexity is often defined as the number of rounds required to reach a target accuracy. Alternative models remove this constraint and instead measure the total number of bits communicated over the entire optimization process [5]. Other works impose structural restrictions on the communicated information, such as requiring exchanged vectors to lie in a certain subspace (e.g., linear combinations of local gradients) [42, 43].

The closest related model to ours is the Graph Oracle Model (GOM) [4, 6, 44]. GOM introduces a computation and communication graph that determines how each device queries its local oracle and how the computed information propagates through the devices during optimization. Once the oracle and the graph structure are fixed, we obtain a specific model that allows to define the corresponding optimization algorithms. A commonly studied setting is the intermittent communication model, where n devices work in parallel and synchronize after every K local oracle queries. This setting becomes conceptually equivalent to our model when 1) all clients participate in every round, 2) the number of local oracle queries K_r is uniformly bounded across all communication rounds, and 3) the server and the clients are allowed to send its entire accumulated information.

Beyond this scenario, there are several main differences between GOM and our proposed model. 1) GOM does not distinguish between different client-selection strategies that might have non-uniform associated costs. 2) Even when all the strategies have the same cost, GOM fixes the maximum number of local oracle queries for each round, whereas our model allows K_r to vary across rounds. This flexibility enables modeling algorithms such as DANE [45, 21], which needs to solve each local subproblem sufficiently accurately, making K_r dependent on the round r. 3) While partial client participation can be modeled in GOM by generating a random graph, the resulting algorithms are generally restricted to using pre-specified groups of clients at each round. This effectively enforces an offline client-selection strategy for GOM, since it may not account for the need to know the past responses of specific clients before deciding which clients to contact next. In contrast, our model fully supports online and adaptive client-selection strategies.

We believe that our model is reasonably simple and it appears to sufficiently capture how federated optimization algorithms work in practice. Even in the cases where our model is mathematically equivalent to existing ones, our model could still be more convenient to be used.

A.2 Comparison with Existing Federated Optimization Methods

Notations in Table 1. We denote $F^0 := f(\mathbf{x}^0) - f^\star$, $n_m := \frac{n}{m}$, $\delta_m^2 := \frac{n-m}{n-1}\frac{\delta^2}{m}$, $\zeta_m^2 := \frac{\zeta^2}{m}$, and $1 \lesssim C_R \lesssim C_A$ are the costs of communicating with a random set of m clients and a specific set of m clients, respectively.

In this section, we compare our proposed methods with several popular federated optimization algorithms in terms of their communication and local complexities (Section 2.1). For simplicity and to ensure fair comparisons across algorithms, we assume that all methods use the deterministic first-order oracle locally, i.e., $\mathrm{O}_{f_i} = \mathrm{O}_{\mathrm{FO}_i}$, for all $i \in [n]$. We first state and discuss the assumptions under which each algorithm was analyzed in the literature. The abbreviations used in Table 1 are defined as follows.

Assumption A.1 (FS (Function Smoothness))**.** There exists $L_f > 0$ such that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have: $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L_f \|\mathbf{x} - \mathbf{y}\|$.

Assumption A.2 (IS (Individual Smoothness))**.** There exists $L_{\max} > 0$ such that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ and any $i \in [n]$, we have: $\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L_{\max} \|\mathbf{x} - \mathbf{y}\|$.

Assumption A.3 (SD (Second-order Dissimilarity) [20, 21, 46])**.** There exists $\Delta_{\max} > 0$ such that for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ and any $i \in [n]$, we have: $\|\nabla h_i(\mathbf{x}) - \nabla h_i(\mathbf{y})\| \leq \Delta_{\max} \|\mathbf{x} - \mathbf{y}\|$.

Assumption A.4 (BGD (Bounded Gradient Dissimilarity))**.** There exists $\zeta > 0$ such that for any $\mathbf{x} \in \mathbb{R}^d$, we have: $\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|^2 \leq \zeta^2$.

Note that the problem class of IS belong to SD, and SD implies Assumption 2.1 and 2.2. Moreover, any functions that satisfy Assumption 2.1 and 2.3 belong to the class of FS. Finally, the class of FS partially overlaps with the problem class defined by Assumptions 2.1 and 2.2. In Table 1, the assumption under which Centralized GD is analyzed is the most general one, and those for I-CGM-RG are the second most general. Finally, the smoothness constants satisfy the following relations:

$$\delta, \Delta_1 \lesssim \Delta_{\max} \lesssim L_{\max}, \quad L_1, L_f \lesssim L_{\max} \ .$$

We next briefly describe each method in Table 1 and discuss how the communication and local complexities are computed for each method. There are two operations that are commonly used in these methods. We describe them here to avoid repetition later. Denote $n_m := n/m$. The first operation is to compute the full gradient ∇f at the server at a certain point $\mathbf{x}$. As discussed in Section 2.1, this can be implemented with $\lceil n_m \rceil$ successive communication rounds, each involving the use of A-CSS and one local gradient computation. Each such operation adds therefore $N_{\nabla f} := C_A \lceil n_m \rceil$ to the total communication complexity and $K_{\nabla f} := \lceil n_m \rceil$ to the total local complexity of an algorithm.

Another commonly used operation is to compute several mini-batch gradients ∇f_S at $b \geq 1$ points where $S \in \binom{[n]}{m}$ is sampled uniformly at random. This requires one communication round using R-CSS. The communication complexity of this operation is $N_{\nabla f_S, b} := C_R$ and the local complexity is $K_{\nabla f_S, b} := b$.

In what follows, we omit the subscript $\mathcal{F}$ in the notation for the complexities $N_{\mathcal{F}}(\varepsilon)$ and $K_{\mathcal{F}}(\varepsilon)$; the corresponding problem class is specified in Table 1 for each method.

Centralized GD. The method iterates:

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \frac{1}{L_f} \nabla f(\mathbf{x}^t) \; .$$

The iteration complexity of GD is $T = \mathcal{O}(\frac{L_f F^0}{\varepsilon^2})$ [47], implying that the communication complexity is $N(\varepsilon) = T N_{\nabla f} = \mathcal{O}(C_A n_m \frac{L_f F^0}{\varepsilon^2})$ and the local complexity is $K(\varepsilon) = T K_{\nabla f} = \mathcal{O}(n_m \frac{L_f F^0}{\varepsilon^2})$.

FEDRED [21]. We consider FEDRED-GD, which initializes $\tilde{\mathbf{x}}_0 = \mathbf{x}_0$ and iterates:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x} \in \mathbb{R}^d} \Big\{ f_1(\mathbf{x}_t) + \langle \nabla f_1(\mathbf{x}_t) + \nabla f(\tilde{\mathbf{x}}_t) - \nabla f_1(\tilde{\mathbf{x}}_t), \mathbf{x} \rangle + \frac{\eta}{2} \left\| \mathbf{x} - \mathbf{x}_t \right\|^2 + \frac{\lambda}{2} \left\| \mathbf{x} - \tilde{\mathbf{x}}_t \right\|^2 \Big\} \; ,$$

where $\tilde{\mathbf{x}}_{t+1} = \mathbf{x}_{t+1}$ w.p. p and $\tilde{\mathbf{x}}_{t+1} = \tilde{\mathbf{x}}_t$ w.p. $1-p$. The solution of the subproblem can be computed in a closed-form. For $p \simeq \frac{\Delta_1}{L_1 + \Delta_1}$, $\eta \simeq L_1$ and $\lambda \simeq \Delta_1$, the iteration complexity of the method is $T = \mathcal{O}(\frac{L_1 F^0}{\varepsilon^2})$ [21]. In expectation, once every $1/p$ iterations, the server computes the full gradient $\nabla f(\tilde{\mathbf{x}}_t) = \nabla f(\mathbf{x}_t)$, which adds $N_{\nabla f}$ to the total communication complexity and $K_{\nabla f}$ to the total local complexity. Then the server makes another communication round with D-CSS, sends $\nabla f(\mathbf{x}_t)$ to client 1 which then performs $1/p$ local steps in expectation and sends the result back to the server. The expected number of times the full gradient $\nabla f(\tilde{\mathbf{x}}_t)$ is computed is $pT = \mathcal{O}(\frac{\Delta_1 F^0}{\varepsilon^2})$. The expected number of communication rounds where D-CSS is used is $\mathbb{E}[N_D] = pT$. Therefore, the communication complexity is

$$N(\varepsilon) = \mathbb{E}[C_A N_A + N_D] = \mathcal{O}(pT N_{\nabla f} + pT) = \mathcal{O}\Big(C_A n_m \frac{\Delta_1 F^0}{\varepsilon^2}\Big) \; .$$

The local complexity is bounded by

$$K(\varepsilon) = \mathcal{O}(pT K_{\nabla f} + T) = \mathcal{O}(pT n_m + T) = \mathcal{O}\Big(n_m \frac{\Delta_1 F^0}{\varepsilon^2} + \frac{L_1 F^0}{\varepsilon^2}\Big) \; .$$

FEDAVG [1]. At each communication round $r \geq 0$, the server uses R-CSS to select clients, sends $\mathbf{x}^r$ to each client $i \in S_r$, which then returns an approximation solution $\mathbf{x}_i^{r+1} \approx \arg\min_{\mathbf{x}} f_i(\mathbf{x})$ by running local GD starting at $\mathbf{x}^r$ for K_r steps. Then the next iterate is defined as $\mathbf{x}^{r+1} = \frac{1}{m} \sum_{i \in S_r} \mathbf{x}_i^{r+1}$. When using local-GD, the optimal number of local steps is of order 1 [20]. Therefore, the local complexity is of the same order as the iteration complexity $T = \mathcal{O}(\frac{\zeta_m^2 F^0}{\varepsilon^4} + \frac{\sqrt{L_{\max}} \zeta}{\varepsilon^3} + \frac{L_{\max} F^0}{\varepsilon^2})$ [20], and the communication complexity is $C_R T$.

MIMEMVR [15]. At each iteration $t \geq 0$, the server first uses R-CSS to get a random client set S_t and computes the mini-batch gradient $\nabla f_{S_t}(\mathbf{x}^t)$. Then the server uses A-CSS to establish communication with the same set of clients S_t, sends $\nabla f_{S_t}(\mathbf{x}^t)$ to them which then updates:

$$\mathbf{x}_i^{t+1} \approx \arg\min_{\mathbf{x} \in \mathbb{R}^d} \big\{ f_i(\mathbf{x}) + \big\langle \nabla f_{S_t}(\mathbf{x}^t) - \nabla f_i(\mathbf{x}^t), \mathbf{x} \big\rangle \big\}$$

by running momentum-based first-order methods locally for $\Theta(\frac{L_{\max}}{\Delta_{\max}})$ steps. The next iterate is defined as: $\mathbf{x}^{t+1} = \frac{1}{m} \sum_{i \in S_t} \mathbf{x}_i^{t+1}$. The communication complexity is thus $N(\varepsilon) = \mathbb{E}[C_A N_A + C_R N_R] = (C_A + C_R) T$, where $T = \mathcal{O}(\frac{\zeta_m^2 F^0}{\varepsilon^2} + \frac{\zeta_m \Delta_{\max} F^0}{\varepsilon^3} + \frac{\Delta_{\max} F^0}{\varepsilon^2})$ is the iteration complexity [15]. The local complexity is $\mathcal{O}(\frac{L_{\max}}{\Delta_{\max}} T)$.

CE-LGD [6]. The method initializes $\mathbf{x}^{-1} = \mathbf{x}^0$ and $\mathbf{v}^{-1} \in \mathbb{R}^d$. At each iteration $t \geq 0$, the server first uses R-CSS to select clients S_t, sends $\mathbf{x}^{t-1}$ and $\mathbf{x}^t$ to the client $i \in S_t$ which then computes $\nabla f_i(\mathbf{x}^t)$ and $\nabla f_i(\mathbf{x}^{t-1})$ and sends them back to the server. The server computes $\mathbf{v}^t = \nabla f_{S_t}(\mathbf{x}^t) + (1-\rho)(\mathbf{v}^{t-1} - \nabla f_{S_t}(\mathbf{x}^{t-1}))$ where $\rho \in [0,1]$. Then the server uses R-CSS again to communicate with a random client. The client returns

$\mathbf{x}^{t+1}$ by running the local SARAH method using $\mathbf{v}^t$ for $\Theta(\frac{L_{\max}}{\Delta_{\max}})$ local steps. The iteration complexity is $T = \mathcal{O}(\frac{\zeta_m^2 F^0}{\varepsilon^2} + \frac{\zeta_m \Delta_{\max} F^0}{\sqrt{m}\varepsilon^3} + \frac{\Delta_{\max} F^0}{\varepsilon^2})$ [6]. The communication complexity is $N(\varepsilon) = \mathcal{O}(C_R T)$ and the local complexity is $K(\varepsilon) = \mathcal{O}(\frac{L_{\max}}{\Delta_{\max}} T)$.

SCAFFOLD [20]. At the beginning, each client $i = 1, \ldots, n$ computes $\mathbf{b}_i^0 = \nabla f_i(\mathbf{x}^0)$ and sends the result to the server; the server then computes $\mathbf{b}^0 = \nabla f(\mathbf{x}^0)$, which adds $N_{\nabla f}$ and $K_{\nabla f}$ to the total communication and local complexities, respectively. At each iteration $t \geq 0$, the server uses R-CSS to generate the client set S_t and sends $\mathbf{x}^t$ to each client $i \in S_t$, which then computes $\mathbf{b}_i^t = \nabla f_i(\mathbf{x}^t)$ and sends $\mathbf{b}_i^t - \mathbf{b}_i^{t-1}$ back to the server. The server then updates $\mathbf{b}^t$ (SAG [16]) according to (10) (for $t \geq 1$). Then the server uses A-CSS to contact the clients in S_t again and sends $\mathbf{b}^t$ to them. Each client $i \in S_t$ computes

$$\mathbf{x}_i^{t+1} \approx \arg\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ f_i(\mathbf{x}) + \left\langle \mathbf{b}^t - \nabla f_i(\mathbf{x}^t), \mathbf{x} \right\rangle \right\}$$

by running local GD for $K \simeq 1$ steps and sends the result back to the server. The server computes the next iterate as $\mathbf{x}^{t+1} = \frac{1}{m} \sum_{i \in S_t} \mathbf{x}_i^{t+1}$. The iteration complexity is $T = \mathcal{O}((\frac{n}{m})^{\frac{2}{3}} \frac{L_{\max} F^0}{\varepsilon^2})$ [20]. The communication complexity $N(\varepsilon) = \mathcal{O}(C_A \lceil n_m \rceil + (C_A + C_R)T)$. The local complexity $K(\varepsilon) = \lceil n_m \rceil + (1 + K)T = \mathcal{O}(n_m + T)$.

The final three algorithms do not strictly satisfy our definition of an algorithm because they do not clearly specify when to terminate the local method. However, we still present the conceptual methods and their communication complexity estimates, assuming (rather informally) that certain "local" operations can be implemented by running a certain local method for a sufficiently long time.

FEDDYN [22]. During initialization, the server needs to collect $\mathbf{x}_i^0$ that satisfies $\nabla f_i(\mathbf{x}_i^0) = 0$ from all clients. Therefore, the communication complexity of this operation is $C_A \lceil n_m \rceil$. At each communication round $r \geq 0$, the server uses R-CSS to select clients S_r and sends $\mathbf{x}^r$ to each client $i \in S_r$ which then sends

$$\mathbf{x}_i^{r+1} = \arg\min_{\mathbf{x}} \left\{ f_i(\mathbf{x}) - \langle \nabla f_i(\mathbf{x}_{i,r}), \mathbf{x} \rangle + \frac{\lambda}{2} \|\mathbf{x} - \mathbf{x}^r\|^2 \right\}, \quad i \in S_r \ ,$$

back to the server. For $i \notin S_r$, $\mathbf{x}_i^{r+1} = \mathbf{x}_i^r$. Then the next iterate is updated as:

$$\mathbf{x}^{r+1} = \frac{1}{m} \sum_{i \in S_r} \mathbf{x}_i^{r+1} - \frac{1}{\lambda} \mathbf{h}^{r+1}, \quad \mathbf{h}^{r+1} = \mathbf{h}^r - \lambda \frac{1}{n} \left(\sum_{i \in S_r} \mathbf{x}_i^{r+1} - \mathbf{x}^r \right) .$$

The iteration complexity is $T = \mathcal{O}(n_m \frac{L_{\max} F^0}{\varepsilon^2})$ [22] and the communication complexity is $N(\varepsilon) = \mathcal{O}(C_A n_m + C_R T)$.

SABER-FULL [12]. The method initializes $\mathbf{x}^{-1} = \mathbf{x}^0$ and $\mathbf{v}^{-1} = \mathbf{v}^0 = \nabla f(\mathbf{x}^0)$. At each iteration $t \geq 0$, w.p. $\frac{1}{n_m}$, the server updates $\mathbf{v}^t = \nabla f(\mathbf{x}^t)$ which adds $N_{\nabla f}$ and $K_{\nabla f}$ to the total communication and local complexity, respectively. With probability $1 - \frac{1}{n_m}$, the server uses R-CSS, obtains the random set S_t, and computes two mini-batch gradients $\nabla f_{S_t}(\mathbf{x}^t)$ and $\nabla f_{S_t}(\mathbf{x}^{t-1})$. This operation adds $N_{\nabla f_{S_t}}^2$ and $K_{\nabla f_{S_t}}^2$ to the communication and local complexity, respectively. Then the server updates $\mathbf{v}^t = \mathbf{v}^{t-1} + \nabla f_{S_t}(\mathbf{x}^t) - \nabla f_{S_t}(\mathbf{x}^{t-1})$. (The original method samples a single index m_t. Here we extend it to S_t.) After the computation of $\mathbf{v}^t$, the server uses R-CSS, samples a random index $\tilde{m}_t \in [n]$, and sends $\mathbf{v}^t$ and $\mathbf{x}^t$ to the client $\tilde{m}_t$, which then returns

$$\mathbf{x}^{t+1} \approx \arg\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ f_{\tilde{m}_t}(\mathbf{x}) + \left\langle \mathbf{v}^t - \nabla f_{\tilde{m}_t}(\mathbf{x}^t), \mathbf{x} \right\rangle + \frac{\lambda}{2} \left\| \mathbf{x} - \mathbf{x}^t \right\|^2 \right\}$$

back to the server. The iteration complexity of the method is $T = \mathcal{O}(\frac{\sqrt{n_m} \Delta_{\max} F^0}{\varepsilon^2})$ if each subproblem is solved exactly. The total communication complexity is

$$N(\varepsilon) = \mathbb{E}[C_A N_A + C_R N_R] = \mathcal{O}\Big(C_A n_m + \frac{1}{n_m} C_A T n_m + (1 - \frac{1}{n_m}) C_R T + C_R T \Big) = \mathcal{O}(C_A n_m + C_A T) \ .$$

SABER-PARTIAL [12]. We refer to Algorithm 2 in the original paper as SABER-PARTIAL. By Theorem 3 in that paper, the best p is 1. The algorithm initializes $\mathbf{v}^0 = \nabla f(\mathbf{x}^0)$, which adds $N_{\nabla f}$ and $K_{\nabla f}$ to the total communication and local complexities, respectively. At each iteration $t \geq 1$, the method also needs to compute a mini-batch gradient $\mathbf{v}^t = \frac{1}{s} \sum_{i \in S_t'} \nabla f_i(\mathbf{x}^t)$ where S_t' is sampled uniformly at random with replacement and $|S_t'| = s$ where s is a parameter of the method. Since we assume that the server can communicate with at most m clients at each round, implementing this operation requires $\lceil s/m \rceil$ sequential communication rounds with R-CSS. After that, the method needs to choose another random set S_t with $|S_t| = s$. This adds $C_R \lceil s/m \rceil$ to the total communication complexity. The server sends $\mathbf{x}^t$ and $\mathbf{v}^t$ to the client $i \in S_t$, which then computes

$$\mathbf{x}_i^{t+1} \approx \arg\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ f_i(\mathbf{x}) + \left\langle \mathbf{v}^t - \nabla f_i(\mathbf{x}^t), \mathbf{x} \right\rangle + \frac{\lambda}{2} \left\| \mathbf{x} - \mathbf{x}^t \right\|^2 \right\} ,$$

and sends the result back to the server. The next iterate is updated as $\mathbf{x}^{t+1} = \frac{1}{s}\sum_{i\in S_t}\mathbf{x}_i^{t+1}$. If $\frac{\zeta^2}{\varepsilon^2} \lesssim n$ and s is chosen as $\Theta(\frac{\zeta^2}{\varepsilon^2})$, then the method can output an ε-approximate stationary point after $T = \mathcal{O}(\frac{\Delta_{\max}F^0}{\sqrt{p}\varepsilon^2})$ iterations if each subproblem is solved exactly. The communication complexity is $N(\varepsilon) = \mathbb{E}[C_R N_R + C_A N_A] = \mathcal{O}(C_R T\lceil\frac{\zeta^2}{m\varepsilon^2}\rceil + C_A n_m)$.

Discussions. FEDDYN, SABER-FULL and SABER-PARTIAL do not strictly satisfy our definition of an algorithm since they do not precisely specify when to terminate the local methods. The problem class for which FEDAVG, MIMEMVR, CE-LGD, and SABER-PARTIAL are analyzed is the smallest among all methods. Specifically, in addition to IS, they also assume BGD, which can be restrictive and exclude simple quadratics. Among these four, MIMEMVR and CE-LGD improve upon FEDAVG and SABER-PARTIAL in terms of their dependence on the target accuracy ε. Except for FEDAVG, the remaining three methods replace the dependence on $L_{\max}$ with $\Delta_{\max}$ in the communication complexity. Furthermore, compared to MIMEMVR, CE-LGD achieves a better dependence on m in the term involving ε^{-3}.

For the remaining methods, SCAFFOLD improves the dependence on n from n_m (as in FEDDYN) to $n_m^{2/3}$. SAVER-FULL further reduces the dependence on n to $\sqrt{n_m}$ and simultaneously improves the smoothness dependence from $L_{\max}$ to $\Delta_{\max}$. I-CGM-RG-SVRG achieves a tighter bound of $C_R\Delta_1 + \sqrt{C_A C_R n_m}\delta_m \lesssim C_A\sqrt{n_m}\Delta_{\max}$. Finally, I-CGM-RG-SAGA improves the communication cost constant from C_A to C_R, compared to I-CGM-RG-SVRG, while maintaining the same local complexity—the best among all existing methods.

Remark A.5. According to [26], one may alternatively compute the full gradient using only R-CSS. Let $m = 1$ for simplicity. Lemma 2 in [26] shows that w.p. $1-\delta$, we can recover the full gradient $\nabla f(\mathbf{x})$ at a given point $\mathbf{x}$ by making $2n^2\log(\frac{2n}{\delta})$ communication rounds with R-CSS. This can be helpful when the cost C_A is extremely large. Indeed, the current communication complexity of I-CGM-RG-SAGA is of order $C_A n + C_R((\Delta_1 + \sqrt{n}\delta)F^0/\varepsilon^2)$. As soon as $C_A \gtrsim C_R((\Delta_1 + \sqrt{n}\delta)F^0/\varepsilon^2)/n$, the complexity is dominated by $C_A n$. This term arises from $2n$ sequential communication rounds with A-CSS for computing the full gradients. Now if we replace these operations with $4n^2\log(\frac{2n}{\delta})$ sequential rounds with R-CSS, the total complexity might be reduced to $C_R\big(n^2\log(n) + (\Delta_1 + \sqrt{n}\delta)F^0/\varepsilon^2\big)$. We leave a full theoretical development of this direction as interesting future work.

B Technical Preliminaries

We frequently use the following lemmas for the proofs.

Lemma B.1. *For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ and any $\gamma > 0$, we have:*

$$|\langle \mathbf{x}, \mathbf{y}\rangle| \le \frac{\gamma}{2}\|\mathbf{x}\|^2 + \frac{1}{2\gamma}\|\mathbf{y}\|^2, \tag{B.1}$$

$$\|\mathbf{x}+\mathbf{y}\|^2 \le (1+\gamma)\|\mathbf{x}\|^2 + \Big(1+\frac{1}{\gamma}\Big)\|\mathbf{y}\|^2 . \tag{B.2}$$

Lemma B.2 ([27], Lemma 13)**.** *Let $\{\mathbf{g}_i\}_{i=1}^n$ be vectors in $\mathbb{R}^d$ with $n \ge 2$. Let $m \in [n]$ and let $S \in \binom{[n]}{m}$ be sampled uniformly at random without replacement. Let $\bar{\mathbf{g}} := \frac{1}{n}\sum_{i=1}^n \mathbf{g}_i$, $\sigma^2 := \frac{1}{n}\sum_{i=1}^n \|\mathbf{g}_i - \bar{\mathbf{g}}\|^2$, and $\bar{\mathbf{g}}_S := \frac{1}{m}\sum_{j\in S}\mathbf{g}_j$. Then,*

$$\mathbb{E}_S[\bar{\mathbf{g}}_S] = \bar{\mathbf{g}} \qquad \textit{and} \qquad \mathbb{E}_S[\|\bar{\mathbf{g}}_S - \bar{\mathbf{g}}\|^2] = \frac{n-m}{n-1}\frac{\sigma^2}{m}. \tag{B.3}$$

Lemma B.3 ([48], Fact 2.3)**.** *Let $A_0, A_1, \ldots$ be reals and let $K \sim \mathrm{Geom}(p)$ with $p \in (0,1]$, that is $\mathbb{P}(K = k) = (1-p)^k p$ for each $k \in \{0,1,2,\ldots\}$. Then it holds that: $\mathbb{E}[K] = \frac{1}{p} - 1$ and*

$$\mathbb{E}[A_K] = (1-p)\,\mathbb{E}[A_{K+1}] + pA_0 . \tag{B.4}$$

Proof. Using the identity $\sum_{k\ge 0} kq^k = \frac{q}{(1-q)^2}$ for any $|q| < 1$, we have:

$$\mathbb{E}[K] = p\sum_{k\ge 0} k(1-p)^k = p\frac{1-p}{p^2} = \frac{1}{p} - 1 .$$

To prove the second part, using the definition of K,

$$\mathbb{E}[A_{K+1}] = p\sum_{k\ge 0} A_{k+1}(1-p)^k = \frac{p}{1-p}\sum_{k\ge 1} A_k(1-p)^k = \frac{1}{1-p}(\mathbb{E}[A_K] - pA_0) .$$

Rearranging gives the claim. □

Lemma B.4. *Let $(A_t)_{t=0}^{\infty}$, $(B_t)_{t=0}^{\infty}$ and be two non-negative sequences such that*

$$A_{i+1} \le (1-\alpha)A_i + B_i$$

for any $i \ge 0$ with $\alpha \in (0,1]$. Then for any $t \ge 1$,

$$A_t \le (1-\alpha)^t A_0 + \sum_{i=1}^{t} (1-\alpha)^{t-i} B_{i-1} \ ,$$

and for any $T \ge 1$,

$$\sum_{t=1}^{T} A_t \le \frac{(1-\alpha)(1-(1-\alpha)^T)}{\alpha} A_0 + \sum_{t=0}^{T-1} \frac{1-(1-\alpha)^{T-t}}{\alpha} B_t \le \frac{1-\alpha}{\alpha} A_0 + \frac{1}{\alpha} \sum_{t=0}^{T-1} B_t \ .$$

Proof. When $\alpha = 1$, the claim clearly holds. Let $0 < \alpha < 1$. Dividing both sides of the main recurrence by $(1-\alpha)^{i+1}$, we have for any $i \ge 0$:

$$\frac{A_{i+1}}{(1-\alpha)^{i+1}} \le \frac{A_i}{(1-\alpha)^i} + \frac{B_i}{(1-\alpha)^{i+1}} \ .$$

Summing up from $i = 0$ to $i = t-1$, we get, for any $t \ge 1$:

$$\frac{A_t}{(1-\alpha)^t} \le A_0 + \sum_{i=0}^{t-1} \frac{B_i}{(1-\alpha)^{i+1}} = A_0 + \sum_{i=1}^{t} \frac{B_{i-1}}{(1-\alpha)^i} \ .$$

This proves the first claim. To prove the second part, we sum up the first claim from $t = 1$ to $t = T$,

$$\begin{aligned}
\sum_{t=1}^{T} A_t &\le \sum_{t=1}^{T} (1-\alpha)^t A_0 + \sum_{t=1}^{T} \sum_{i=1}^{t} (1-\alpha)^{t-i} B_{i-1} \\
&= \frac{(1-\alpha)(1-(1-\alpha)^T)}{\alpha} A_0 + \sum_{i=1}^{T} \sum_{t=i}^{T} (1-\alpha)^{t-i} B_{i-1} \\
&= \frac{(1-\alpha)(1-(1-\alpha)^T)}{\alpha} A_0 + \sum_{i=1}^{T} \frac{1-(1-\alpha)^{T-i+1}}{\alpha} B_{i-1} \\
&= \frac{(1-\alpha)(1-(1-\alpha)^T)}{\alpha} A_0 + \sum_{t=0}^{T-1} \frac{1-(1-\alpha)^{T-t}}{\alpha} B_t \ .
\end{aligned}$$

□

Lemma B.5. *Let $p \in (0,1)$. The minimizer of the problem $\min_{\gamma \in (0,1)} \{ f(\gamma) := \frac{1-\gamma p}{\gamma(1-\gamma)} \}$ is attained at $\gamma^\star = \frac{1-\sqrt{1-p}}{p}$.*

Proof. Differentiate $f(\gamma)$, we have $f'(\gamma) = \frac{-1+2\gamma-p\gamma^2}{\gamma^2(1-\gamma)^2}$. Setting $f'(\gamma) = 0$ with $\gamma \in (0,1)$ gives $\gamma^\star = \frac{1-\sqrt{1-p}}{p}$. Since $f(\gamma) \to \infty$ as $\gamma \to 0^+$ or $\gamma \to 1^-$, the critical point $\gamma^\star$ is the minimizer over $(0,1)$. □

C Proofs for I-CGM

Lemma C.1. *Let I-CGM be applied to Problem* (1) *under Assumption 2.1. Let $\lambda > \Delta_1$. Then for any $t \ge 0$,*

$$\left\| \nabla f(\mathbf{x}^{t+1}) \right\| \le (\lambda + \Delta_1) \hat{\chi}_{t+1} + \hat{\Sigma}_t + e_t \ ,$$

where $\hat{\chi}_t := \|\mathbf{x}^t - \mathbf{x}^{t-1}\|$. For any $\mathbf{x} \in \mathbb{R}^d$, we have:

$$F_t(\mathbf{x}^t) - F_t(\mathbf{x}) \le f(\mathbf{x}^t) - f(\mathbf{x}) + \frac{\hat{\Sigma}_t^2}{2(\lambda - \Delta_1)} \ .$$

Suppose the iterates satisfy (5)*, then the function value decreases as:*

$$f(\mathbf{x}^{t+1}) \le f(\mathbf{x}^t) - \frac{\lambda - \Delta_1}{4} \hat{\chi}_{t+1}^2 + \frac{\hat{\Sigma}_t^2}{\lambda - \Delta_1} \ .$$

Proof. By the definition of F_t, we have:

$$\begin{aligned}
\nabla F_t(\mathbf{x}^{t+1}) &= \nabla f_1(\mathbf{x}^{t+1}) + \mathbf{g}^t - \nabla f_1(\mathbf{x}^t) + \lambda(\mathbf{x}^{t+1} - \mathbf{x}^t) \\
&= \nabla f(\mathbf{x}^{t+1}) + \big(\mathbf{g}^t - \nabla f(\mathbf{x}^t) \big) + \big(\nabla h_1(\mathbf{x}^t) - \nabla h_1(\mathbf{x}^{t+1}) \big) + \lambda(\mathbf{x}^{t+1} - \mathbf{x}^t) \ .
\end{aligned}$$

It follows that:

$$\begin{aligned}\big\|\nabla f(\mathbf{x}^{t+1})\big\| &\le \lambda\hat{\chi}_{t+1} + \|\nabla h_1(\mathbf{x}^t) - \nabla h_1(\mathbf{x}^{t+1})\| + \hat{\Sigma}_t + e_t \\ &\overset{(2)}{\le} (\lambda + \Delta_1)\hat{\chi}_{t+1} + \hat{\Sigma}_t + e_t \ ,\end{aligned}$$

which proves the first inequality. Using the definition of F_t, for any $\mathbf{x} \in \mathbb{R}^d$, we have:

$$\begin{aligned}&F_t(\mathbf{x}^t) - F_t(\mathbf{x}) \\ &= f_1(\mathbf{x}^t) + h_1(\mathbf{x}^t) - f_1(\mathbf{x}) - h_1(\mathbf{x}^t) - \langle \mathbf{g}^t - \nabla f_1(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t\rangle - \frac{\lambda}{2}\|\mathbf{x} - \mathbf{x}^t\|^2 \\ &= f(\mathbf{x}^t) - f(\mathbf{x}) + f(\mathbf{x}) - f_1(\mathbf{x}) - h_1(\mathbf{x}^t) - \langle \mathbf{g}^t - \nabla f_1(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t\rangle - \frac{\lambda}{2}\|\mathbf{x} - \mathbf{x}^t\|^2 \\ &= f(\mathbf{x}^t) - f(\mathbf{x}) + \big(h_1(\mathbf{x}) - h_1(\mathbf{x}^t) - \big\langle \nabla h_1(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t\big\rangle\big) - \frac{\lambda}{2}\|\mathbf{x} - \mathbf{x}^t\|^2 - \big\langle \mathbf{g}^t - \nabla f(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t\big\rangle \\ &\overset{(2.1)}{\le} f(\mathbf{x}^t) - f(\mathbf{x}) - \frac{\lambda - \Delta_1}{2}\|\mathbf{x} - \mathbf{x}^t\|^2 - \big\langle \mathbf{g}^t - \nabla f(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t\big\rangle \ .\end{aligned}$$

Using (B.1), we can bound the last two terms by: $\frac{\|\mathbf{g}^t - \nabla f(\mathbf{x}^t)\|^2}{2(\lambda - \Delta_1)}$, which proves the second claim. Substituting $\mathbf{x} = \mathbf{x}^{t+1}$ and using (5), we get:

$$\begin{aligned}f(\mathbf{x}^{t+1}) &\le f(\mathbf{x}^t) - \frac{\lambda - \Delta_1}{2}\hat{\chi}_{t+1}^2 - \big\langle \mathbf{g}^t - \nabla f(\mathbf{x}^t), \mathbf{x}^{t+1} - \mathbf{x}^t\big\rangle \\ &\overset{(B.1)}{\le} f(\mathbf{x}^t) - \frac{\lambda - \Delta_1}{4}\hat{\chi}_{t+1}^2 + \frac{\hat{\Sigma}_t^2}{\lambda - \Delta_1} \ .\end{aligned}$$

□

C.1 Proof for Theorem 3.1.

Proof. Let $t \ge 0$. By Lemma C.1, we have:

$$\frac{\lambda - \Delta_1}{4}\hat{\chi}_{t+1}^2 \le f(\mathbf{x}^t) - f(\mathbf{x}^{t+1}) + \frac{\hat{\Sigma}_t^2}{\lambda - \Delta_1} \ .$$

Using the first claim of Lemma C.1, we have:

$$\|\nabla f(\mathbf{x}^{t+1})\|^2 \le \big(\lambda + \Delta_1)\hat{\chi}_{t+1} + \hat{\Sigma}_t + e_t\big)^2 \le 2(\lambda + \Delta_1)^2\hat{\chi}_{t+1}^2 + 2(\hat{\Sigma}_t + e_t)^2 \ ,$$

Adding $(\lambda + \Delta_1)^2\hat{\chi}_{t+1}^2$ to both sides of this inequality and substituting the first display, we have:

$$\begin{aligned}&\|\nabla f(\mathbf{x}^{t+1})\|^2 + (\lambda + \Delta_1)^2\hat{\chi}_{t+1}^2 \\ &\le 3(\lambda + \Delta_1)^2\Big(\frac{4}{\lambda - \Delta_1}\big(f(\mathbf{x}^t) - f(\mathbf{x}^{t+1})\big) + \frac{4}{\lambda - \Delta_1}\frac{\hat{\Sigma}_t^2}{\lambda - \Delta_1}\Big) + 2(\hat{\Sigma}_t + e_t)^2 \\ &\le \frac{12(\lambda + \Delta_1)^2}{\lambda - \Delta_1}\big(f(\mathbf{x}^t) - f(\mathbf{x}^{t+1})\big) + \Big(\frac{12(\lambda + \Delta_1)^2}{(\lambda - \Delta_1)^2} + 4\Big)\hat{\Sigma}_t^2 + 4e_t^2 \ .\end{aligned}$$

Summing up from $t = 0$ to $T - 1$, we get the claim. □

C.2 Proofs of Local CGM for Solving the Subproblems

Lemma C.2 (Composite gradient method)**.** *Consider the composite problem:*

$$\min_{\mathbf{x}\in\mathbb{R}^d}\big\{F(\mathbf{x}) := \phi(\mathbf{x}) + \psi(\mathbf{x})\big\} \ ,$$

where ϕ is L_ϕ-smooth and ψ is λ_ψ-strongly convex and simple with $\lambda_\psi \ge 0$. Let $\eta = L_\phi$. Consider the composite gradient method:

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}\in\mathbb{R}^d}\big\{L_k(\mathbf{x}) := \phi(\mathbf{x}_k) + \langle\nabla\phi(\mathbf{x}_k), \mathbf{x} - \mathbf{x}_k\rangle + \frac{\eta}{2}\|\mathbf{x} - \mathbf{x}_k\|^2 + \psi(\mathbf{x})\big\} \ .$$

Then for any $k \ge 0$, $F(\mathbf{x}_{k+1}) \le F(\mathbf{x}_k)$. For any $K \ge 1$, it holds that:

$$\|\nabla F(\mathbf{x}_K^\star)\|^2 \le \frac{8L_\phi^2\big[F(\mathbf{x}_0) - F(\mathbf{x}_K)\big]}{(L_\phi + \lambda_\phi)K} \ ,$$

where $\mathbf{x}_K^\star = \arg\min_{(\mathbf{x}_k)_{k=1}^K}\|\nabla F(\mathbf{x}_k)\|$. Furthermore, if $\hat{K} \sim \mathrm{Geom}(p)$ with $p \in (0, 1]$, then we also have:

$$\mathbb{E}_{\hat{K}}\big[\|\nabla F(\mathbf{x}_{\hat{K}+1})\|^2\big] \le \frac{8L_\phi^2 p}{L_\phi + \lambda_\psi}\big[F(\mathbf{x}_0) - \mathbb{E}_{\hat{K}}[F(\mathbf{x}_{\hat{K}+1})]\big] \ .$$

Proof. Let $k \geq 0$. By $(L_\phi + \lambda_\psi)$-strong convexity of L_k, for any $\mathbf{x} \in \mathbb{R}^d$, we have,

$$L_k(\mathbf{x}) \geq L_k(\mathbf{x}_{k+1}) + \frac{L_\phi + \lambda_\psi}{2} \|\mathbf{x} - \mathbf{x}_{k+1}\|^2 \ .$$

Substituting $\mathbf{x} = \mathbf{x}_k$, it follows that,

$$\begin{aligned} F(\mathbf{x}_k) &\geq \phi(\mathbf{x}_k) + \langle \nabla\phi(\mathbf{x}_k), \mathbf{x}_{k+1} - \mathbf{x}_k \rangle + \frac{L_\phi}{2} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^2 + \psi(\mathbf{x}_{k+1}) + \frac{L_\phi + \lambda_\psi}{2} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^2 \\ &\geq \phi(\mathbf{x}_{k+1}) + \psi(\mathbf{x}_{k+1}) + \frac{L_\phi + \lambda_\psi}{2} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^2 = F(\mathbf{x}_{k+1}) + \frac{L_\phi + \lambda_\psi}{2} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^2 \ . \end{aligned}$$

This proves that the function value of F monotinically decreases. By the definition of $\mathbf{x}_{k+1}$, we get:

$$\nabla\phi(\mathbf{x}_k) + L_\phi(\mathbf{x}_{k+1} - \mathbf{x}_k) + \nabla\psi(\mathbf{x}_{k+1}) = 0 \ .$$

It follows that:

$$\nabla F(\mathbf{x}_{k+1}) = \nabla\phi(\mathbf{x}_{k+1}) + \nabla\psi(\mathbf{x}_{k+1}) = \nabla\phi(\mathbf{x}_{k+1}) - \nabla\phi(\mathbf{x}_k) + L_\phi(\mathbf{x}_k - \mathbf{x}_{k+1}) \ ,$$

and hence,

$$\|\nabla F(\mathbf{x}_{k+1})\| \leq \|\nabla\phi(\mathbf{x}_{k+1}) - \nabla\phi(\mathbf{x}_k)\| + \eta \|\mathbf{x}_{k+1} - \mathbf{x}_k\| \leq 2L_\phi \|\mathbf{x}_{k+1} - \mathbf{x}_k\| \ .$$

Substituting this inequality into the second display, we get, for any $k \geq 0$:

$$\|\nabla F(\mathbf{x}_{k+1})\|^2 \leq \frac{8L_\phi^2}{L_\phi + \lambda_\psi} \big[F(\mathbf{x}_k) - F(\mathbf{x}_{k+1})\big] \ .$$

Summing up from $k = 0$ to $K - 1$, we have:

$$\sum_{k=1}^{K} \|\nabla F(\mathbf{x}_k)\|^2 \leq \frac{8L_\phi^2}{L_\phi + \lambda_\phi} \big[F(\mathbf{x}_0) - F(\mathbf{x}_K)\big] \ .$$

Dividing both sides by K, we get the first claim.

For the second claim, substituting $k = \hat{K}$ with $\hat{K} \sim \mathrm{Geom}(p)$ into the last second display, passing to the expectations and applying Lemma B.3, we have:

$$\begin{aligned} \mathbb{E}_{\hat{K}}[\|\nabla F(\mathbf{x}_{\hat{K}+1})\|^2] &\leq \frac{8L_\phi^2}{L_\phi + \lambda_\psi} \mathbb{E}_{\hat{K}}[F(\mathbf{x}_{\hat{K}}) - F(\mathbf{x}_{\hat{K}+1})] \\ &\leq \frac{8L_\phi^2}{L_\phi + \lambda_\psi} \big((1-p)\, \mathbb{E}_{\hat{K}}[F(\mathbf{x}_{\hat{K}+1})] + pF(\mathbf{x}_0) - \mathbb{E}_{\hat{K}}[F(\mathbf{x}_{\hat{K}+1})]\big) \\ &= \frac{8L_\phi^2 p}{L_\phi + \lambda_\psi} \big[F(\mathbf{x}_0) - \mathbb{E}_{\hat{K}}[F(\mathbf{x}_{\hat{K}+1})]\big] \ . \end{aligned}$$

□

C.2.1 Proof of Lemma 3.3

Proof. Applying Lemma C.2 (with $\phi(\mathbf{x}) = f_1(\mathbf{x})$ and $\psi(\mathbf{x}) = \langle \mathbf{g}^t - \nabla f_1(\mathbf{x}^t), \mathbf{x} - \mathbf{x}^t \rangle + \frac{\lambda}{2} \|\mathbf{x} - \mathbf{x}^t\|^2$), we have for any $t \geq 0$:

$$e_t^2 = \big\|\nabla F_t(\mathbf{x}^{t+1})\big\|^2 \leq \frac{8L_1^2(F_t(\mathbf{x}^t) - F_t^\star)}{(L_1 + \lambda)K} \ ,$$

where $F_t^\star := \min_{\mathbf{x} \in \mathbb{R}^d}\{F_t(\mathbf{x})\}$. Applying Lemma C.1, we get:

$$F_t(\mathbf{x}^t) - F_t^\star \leq f(\mathbf{x}^t) - f^\star + \frac{\hat{\Sigma}_t^2}{2(\lambda - \Delta_1)} \ .$$

It follows that:

$$\sum_{t=0}^{T-1} e_t^2 \leq \frac{8L_1^2}{(L_1 + \lambda)K} \Big(\sum_{t=0}^{T-1} (f(\mathbf{x}^t) - f^\star) + \sum_{t=0}^{T-1} \frac{\hat{\Sigma}_t^2}{2(\lambda - \Delta_1)}\Big) \ .$$

We next upper bound $\sum_{t=0}^{T-1}(f(\mathbf{x}^t) - f^\star)$. Applying Lemma C.2, we have for any $i \geq 0$:

$$f(\mathbf{x}^{i+1}) \leq f(\mathbf{x}^i) + \frac{\hat{\Sigma}_i^2}{2(\lambda - \Delta_1)} \ .$$

Summing up from $i = 0$ to $i = t - 1$, we have:

$$f(\mathbf{x}^t) \leq f(\mathbf{x}^0) + \sum_{i=0}^{t-1} \frac{\hat{\Sigma}_i^2}{2(\lambda - \Delta_1)} \ .$$

Hence,

$$\sum_{t=0}^{T-1}(f(\mathbf{x}^t)-f^\star)\leq T(f(\mathbf{x}^0)-f^\star)+\sum_{t=0}^{T-1}\sum_{i=0}^{t-1}\frac{\hat{\Sigma}_i^2}{2(\lambda-\Delta_1)}\leq TF^0+T\sum_{t=0}^{T-2}\frac{\hat{\Sigma}_t^2}{2(\lambda-\Delta_1)}\ .$$

It follows that:

$$\sum_{t=0}^{T-1}e_t^2\leq\frac{8L_1^2}{(L_1+\lambda)K}\Big(TF^0+(T+1)\sum_{t=0}^{T-1}\frac{\hat{\Sigma}_t^2}{2(\lambda-\Delta_1)}\Big)\ .$$

To achieve the accuracy condition 6, by the choice of K, we have

$$\frac{8L_1^2T}{(L_1+\lambda)K}\leq\frac{8L_1T}{K}\leq\lambda-\Delta_1\leq\frac{(\lambda+\Delta_1)^2}{\lambda-\Delta_1}\ ,\quad\text{and}\quad\frac{8L_1^2}{(L_1+\lambda)K}\frac{(T+1)}{2(\lambda-\Delta_1)}\leq\frac{8L_1T}{(\lambda-\Delta_1)K}\leq 1\ .$$

Passing to the full expectation, we get the claim. □

C.2.2 Proof of Lemma 3.4

Proof. Applying Lemma C.2 and C.1, we have for any $t\geq 0$:

$$\mathbb{E}_{\hat{K}_t}[e_t^2]\leq\frac{8L_1^2p}{L_1+\lambda}\,\mathbb{E}_{\hat{K}_t}[F_t(\mathbf{x}^t)-F_t(\mathbf{x}^{t+1})]\leq\frac{8L_1^2p}{L_1+\lambda}\Big(\mathbb{E}_{\hat{K}_t}\big[f(\mathbf{x}^t)-f(\mathbf{x}^{t+1})\big]+\frac{\hat{\Sigma}_t^2}{2(\lambda-\Delta_1)}\Big)\ .$$

Taking the full expectation and summing up from $t=0$ to $t=T-1$, we have:

$$\sum_{t=0}^{T-1}\mathbb{E}[e_t^2]\leq\frac{8L_1^2p}{L_1+\lambda}\Big(f(\mathbf{x}^0)-f^\star+\frac{1}{2(\lambda-\Delta_1)}\sum_{t=0}^{T-1}\Sigma_t^2\Big)\ .$$

By the choice of p, it holds that:

$$\frac{8L_1^2p}{L_1+\lambda}=\frac{L_1^2(\lambda-\Delta_1)}{(L_1+\lambda)^2}\leq\lambda-\Delta_1\leq\frac{(\lambda+\Delta_1)^2}{\lambda-\Delta_1}\ ,$$

and

$$\frac{4L_1^2p}{(L_1+\lambda)(\lambda-\Delta_1)}=\frac{L_1^2}{2(L_1+\lambda)^2}<1\ .$$

Hence, condition (6) is satisfied. □

C.3 Proofs of Properties of the SAGA and SVRG Estimators

Lemma C.3. *Consider the SAGA estimator. Then for any $t\geq 2$, it holds that:*

$$\mathbf{b}^t=\mathbf{b}^{t-1}+\frac{1}{n_m}[\nabla f_{S_t}(\mathbf{x}^t)-\mathbf{b}_{S_t}^{t-1}]\ .$$

Proof. Indeed,

$$\begin{aligned}\mathbf{b}^t&=\frac{1}{n}\sum_{i=1}^n\mathbf{b}_i^t=\frac{1}{n}\Big[\sum_{i\notin S_t}\mathbf{b}_i^{t-1}+\sum_{i\in S_t}\nabla f_i(\mathbf{x}^t)\Big]=\frac{1}{n}\Big[\sum_{i=1}^n\mathbf{b}_i^{t-1}+\sum_{i\in S_t}[\nabla f_i(\mathbf{x}^t)-\mathbf{b}_i^{t-1}]\Big]\\&=\mathbf{b}^{t-1}+\frac{1}{n_m}[\nabla f_{S_t}(\mathbf{x}^t)-\mathbf{b}_{S_t}^{t-1}]\ .\end{aligned}$$

□

C.3.1 Proof of Lemma 4.1

Proof. Let $t\geq 2$. By definition, $\mathbf{G}^t=\frac{1}{m}\sum_{i\in S_t}\mathbf{G}_i^t$, where $\mathbf{G}_i^t:=\nabla f_i(\mathbf{x}^t)-\mathbf{b}_i^{t-1}+\mathbf{b}^{t-1}$ and S_t is independent from $\mathbf{x}^t$ and $(\mathbf{b}_i^{t-1})_{i=1}^n$. Therefore, according to Lemma B.2, we have:

$$\mathbb{E}_{S_t}[\mathbf{G}^t]=\frac{1}{n}\sum_{i=1}^n\mathbf{G}_i^t=\nabla f(\mathbf{x}^t)\quad\text{and}\quad\mathbb{E}_{S_t}[\|\mathbf{G}^t-\nabla f(\mathbf{x}^t)\|^2]=\frac{n-m}{n-1}\frac{1}{m}\hat{\sigma}_{t,1}^2\ ,$$

where $\hat{\sigma}_{t,1}^2:=\frac{1}{n}\sum_{i=1}^n\|(\nabla f_i(\mathbf{x}^t)-\mathbf{b}_i^{t-1})-(\nabla f(\mathbf{x}^t)-\mathbf{b}^{t-1})\|^2$. Taking the expectation w.r.t. $S_{[t-1]}$ on both sides, we get:

$$\sigma_t^2=\frac{n-m}{n-1}\frac{1}{m}\,\mathbb{E}_{S_{[t-1]}}[\hat{\sigma}_{t,1}^2]:=\frac{q_m}{m}\sigma_{t,1}^2\ .$$

We next derive the recurrence for $\sigma_{t,1}^2$. Denote $\hat{\chi}_t := \|\mathbf{x}^t - \mathbf{x}^{t-1}\|$. For any $\alpha > 0$, we obtain:

$$\hat{\sigma}_{t+1,1}^2 = \frac{1}{n}\sum_{i=1}^n \|(\nabla f_i(\mathbf{x}^{t+1}) - \mathbf{b}_i^t) - (\nabla f(\mathbf{x}^{t+1}) - \mathbf{b}^t)\|^2$$

$$= \frac{1}{n}\sum_{i=1}^n \big\|(\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^t) - (\nabla f(\mathbf{x}^t) - \mathbf{b}^t) + [\nabla h_i(\mathbf{x}^t) - \nabla h_i(\mathbf{x}^{t+1})]\big\|^2$$

$$\overset{\text{(B.2),(3)}}{\leq} (1+\alpha)\frac{1}{n}\sum_{i=1}^n \|(\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^t) - (\nabla f(\mathbf{x}^t) - \mathbf{b}^t)\|^2 + \Big(1 + \frac{1}{\alpha}\Big)\delta^2\hat{\chi}_{t+1}^2$$

$$= (1+\alpha)\Big[\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^t - \nabla f(\mathbf{x}^t)\|^2 - \|\mathbf{b}^t\|^2\Big] + \Big(1 + \frac{1}{\alpha}\Big)\delta^2\hat{\chi}_{t+1}^2$$

$$= (1+\alpha)\Big[\frac{1}{n_m}\|\nabla f(\mathbf{x}^t)\|^2 + \frac{1}{n}\sum_{i\notin S_t} \|\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^{t-1} - \nabla f(\mathbf{x}^t)\|^2 - \|\mathbf{b}^t\|^2\Big] + \Big(1 + \frac{1}{\alpha}\Big)\delta^2\hat{\chi}_{t+1}^2 \,,$$

where the last second equality follows from the identity $\frac{1}{n}\sum_{i=1}^n[\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^t - \nabla f(\mathbf{x}^t)] = \mathbf{b}^t$, and the last equality follows from the definition of $\mathbf{b}_i^t$. Further note that

$$\mathbb{E}_{S_t}\Big[\frac{1}{n}\sum_{i\notin S_t} \|\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^{t-1} - \nabla f(\mathbf{x}^t)\|^2\Big] = \frac{1}{n}\sum_{i=1}^n \mathbb{P}(i \notin S_t)\|\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^{t-1} - \nabla f(\mathbf{x}^t)\|^2$$

$$= \Big(1 - \frac{1}{n_m}\Big)\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}^t) - \mathbf{b}_i^{t-1} - \nabla f(\mathbf{x}^t)\|^2 = \Big(1 - \frac{1}{n_m}\Big)[\hat{\sigma}_{t,1}^2 + \|\mathbf{b}^{t-1}\|^2]$$

Taking the expectation w.r.t. S_t on both sides of the last second display and plugging in this identity, we obtain:

$$\mathbb{E}_{S_t}[\hat{\sigma}_{t+1,1}^2 + (1+\alpha)\|\mathbf{b}^t\|^2] \leq (1+\alpha)\Big(1 - \frac{1}{n_m}\Big)\big[\hat{\sigma}_{t,1}^2 + \|\mathbf{b}^{t-1}\|^2\big] + (1+\alpha)\frac{1}{n_m}\|\nabla f(\mathbf{x}^t)\|^2 + \Big(1 + \frac{1}{\alpha}\Big)\delta^2\,\mathbb{E}_{S_t}[\hat{\chi}_{t+1}^2] \,.$$

Taking the expectation w.r.t. $S_{[t-1]}$ on both sides and denoting $\mathbf{B}_t^2 := (1+\alpha)\,\mathbb{E}_{S_{[t]}}[\|\mathbf{b}^t\|^2]$, we get:

$$\sigma_{t+1,1}^2 + \mathbf{B}_t^2 \leq (1+\alpha)\Big(1 - \frac{1}{n_m}\Big)[\sigma_{t,1}^2 + \mathbf{B}_{t-1}^2] + \frac{1+\alpha}{n_m}G_t^2 + (1 + \frac{1}{\alpha})\delta^2\chi_{t+1}^2 \,.$$

Let $1 - \gamma/n_m := (1+\alpha)(1-1/n_m) \in (1-1/n_m, 1)$. We then have: $\gamma \in (0,1)$, $1 + \alpha = \frac{n_m - \gamma}{n_m - 1}$ and $1 + \frac{1}{\alpha} = \frac{n_m - \gamma}{1-\gamma}$. The previous display can thus be reformulated as:

$$\sigma_{t+1,1}^2 + \mathbf{B}_t^2 \leq \Big(1 - \frac{\gamma}{n_m}\Big)[\sigma_{t,1}^2 + \mathbf{B}_{t-1}^2] + \frac{n_m - \gamma}{n_m(n_m-1)}G_t^2 + \frac{n_m - \gamma}{1-\gamma}\delta^2\chi_{t+1}^2 \,.$$

Let $T \geq 3$. Applying Lemma B.4 (starting from $t = 2$), we have:

$$\sum_{t=3}^T [\sigma_{t,1}^2 + \mathbf{B}_{t-1}^2] \leq \frac{1 - \gamma/n_m}{\gamma/n_m}[\sigma_{2,1}^2 + \mathbf{B}_1^2] + \frac{1}{\gamma/n_m}\sum_{t=2}^{T-1}\Big[\frac{n_m - \gamma}{n_m(n_m-1)}G_t^2 + \frac{n_m - \gamma}{1-\gamma}\delta^2\chi_{t+1}^2\Big] \,.$$

Adding $\sigma_{2,1}^2$ to both sides and dropping the non-negative $\mathbf{B}_t^2$, we obtain:

$$\sum_{t=2}^T \sigma_{t,1}^2 \leq \frac{n_m}{\gamma}\sigma_{2,1}^2 + \frac{n_m - \gamma}{\gamma}\mathbf{B}_1^2 + \frac{n_m - \gamma}{\gamma(n_m-1)}\sum_{t=2}^{T-1} G_t^2 + \frac{n_m(n_m - \gamma)}{\gamma(1-\gamma)}\delta^2\sum_{t=2}^{T-1}\chi_{t+1}^2 \,.$$

Recall that $\mathbf{b}_i^1 = \nabla f_i(\mathbf{x}^1)$ for all $i \in [n]$. It holds that:

$$\sigma_{2,1}^2 = \hat{\sigma}_{2,1}^2 = \frac{1}{n}\sum_{i=1}^n [\|(\nabla f_i(\mathbf{x}^2) - \nabla f_i(\mathbf{x}^1)) - (\nabla f(\mathbf{x}^2) - \nabla f(\mathbf{x}^1))\|^2 \overset{(2.2)}{\leq} \delta^2\hat{\chi}_2^2 = \delta^2\chi_2^2,\ \mathbf{B}_1^2 = \frac{n_m - \gamma}{n_m - 1}G_1^2 \,.$$

It follows that:

$$\sum_{t=2}^T \sigma_{t,1}^2 \leq \frac{(n_m - \gamma)^2}{\gamma(n_m-1)}G_1^2 + \frac{n_m - \gamma}{\gamma(n_m-1)}\sum_{t=2}^{T-1} G_t^2 + \frac{n_m(n_m-\gamma)}{\gamma(1-\gamma)}\delta^2\sum_{t=1}^{T-1}\chi_{t+1}^2 \,.$$

Let us choose γ which minimizes the coefficient in front of $\sum_{t=1}^{T-1}\chi_{t+1}^2$ over $(0,1)$. By Lemma B.5, we get $\gamma^\star = n_m - \sqrt{n_m^2 - n_m}$. Substituting $\gamma = \gamma^\star$, we have:

$$\sum_{t=2}^T \sigma_{t,1}^2 \leq (\sqrt{n_m^2 - n_m} + n_m)G_1^2 + \Big(1 + \frac{\sqrt{n_m}}{\sqrt{n_m - 1}}\Big)\sum_{t=2}^{T-1} G_t^2 + n_m(\sqrt{n_m} + \sqrt{n_m - 1})^2\delta^2\sum_{t=1}^{T-1}\chi_{t+1}^2$$

$$\leq 2n_m G_1^2 + \Big(1 + \frac{\sqrt{n_m}}{\sqrt{n_m - 1}}\Big)\sum_{t=2}^{T-1} G_t^2 + 4n_m^2\delta^2\sum_{t=1}^{T-1}\chi_{t+1}^2 \,.$$

Multiplying both sides by $\frac{q_m}{m}$, substituting the identity $\sigma_t^2 = \frac{q_m}{m}\sigma_{t,1}^2$ and $\frac{q_m}{m}\big(1 + \frac{\sqrt{n_m}}{\sqrt{n_m - 1}}\big) = \frac{n-m}{n-1}\frac{1}{m} + \frac{\sqrt{n-m}\sqrt{n}}{m(n-1)}$, we obtain:

$$\sum_{t=2}^{T} \sigma_t^2 \leq \frac{2n_m q_m}{m} G_1^2 + \frac{n_m - 1 + \sqrt{n_m^2 - n_m}}{(n-1)} \sum_{t=2}^{T-1} G_t^2 + 4n_m^2 \delta_m^2 \sum_{t=2}^{T} \chi_t^2 \ .$$

Adding $\sigma_0^2 = 0$ and $\sigma_1^2 = 0$ to both sides, we prove the variance bound for $T \geq 3$, since $\mathbf{G}^0 = \nabla f(\mathbf{x}^0)$ and $\mathbf{G}^1 = \nabla f(\mathbf{x}^1)$. The same inequality also holds for $T = 1$ and $T = 2$, since $\sigma_0^2 = \sigma_1^2 = 0$ and $\sigma_2^2 = \frac{q_m}{m}\sigma_{2,1}^2 \leq \frac{q_m}{m}\delta^2\chi_2^2$.

□

C.3.2 Proof of Lemma 4.3

Proof. Let $t \geq 1$. By definition, $\mathbf{G}^t = \frac{1}{m}\sum_{i \in S_t} \mathbf{G}_i^t$, where $\mathbf{G}_i^t := \nabla f_i(\mathbf{x}^t) - \nabla f_i(\mathbf{w}^t) + \nabla f(\mathbf{w}^t)$ and S_t is independent from $\mathbf{x}^t$ and $\mathbf{w}^t$. Therefore, according to Lemma B.2, we have:

$$\mathbb{E}_{S_t}[\mathbf{G}^t] = \frac{1}{n}\sum_{i=1}^{n} \mathbf{G}_i^t = \nabla f(\mathbf{x}^t) \quad \text{and} \quad \mathbb{E}_{S_t}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2] = \frac{n-m}{n-1}\frac{1}{m}\hat{\sigma}_{t,1}^2 \ ,$$

where $\hat{\sigma}_{t,1}^2 := \frac{1}{n}\sum_{i=1}^{n} \|\nabla h_i(\mathbf{x}^t) - \nabla h_i(\mathbf{w}^t)\|^2$. Since ω_{t+1} is independent of $\mathbf{x}^{t+1}$ and $\mathbf{w}^t$, we have for any $\alpha > 0$:

$$\begin{aligned}
\mathbb{E}_{\omega_{t+1}}[\hat{\sigma}_{t+1,1}^2] &= (1-p_B)\frac{1}{n}\sum_{i=1}^{n} \|\nabla h_i(\mathbf{x}^{t+1}) - \nabla h_i(\mathbf{w}^t)\|^2 \\
&\overset{(3),(B.2)}{\leq} (1-p_B)(1+\alpha)\hat{\sigma}_{t,1}^2 + (1-p_B)\Big(1 + \frac{1}{\alpha}\Big)\delta^2 \hat{\chi}_{t+1}^2 \ .
\end{aligned}$$

where $\hat{\chi}_{t+1} := \|\mathbf{x}^{t+1} - \mathbf{x}^t\|$. Let $1 - \gamma p_B := (1-p_B)(1+\alpha) \in (1-p_B, 1)$. We then have $\gamma \in (0,1)$ and $1 + 1/\alpha = \frac{1 - p_B\gamma}{p_B(1-\gamma)}$. Therefore, the previous display can be reformulated as:

$$\mathbb{E}_{\omega_{t+1}}[\hat{\sigma}_{t+1,1}^2] \leq (1-\gamma p_B)\hat{\sigma}_{t,1}^2 + \frac{(1-p_B)(1-p_B\gamma)}{p_B(1-\gamma)}\delta^2\hat{\chi}_{t+1}^2 \ .$$

Taking the expectation w.r.t, $\omega_{[t]}$ on both sides and denoting $\sigma_{t,1}^2 := \mathbb{E}_{\omega_{[t]}}[\hat{\sigma}_{t,1}^2]$, we have:

$$\sigma_{t+1,1}^2 \leq (1-\gamma p_B)\sigma_{t,1}^2 + \frac{(1-p_B)(1-p_B\gamma)}{p_B(1-\gamma)}\delta^2\chi_{t+1}^2 \ .$$

Let $T \geq 2$. Applying Lemma B.4 (starting from $t = 1$), we obtain:

$$\sum_{t=2}^{T} \sigma_{t,1}^2 \leq \frac{1-\gamma p_B}{\gamma p_B}\sigma_{1,1}^2 + \frac{(1-p_B)(1-p_B\gamma)}{p_B^2\gamma(1-\gamma)}\delta^2 \sum_{t=1}^{T-1} \chi_{t+1}^2 \ .$$

Adding $\sigma_{1,1}^2$ to both sides and using $\sigma_{1,1}^2 = \mathbb{E}_{\omega_1}[\hat{\sigma}_{1,1}^2] \leq (1-p_B)\delta^2\hat{\chi}_1^2 = (1-p_B)\delta^2\chi_1^2$, we obtain:

$$\begin{aligned}
\sum_{t=1}^{T} \sigma_{t,1}^2 &\leq \frac{1}{\gamma p_B}(1-p_B)\delta^2\chi_1^2 + \frac{(1-p_B)(1-p_B\gamma)}{p_B^2\gamma(1-\gamma)}\delta^2 \sum_{t=1}^{T-1} \chi_{t+1}^2 \\
&\leq \frac{(1-p_B)(1-p_B\gamma)}{p_B^2\gamma(1-\gamma)}\delta^2 \sum_{t=0}^{T-1} \chi_{t+1}^2 \ .
\end{aligned}$$

According to Lemma B.5, the minimizer of $\frac{(1-p_B)(1-p_B\gamma)}{p_B^2\gamma(1-\gamma)}$ over $\gamma \in (0,1)$ is $\gamma^\star = \frac{1-\sqrt{1-p_B}}{p_B}$. Substituting $\gamma = \gamma^\star$, we get:

$$\sum_{t=1}^{T} \sigma_{t,1}^2 \leq \frac{(1-p_B)\delta^2}{(1-\sqrt{1-p_B})^2} \sum_{t=1}^{T} \chi_t^2 \ .$$

Multiplying both sides by $\frac{q_m}{m}$ and using the identity $\sigma_t^2 = \mathbb{E}_{S_t,\omega_{[t]}}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2] = \frac{q_m}{m}\mathbb{E}_{\omega_{[t]}}[\hat{\sigma}_{t,1}^2] = \frac{q_m}{m}\sigma_{t,1}^2$, we have:

$$\sum_{t=1}^{T} \sigma_t^2 \leq \frac{(1-p_B)\delta_m^2}{(1-\sqrt{1-p_B})^2} \sum_{t=1}^{T} \chi_t^2 \leq \frac{4\delta_m^2}{p_B^2} \sum_{t=1}^{T} \chi_t^2 \ .$$

Adding $\sigma_0^2 = \|\mathbf{G}^0 - \nabla f(\mathbf{x}^0)\|^2 = 0$ to both sides, we get the variance bound for $T \geq 2$. The same bound holds for $T = 1$ since $\sigma_0^2 = 0$ and $\sigma_1^2 = \frac{q_m}{m}\sigma_{1,1}^2 \leq (1-p_B)\delta_m^2\chi_1^2$. □

Theorem C.4. *Let I-CGM be applied to Problem 1 with the SVRG estimator under Assumption 2.1 and 2.2. Suppose the inaccuracies in solving the subproblems satisfy* (6)*. Then by choosing* $\lambda = 3\Delta_1 + 16\delta_m/p_B$*, after* $T = \lceil \frac{(256(\Delta_1+6\delta_m/p_B)F^0}{\varepsilon^2} \rceil$ *iterations, we have* $\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \varepsilon^2$*, where* $\bar{\mathbf{x}}^T$ *is uniformly sampled from* $(\mathbf{x}^t)_{t=1}^T$*. By choosing* $p_B = \frac{C_R}{C_A\lceil n_m \rceil}$ *The communication complexity is at most* $C_A\lceil n_m \rceil + (2C_R + 1)\lceil \frac{(256(\Delta_1+6\delta_m C_A\lceil n_m \rceil/C_R)F^0}{\varepsilon^2} \rceil$.

Proof. The proof strategy is the same as the one for Theorem C.7. □

C.4 Properties of the RG Estimator

C.4.1 Proof of Lemma 5.2

Proof. Let $t \geq 0$. By the definition of RG, we obtain:

$$\begin{aligned}
\hat{\Sigma}_{t+1}^2 &= \|\mathbf{g}^{t+1} - \nabla f(\mathbf{x}^{t+1})\|^2 \\
&= \left\|(1-\beta)\mathbf{g}^t + \beta\mathbf{G}^t + \nabla f_{S_t}(\mathbf{x}^{t+1}) - \nabla f_{S_t}(\mathbf{x}^t) - \nabla f(\mathbf{x}^{t+1})\right\|^2 \\
&= \left\|(1-\beta)\big(\mathbf{g}^t - \nabla f(\mathbf{x}^t)\big) + \beta\big(\mathbf{G}^t - \nabla f(\mathbf{x}^t)\big) + \big(\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})\big)\right\|^2 \\
&= (1-\beta)^2\hat{\Sigma}_t^2 + \left\|\beta\big(\mathbf{G}^t - \nabla f(\mathbf{x}^t)\big) + \big(\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})\big)\right\|^2 \\
&\quad + 2(1-\beta)\left\langle \mathbf{g}^t - \nabla f(\mathbf{x}^t), \beta\big(\mathbf{G}^t - \nabla f(\mathbf{x}^t)\big) + \big(\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})\big)\right\rangle .
\end{aligned}$$

By Assumption 5.1, S_t is independent of $\mathbf{x}^t$, $\mathbf{x}^{t+1}$ and $\mathbf{G}^{t-1}$. Furthermore, since $\mathbf{g}^t$ is a deterministic function of $\mathbf{G}^{t-1}$, $\mathbf{x}^t$, $\mathbf{x}^{t-1}$, S_{t-1} and $\mathbf{g}^{t-1}$, by induction, S_t is also independent of $\mathbf{g}^t$. Hence, it holds that:

$$\begin{aligned}
&\mathbb{E}_{S_t}[\left\langle \mathbf{g}^t - \nabla f(\mathbf{x}^t), \beta\big(\mathbf{G}^t - \nabla f(\mathbf{x}^t)\big) + \nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})\right\rangle] \\
&= \left\langle \mathbf{g}^t - \nabla f(\mathbf{x}^t), \beta\,\mathbb{E}_{S_t}[\mathbf{G}^t - \nabla f(\mathbf{x}^t)] + \mathbb{E}_{S_t}[\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})]\right\rangle .
\end{aligned}$$

By Assumption 5.1, we have $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$. Using Lemma B.2, we have $\mathbb{E}_{S_t}[\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})] = \frac{1}{n}\sum_{i=1}^n[\nabla h_i(\mathbf{x}^t) - \nabla h_i(\mathbf{x}^{t+1})] = 0$ and

$$\mathbb{E}_{S_t}[\|\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})\|^2] \overset{(B.3)}{=} \frac{q_m}{m}\frac{1}{n}\sum_{i=1}^n\|\nabla h_i(\mathbf{x}^t) - \nabla h_i(\mathbf{x}^{t+1})\|^2 \overset{(2.2)}{\leq} \delta_m^2\hat{\chi}_{t+1}^2 .$$

Taking the expectation w.r.t. S_t on both sides of the first display, we get:

$$\begin{aligned}
&\mathbb{E}_{S_t}\left[\hat{\Sigma}_{t+1}^2\right] \\
&= (1-\beta)^2\hat{\Sigma}_t^2 + \mathbb{E}_{S_t}[\|\beta\big(\mathbf{G}^t - \nabla f(\mathbf{x}^t)\big) + \big(\nabla h_{S_t}(\mathbf{x}^t) - \nabla h_{S_t}(\mathbf{x}^{t+1})\big)\|^2] \\
&\leq (1-\beta)^2\hat{\Sigma}_t^2 + 2\beta^2\,\mathbb{E}_{S_t}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2] + 2\delta_m^2\hat{\chi}_{t+1}^2 .
\end{aligned}$$

Taking the expectation w.r.t. $S_{[t-1]}$ on both sides and substituting the notations, we get:

$$\Sigma_{t+1}^2 \leq (1-\beta)^2\Sigma_t^2 + 2\beta^2\sigma_t^2 + 2\delta_m^2\chi_{t+1}^2 .$$

Applying Lemma B.4, we get for any $T \geq 1$:

$$\sum_{t=1}^T \Sigma_t^2 \leq \frac{(1-\beta)^2}{2\beta-\beta^2}\Sigma_0^2 + \frac{2\beta}{2-\beta}\sum_{t=0}^{T-1}\sigma_t^2 + \frac{2\delta_m^2}{2\beta-\beta^2}\sum_{t=0}^{T-1}\chi_{t+1}^2 .$$

This proves the claim since $\mathbf{g}^0 = \nabla f(\mathbf{x}^0)$ and so $\Sigma_0^2 = 0$. □

C.4.2 Proof of Corollary 5.3.

Proof. Let $T \geq 1$. Note that, under Assumption 5.1, we have $\mathbb{E}_{S_{[t-1]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|]^2 = \mathbb{E}_{S_{[t-2]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|]^2 = \chi_t^2$ and $\mathbb{E}_{S_{[t-1]}}[\|\nabla f(\mathbf{x}^t)\|^2] = \mathbb{E}_{S_{[t-2]}}[\|\nabla f(\mathbf{x}^t)\|^2] = G_t^2$. Applying Lemma 4.1, we have:

$$\sum_{t=0}^T \sigma_t^2 \leq \frac{2n_m q_m}{m}G_1^2 + \frac{n_m - 1 + \sqrt{n_m^2 - n_m}}{(n-1)}\sum_{t=2}^{T-1}G_t^2 + 4n_m^2\delta_m^2\sum_{t=2}^T\chi_t^2 .$$

Applying Lemma 5.2 , we obtain:

$$\sum_{t=0}^T \Sigma_t^2 \leq \frac{2\beta}{2-\beta}\sum_{t=0}^{T-1}\sigma_t^2 + \frac{2\delta_m^2}{2\beta-\beta^2}\sum_{t=1}^T\chi_t^2 .$$

Combining the previous two displays, we have:

$$\sum_{t=0}^T \Sigma_t^2 \leq \frac{4\beta n_m q_m}{(2-\beta)m}G_1^2 + \frac{2\beta(n_m - 1 + \sqrt{n_m^2 - n_m})}{(2-\beta)(n-1)}\sum_{t=2}^{T-1}G_t^2 + \frac{8\beta^2 n_m^2\delta_m^2 + 2\delta_m^2}{2\beta-\beta^2}\sum_{t=1}^T\chi_t^2 .$$ □

C.4.3 Proof of Corollary 5.4.

Proof. Let $T \geq 1$. Applying Lemma 4.3, we get:

$$\sum_{t=0}^{T} \mathbb{E}_{S_t,\omega_{[t]}}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2] \leq \frac{4\delta_m^2}{p_B^2} \sum_{t=1}^{T} \mathbb{E}_{\omega_{[t-1]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2] \,.$$

Note that, under Assumption 5.1, S_{t-1} is independent of $\mathbf{x}^t$ and $\mathbf{x}^{t-1}$. Moreover, the first iterate that might depend on ω_{t-1} is $\mathbf{x}^{t+1}$ since $\mathbf{g}^t$ is computed using $\mathbf{x}^t$ and $\mathbf{G}^{t-1}$ which is a function of ω_{t-1}. Therefore, ω_{t-1} is also independent of $\mathbf{x}^t$ and $\mathbf{x}^{t-1}$. Hence, we have $\mathbb{E}_{S_{[t-1]},\omega_{[t-1]}}[\|\mathbf{x}^t - \mathbf{x}^{t-1}\|^2] = \mathbb{E}_{S_{[t-2]},\omega_{[t-2]}}[\hat{\chi}_t^2] = \chi_t^2$. Taking the expectation w.r.t. $S_{[t-1]}$ on both sides of the first display, we get:

$$\sum_{t=0}^{T} \sigma_t^2 \leq \frac{4\delta_m^2}{p_B^2} \sum_{t=1}^{T} \chi_t^2 \,.$$

where $\sigma_t^2 := \mathbb{E}_{S_{[t]},\omega_{[t]}}[\|\mathbf{G}^t - \nabla f(\mathbf{x}^t)\|^2]$. Applying Lemma 5.2, taking the expectation w.r.t. $\omega_{[t]}$ and substituting the identity $\mathbb{E}_{S_{[t-1]},\omega_{[t]}}[\hat{\Sigma}_t^2] = \mathbb{E}_{S_{[t-1]},\omega_{[t-1]}}[\hat{\Sigma}_t^2] = \Sigma_t^2$ and $\mathbb{E}_{S_{[t-2]},\omega_{[t]}}[\hat{\chi}_t^2] = \mathbb{E}_{S_{[t-2]},\omega_{[t-2]}}[\hat{\chi}_t^2] = \chi_t^2$, we obtain:

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{2\beta}{2-\beta} \sum_{t=0}^{T-1} \sigma_t^2 + \frac{2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \,.$$

Combining the previous two displays, we have:

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{8\beta^2\delta_m^2/p_B^2 + 2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \,. \qquad \square$$

C.5 Proofs for I-CGM with RG-SAGA

Lemma C.5. *Let $\mathbf{x}^t$ be the iterates of I-CGM-RG-SAGA and let $\mathbf{G}^t$ be the-SAGA estimator for all $t \geq 0$. Let ζ_t denote the randomness generated during the process of solving the subproblem F_{t-1} in I-CGM for any $t \geq 1$. Assume that $\{\zeta_t\}_{t=1}^{\infty}$ are mutually independent across t. Then the iterates $\{\mathbf{x}^t\}_{t=0}^{\infty}$ and the estimators $\{\mathbf{G}^t\}_{t=0}^{\infty}$ satisfy Assumption 5.1.*

Proof. The equation $\mathbb{E}_{S_t}[\mathbf{G}^t] = \nabla f(\mathbf{x}^t)$ has been proved in Lemma 4.1. We next verify the dependency of randomness. Let $t \geq 1$ and denote $S_{[t]} := (S_0, \ldots, S_t)$ and $\zeta_{[t]} := (\zeta_1, \ldots, \zeta_t)$. Assume that $\mathbf{x}^t$ is a deterministic function of $(S_{[t-2]}, \zeta_{[t]})$. Then $\mathbf{G}^t$ is a deterministic function of $(S_{[t]}, \zeta_{[t]})$ since $\mathbf{G}^t$ depends only on $\mathbf{x}_{[t]} := (\mathbf{x}^0, \mathbf{x}^1, ..., \mathbf{x}^t)$ and S_t. Next observe that $\mathbf{g}^{t-1}$ is a function of S_{t-2}, $\mathbf{x}^{t-1}$, $\mathbf{x}^{t-2}$, $\mathbf{G}^{t-2}$ and $\mathbf{g}^{t-2}$. Therefore, $\mathbf{g}^{t-1}$ is a deterministic function of $S_{[t-2]}$ and $\zeta_{[t-1]}$. Finally, from the update rule of I-CGM, $\mathbf{x}^t$ is a deterministic function of $\mathbf{g}^{t-1}$, ζ_t and $\mathbf{x}^{t-1}$. Therefore, the assumption that $\mathbf{x}^t$ is determinsitic conditioned on $(S_{[t-2]}, \zeta_{[t]})$ is satisfied. This implies that S_t is independent of $\mathbf{x}_{[t+1]}, \mathbf{G}^{t-1}, \ldots, \mathbf{G}^0$. $\square$

C.5.1 Proof of Theorem 6.1.

Proof. According to Lemma 3.4, by choosing $p = \frac{\lambda - \Delta_1}{8(L_1 + \lambda)}$, the accuracy condition (6) for solving the subproblems is satisfied. Applying Corollary 5.3 and taking the full expectation, for any $T \geq 1$, we have:

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{4\beta q_m n_m}{(2-\beta)m} G_1^2 + \frac{2\beta(n_m - 1 + \sqrt{n_m^2 - n_m})}{(2-\beta)(n-1)} \sum_{t=2}^{T-1} G_t^2 + \frac{8\beta^2 n_m^2 \delta_m^2 + 2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \,,$$

where Σ_t^2, G_t^2 and χ_t^2 are defined in Corollary 3.2. Using $\frac{1}{2-\beta} \leq 1$, $\frac{q_m}{m} \leq 1$, $\frac{n_m - 1}{n-1} \leq 1 \leq n_m$, and $\frac{\sqrt{n_m^2 - n_m}}{n-1} \leq n_m$ as $n \geq 2$, we get:

$$\sum_{t=0}^{T} \Sigma_t^2 \leq 4\beta n_m \sum_{t=1}^{T-1} G_t^2 + \Big(8\beta n_m^2 \delta_m^2 + \frac{2\delta_m^2}{\beta}\Big) \sum_{t=1}^{T} \chi_t^2 \,.$$

Let $\lambda = \frac{1}{a}\Delta_1 + b\sqrt{n_m}\delta_m$ and $\beta = \frac{1}{cn_m}$ where $0 < a < 1$ and $b, c > 0$. To achieve the error condition (7), the constants should satisfy:

$$\Big(\frac{12(\lambda + \Delta_1)^2}{(\lambda - \Delta_1)^2} + 8\Big) 4\beta n_m \leq \Big(\frac{12(1+a)^2}{(1-a)^2} + 8\Big) \frac{4}{c} \leq \frac{1}{2} \,,$$

and

$$\Big(\frac{12(\lambda+\Delta_1)^2}{(\lambda-\Delta_1)^2}+8\Big)\Big(8\beta n_m^2\delta_m^2+\frac{2\delta_m^2}{\beta}\Big)\le\Big(\frac{12(1+a)^2}{(1-a)^2}+8\Big)(8/c+2c)n_m\delta_m^2\le b^2n_m\delta_m^2\le(\lambda+\Delta_1)^2\ ,$$

which gives:

$$\Big(\frac{12(1+a)^2}{(1-a)^2}+8\Big)\le\frac{c}{2},\quad\Big(\frac{12(1+a)^2}{(1-a)^2}+8\Big)(8/c+2c)\le b^2\ . \tag{C.1}$$

Let a,b,c satisfy (C.1). We can apply Corollary 3.2 and obtain:

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2]\le\frac{32(\lambda+\Delta_1)^2}{\lambda-\Delta_1}\frac{F^0}{T}\le\frac{32(1+a)^2}{1-a}\Big(\frac{1}{a}\Delta_1+b\sqrt{n_m}\delta_m\Big)\frac{F^0}{T}\ .$$

Minimizing the coefficient in front of Δ_1 gives $a^\star=\frac{1}{3}$. Choosing $b=113$ and $c=112$, the condition (C.1) is satisfied and we have:

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2]\le\frac{256(\Delta_1+38\sqrt{n_m}\delta_m)F^0}{T}\ .$$

Therefore, to achieve $\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2]\le\varepsilon^2$, we need at most $T=\lceil\frac{(256(\Delta_1+38\sqrt{n_m}\delta_m)F^0}{\varepsilon^2}\rceil$ iterations. We next compute the communication and local complexity. At the beginning when $t=0$ and $t=1$, we need $2\lceil n_m\rceil$ communication rounds with A-CSS to compute two full gradients and the associated local complexity is 1 for each round. Additionally, 2 communication rounds with D-CSS are needed to compute $\mathbf{x}^1$ and $\mathbf{x}^2$, where the local complexity is $\frac{1}{p}$ for each round. For subsequent iterations $t\ge2$, one communication round with R-CSS is needed for updating $\mathbf{g}^t$ and its associated local complexity is 2 since each client in S_{t-1} needs to compute $\nabla f_i(\mathbf{x}^t)$ and $\nabla f_i(\mathbf{x}^{t-1})$. Then another round with D-CSS is required to compute the next iterate, where the local complexity is $\frac{1}{p}$. Therefore, the total communication complexity is at most:

$$\begin{aligned}N(\varepsilon)&=\mathbb{E}[C_AN_A+C_RN_R+N_D]\\&\le2C_A\lceil n_m\rceil+C_RT+T\\&=2C_A\lceil n_m\rceil+(C_R+1)\left\lceil\frac{(256(\Delta_1+38\sqrt{n_m}\delta_m)F^0}{\varepsilon^2}\right\rceil.\end{aligned}$$

The local complexity is bounded by:

$$\begin{aligned}K(\varepsilon)&=\mathbb{E}[N_A+N_D/p+2N_R]\\&\le2\lceil n_m\rceil+\frac{1}{p}T+2T\\&=2\lceil n_m\rceil+\Big(2+\frac{8(L_1+\lambda)}{\lambda-\Delta_1}\Big)T\\&\le2\lceil n_m\rceil+\frac{28\Delta_1+1130\sqrt{n_m}\delta_m+8L_1}{2\Delta_1+113\sqrt{n_m}\delta_m}\Big(\frac{(256(\Delta_1+38\sqrt{n_m}\delta_m)F^0}{\varepsilon^2}+1\Big)\\&\le2\lceil n_m\rceil+\frac{512(7\Delta_1+283\sqrt{n_m}\delta_m+2L_1)F^0}{\varepsilon^2}+14+\frac{4L_1}{\Delta_1+28\sqrt{n_m}\delta_m}\ .\end{aligned}$$

□

C.6 Proofs for I-CGM with RG-SVRG

Lemma C.6. *Let $\mathbf{x}^t$ be the iterates of I-CGM-RG-SVRG and let $\mathbf{G}^t$ be the-SVRG estimator for all $t\ge0$. Let ζ_t denote the randomness generated during the process of solving the subproblem F_{t-1} in I-CGM for any $t\ge1$. Assume that $\{\zeta_t\}_{t=1}^\infty$ are mutually independent across t. Then the iterates $\{\mathbf{x}^t\}_{t=0}^\infty$ and the estimators $\{\mathbf{G}^t\}_{t=0}^\infty$ satisfy Assumption 5.1.*

Proof. The equation $\mathbb{E}_{S_t}[\mathbf{G}^t]=\nabla f(\mathbf{x}^t)$ has been proved in Lemma 4.3. We next verify the dependency of randomness. Let $t\ge1$ and denote $\mathbf{x}_{[t]}=(\mathbf{x}^0,..,\mathbf{x}^t)$, $\omega_{[t]}=(\omega_1,...,\omega_t)$ and $\zeta_{[t]}=(\zeta_1,\dots,\zeta_t)$. Assume that $\mathbf{x}^t$ is a deterministic function of $(S_{[t-2]},\omega_{[t-2]},\zeta_{[t]})$. It follows that $\mathbf{w}^t$ is a deterministic function of $(S_{[t-2]},\omega_{[t]},\zeta_{[t]})$ since $\mathbf{w}^t$ depends only on $\mathbf{x}^t$, $\mathbf{w}^{t-1}$ and ω_t. Then $\mathbf{G}^t$ is deterministic conditioned on $(S_{[t]},\omega_{[t]},\zeta_{[t]})$ since $\mathbf{G}^t$ is a function of $\mathbf{x}_t$, $\mathbf{w}_t$ and S_t. Next observe that $\mathbf{g}^{t-1}$ is a function of S_{t-2}, $\mathbf{x}^{t-1}$, $\mathbf{x}^{t-2}$, $\mathbf{G}^{t-2}$ and $\mathbf{g}^{t-2}$. Hence, $\mathbf{g}^{t-1}$ is a deterministic function of $(S_{[t-2]},\omega_{[t-2]},\zeta_{[t-1]})$. Finally, from the update rule of I-CGM, $\mathbf{x}^t$ is a deterministic function of $\mathbf{g}^{t-1}$, ζ_t and $\mathbf{x}^{t-1}$. Therefore, the assumption that $\mathbf{x}^t$ is deterministic conditioned on $(S_{[t-2]},\omega_{[t-2]},\zeta_{[t]})$ is satisfied. This implies that S_t is independent of $\mathbf{x}_{[t+1]}$, $\mathbf{G}^0,\dots,\mathbf{G}^{t-1}$. □

Theorem C.7 (I-CGM-RG-SVRG). *Let I-CGM be applied to Problem 1 under Assumptions 2.1, 2.2 and 2.3, where* $\mathbf{x}^{t+1} = \mathrm{CGM}_{\mathrm{rand}}(\lambda, \hat{K}_t, \mathbf{x}^t, \mathbf{g}^t)$ *with* $\hat{K}_t \sim \mathrm{Geom}(p)$ *and* $\mathbf{g}^t$ *is generated by the RG-SVRG estimator. Then by choosing* $\lambda = 3\Delta_1 + 22\delta_m/\sqrt{p_B}$, $\beta = \frac{p_B}{2}$, *and* $p = \frac{\lambda - \Delta_1}{8(L_1+\lambda)}$, *after* $T = \lceil \frac{(256(\Delta_1 + 8\delta_m/\sqrt{p_B})F^0}{\varepsilon^2} \rceil$ *iterations, we have* $\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \varepsilon^2$, *where* $\bar{\mathbf{x}}^T$ *is is uniformly sampled from* $(\mathbf{x}^t)_{t=1}^T$. *Further let* $p_B = \frac{C_R}{C_A\lceil n_m \rceil}$. *The communication complexity is at most* $C_A\lceil n_m \rceil + (2C_R+1)\lceil \frac{(256(\Delta_1 + 8\delta_m\sqrt{C_A\lceil n_m \rceil/C_R}F^0}{\varepsilon^2} \rceil$ *and the local complexity is bounded by* $16 + \lceil n_m \rceil + \frac{1024(L_1 + 4\Delta_1 + 33\sqrt{C_A\lceil n_m \rceil/C_R}\delta_m)F^0}{\varepsilon^2} + \frac{4L_1}{\Delta_1 + 11\sqrt{C_A\lceil n_m \rceil/C_R}\delta_m}$.

Proof. According to Lemma 3.4, by choosing $p = \frac{\lambda - \Delta_1}{8(L_1+\lambda)}$, the accuracy condition (6) for solving the subproblems is satisfied. Applying Corollary 5.4 and taking the full expectation, for any $T \geq 1$, we have:

$$\sum_{t=0}^{T} \Sigma_t^2 \leq \frac{8\beta^2\delta_m^2/p_B^2 + 2\delta_m^2}{2\beta - \beta^2} \sum_{t=1}^{T} \chi_t^2 \leq \Big(\frac{8\beta\delta_m^2}{p_B^2} + \frac{2\delta_m^2}{\beta}\Big) \sum_{t=1}^{T} \chi_t^2 \ .$$

Let $\lambda = \frac{1}{a}\Delta_1 + b\delta_m/\sqrt{p_B}$ and $\beta = \frac{p_B}{c}$ where $0 < a < 1$ and $b, c > 0$. To achieve the error condition (7), the constants should satisfy:

$$\Big(\frac{12(\lambda + \Delta_1)^2}{(\lambda - \Delta_1)^2} + 8\Big)\Big(\frac{8\beta\delta_m^2}{p_B^2} + \frac{2\delta_m^2}{\beta}\Big) \leq \Big(\frac{12(1+a)^2}{(1-a)^2} + 8\Big)(8/c + 2c)\delta_m^2/p_B \leq b^2\delta_m^2/p_B \leq (\lambda + \Delta_1)^2 \ ,$$

which gives:

$$\Big(\frac{12(1+a)^2}{(1-a)^2} + 8\Big)(8/c + 2c) \leq b^2 \ . \tag{C.2}$$

Let a, b, c satisfy (C.2). We can apply Corollary 3.2 and obtain:

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \frac{32(\lambda + \Delta_1)^2}{\lambda - \Delta_1}\frac{F^0}{T} \leq \frac{32(1+a)^2}{1-a}\Big(\frac{1}{a}\Delta_1 + b\delta_m/\sqrt{p_B}\Big)\frac{F^0}{T} \ .$$

Minimizing the coefficient in front of Δ_1 gives $a^\star = \frac{1}{3}$. Choosing $b = 22$ and $c = 2$, the condition (C.2) is satisfied and we have:

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \frac{256(\Delta_1 + 8\delta_m/\sqrt{p_B})F^0}{T} \ .$$

Therefore, to achieve $\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^T)\|^2] \leq \varepsilon^2$, we need at most $T = \lceil \frac{(256(\Delta_1 + 8\delta_m/\sqrt{p_B})F^0}{\varepsilon^2} \rceil$ iterations. We next compute the communication and local complexity. At iteration $t = 0$, the full gradient $\nabla f(\mathbf{x}^0)$ is computed which requires $\lceil n_m \rceil$ communication rounds with A-CSS. At each iteration $t \geq 1$, with probability p_B, the full gradient is computed, which requires $\lceil n_m \rceil$ rounds with A-CSS. The expected total number of rounds where A-CSS is used is thus bounded by: $\lceil n_m \rceil + \lceil n_m \rceil p_B T$. The associated local complexity for each round with A-CSS is always 1. For $t \geq 1$, one communication round with R-CSS is needed for updating $\mathbf{g}^t$ and its associated local complexity is 3 since the client $i \in S_{t-1}$ needs to compute $\nabla f_i(\mathbf{x}^{t-1})$, $\nabla f_i(\mathbf{w}^{t-1})$ and $\nabla f_i(\mathbf{x}^t)$. Then another round with D-CSS is established, which has the local complexity of $1/p$. Therefore, the communication complexity is bounded by:

$$N(\varepsilon) = \mathbb{E}[C_A N_A + C_R N_R + N_D] \leq C_A(\lceil n_m \rceil + \lceil n_m \rceil p_B T) + C_R T + T = C_A\lceil n_m \rceil + (C_A\lceil n_m \rceil p_B + C_R + 1)T \ .$$

Let $C_A\lceil n_m \rceil p_B = C_R$. We have

$$N(\varepsilon) \leq C_A\lceil n_m \rceil + (2C_R + 1)\left\lceil \frac{(256(\Delta_1 + 8\delta_m\sqrt{C_A\lceil n_m \rceil/C_R})F^0}{\varepsilon^2} \right\rceil \ .$$

The local complexity $K(\varepsilon)$ is bounded by:

$$\begin{aligned}
\mathbb{E}[N_A + N_D/p + 3N_R] &= \lceil n_m \rceil + \lceil n_m \rceil p_B T + T/p + 3T \\
&\leq \lceil n_m \rceil + (\frac{8(L_1 + \lambda)}{\lambda - \Delta_1} + 4)T \\
&= \lceil n_m \rceil + \frac{8L_1 + 32\Delta_1 + 264\delta_m/\sqrt{p_B}}{2\Delta_1 + 22\delta_m/\sqrt{p_B}}\big(\frac{(256(\Delta_1 + 8\delta_m/\sqrt{p_B})F^0}{\varepsilon^2} + 1\big) \\
&\leq \lceil n_m \rceil + 128\frac{(8L_1 + 32\Delta_1 + 264\delta_m/\sqrt{p_B})F^0}{\varepsilon^2} + 16 + \frac{4L_1}{\Delta_1 + 11\delta_m/\sqrt{p_B}} \ . \ \square
\end{aligned}$$

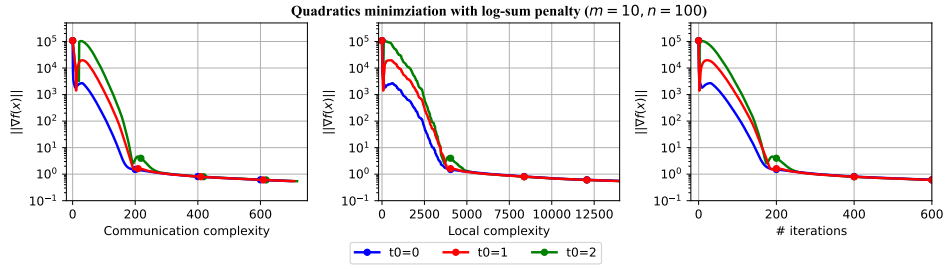

Figure E.1: Comparisons of different initialization strategies of I-CGM-RG-SAGA for solving the quadratic minimization problems with non-convex log-sum penalty.

D Discussion on the SAG estimator

SAG is another incremental gradient method [16]. SCAFFOLD has successfully applied it to the FL settings. Specifically, the local update rule of device 1 at outer iteration t (assuming no stochasticity for simplicity) is:

$$\mathbf{y}_{k+1}^t = \mathbf{y}_k^t - \frac{1}{\eta}\big(\nabla f_1(\mathbf{y}_k^t) + \mathbf{b}^t - \nabla f_1(\mathbf{x}^t)\big) \ .$$

Compared with the local CGM (8), Scaffold sets $\lambda = 0$ and uses $\mathbf{b}^t$ (10)(SAG) instead of $\mathbf{G}^t$ (SAGA) in the control variate. we next show that the variance of $\mathbf{b}^t$ cannot be controlled by δ. Let $n = 2$, $t = 1$, $\mathbf{b}_1^0 = \nabla f_1(\mathbf{x}^0)$ and $\mathbf{b}_2^0 = \nabla f_2(\mathbf{x}^0)$. Then we get: $\mathbf{b}^1 = \frac{1}{2}\big(\nabla f_1(\mathbf{x}^1) + \nabla f_2(\mathbf{x}^0)\big)$, if $S_1 = \{1\}$ and $\mathbf{b}^1 = \frac{1}{2}\big(\nabla f_2(\mathbf{x}^1) + \nabla f_1(\mathbf{x}^0)\big)$, if $S_1 = \{2\}$. Then the variance can be computed as:

$$\mathbb{E}_{S_1}\left[\left\|\mathbf{b}^1 - \nabla f(\mathbf{x}^1)\right\|^2\right] = \frac{1}{8}\sum_{i=1}^{2}\left\|\nabla f_i(\mathbf{x}^0) - \nabla f_i(\mathbf{x}^1)\right\|^2 \ .$$

While for SAGA, we have:

$$\mathbb{E}_{S_1}\left[\left\|\mathbf{G}^1 - \nabla f(\mathbf{x}^1)\right\|^2\right] = \frac{1}{2}\sum_{i=1}^{2}\left\|\nabla h_i(\mathbf{x}^0) - \nabla h_i(\mathbf{x}^1)\right\|^2 \ ,$$

where $h_i := f - f_i$. Therefore, the SAG estimator cannot fully exploit functional similarity as efficiently as SAGA in the worst case from a theoretical perspective.

E Additional details and experiments

We simulate the deep learning experiments on one NVIDIA DGX A100. All the other experiments are run on a MacBook Pro laptop.

E.1 Quadratic minimization with log-sum penalty.

Everywhere in the paper, we use the first choice of the control variate for SCAFFOLD [20]. We set the number of local steps K to be 20 and the local learning rate to be 0.003 (0.005 diverges at the beginning) for FEDAVG and SCAFFOLD. For SABER-FULL, we use the standard gradient method as the local solver and set K to be 20, local learning rate to be 0.005 and the probability for computing the full gradient to be 0.1, matching I-CGM-RG-SVRG. For GD, we run $14000 = 20 * 700$ iterations to match the local gradient computations of other algorithms. Finally, the comparisons of different initialization strategies for I-CGM-RG-SAGA can be found in Figure E.1 ($t_0 = 0, 1, 2$ correspond to computing the full gradient $0, 1, 2$ times at the beginning).

E.1.1 Ablation studies of I-CGM-RG-SAGA

Initialization strategies. The comparisons of different initialization strategies for I-CGM-RG-SAGA can be found in Figure E.1 ($t_0 = 0, 1, 2$ correspond to computing the full gradient $0, 1, 2$ times at the beginning. The result shows that the method works well without any full gradient computations.

Local steps. We now compare the performance of I-CGM-RG-SAGA under different choices of the parameter p, which is defined in Local CGM (8). Theoretically, $p \simeq \frac{\lambda}{\lambda + L_1}$. Since the expected number of local steps per iteration is $\frac{1}{p}$, a smaller p corresponds to more local computations. In the previous experiments, we used the default value $p = \frac{\delta}{L_1} \approx \frac{5}{100} = 0.05$. We now vary $p \in \{0.5, 0.05, 0.005\}$. From Figure E.2, we observe that 1) Large $p = 0.5$ results in worse communication complexity since the local accuracy condition is not fully

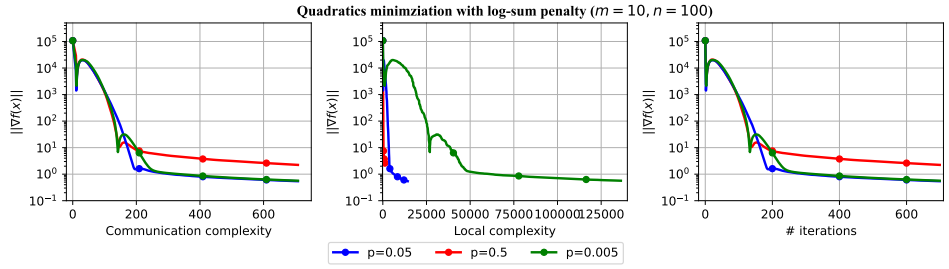

Figure E.2: Comparisons of different p (number of local steps) used in local CGM for I-CGM-RG-SAGA when solving the quadratic minimization problems with non-convex log-sum penalty.

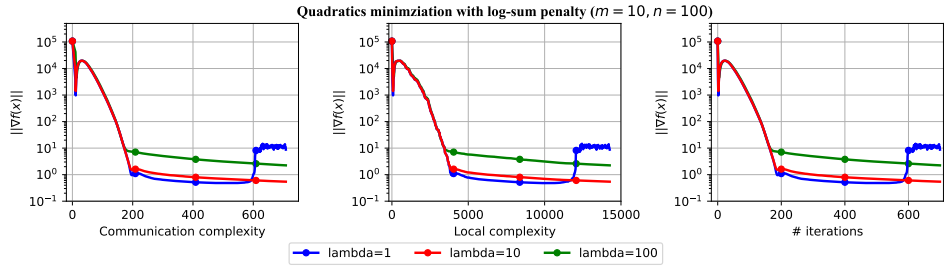

Figure E.3: Comparisons of different λ used I-CGM-RG-SAGA for solving the quadratic minimization problems with non-convex log-sum penalty.

satisfied; 2) Small $p = 0.005$ achieves similar performance to $p = 0.05$ in terms of communication complexity. This is expected, since communication complexity is determined by the fixed parameter λ. However, the local complexity becomes worse, as the total number of local steps increase and becomes unnecessarily large.

Constant λ. We now study the impact of the constant λ on the performance of I-CGM-RG-SAGA. Note that λ directly determines the iteration complexity. Theoretically the best $\lambda \simeq \Delta_1 + \sqrt{n_m}\delta$. In the previous experiments, we used the default value $\lambda = \sqrt{n_m}\delta \approx 15$. We now vary $\lambda \in \{1, 10, 100\}$. From Figure E.3, we observe that: 1) Large $\lambda = 100$ results in worse communication complexity since it does not fully use the similarity structure; 2) Small $\lambda = 1$ does not converge as the theory requires $\lambda \gtrsim \Delta_1 + \sqrt{n_m}\delta_m$, all matching the theory.

Constant β. We now test the effect of β used in the RG estimator. Both larger or smaller β can theoretically increase the variance bound (Lemma 5.2). Theoretically, the best $\beta \simeq \frac{1}{n_m}$. We now vary $\beta \in \{0.5, 0.1, 0.05, 0.01, 0.005, 0.001\}$. From Figure E.4, we see that $\beta \in [0.05, 0.5]$ results in relatively better performance as $\frac{1}{n_m} = 0.1$ and the values that fall outside this range lead to worse communication complexity.

Ratio $\frac{C_A}{C_R}$. In the main text, we report results under the extreme setting where $C_A = C_R = 1$. Now we test how increasing the ratio C_A/C_R affects the performance. Specifically, we vary $C_A \in \{1, 5, 10, 20\}$ while keeping $C_R = 1$, and repeat the same experiments. From Figure E.5, we observe that the performance of I-CGM-RG-SVRG degrades as C_A increases since each use of A-CSS becomes more costly. In contrast,

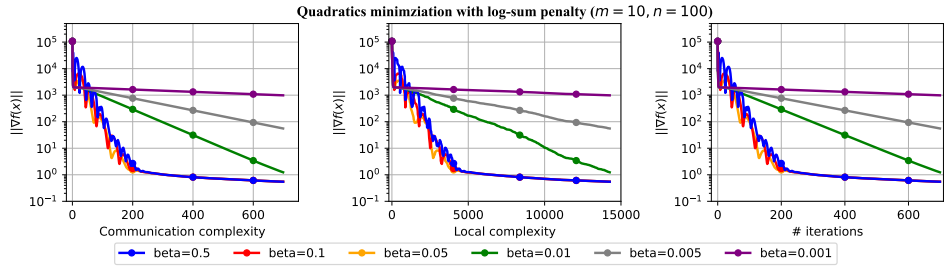

Figure E.4: Comparisons of different β used I-CGM-RG-SAGA for solving the quadratic minimization problems with non-convex log-sum penalty.

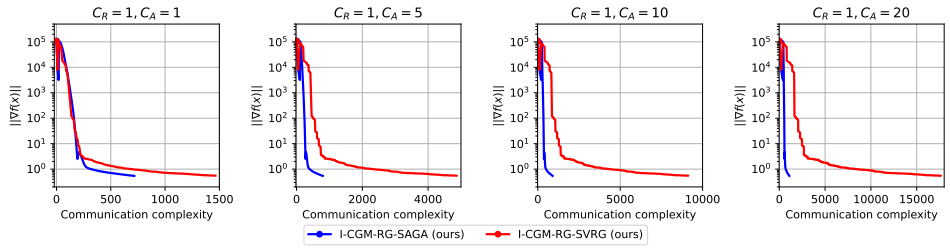

Figure E.5: Comparisons of I-CGM-RG-SAGA against I-CGM-RG-SVRG under different C_A/C_R for solving the quadratic minimization problems with non-convex log-sum penalty.

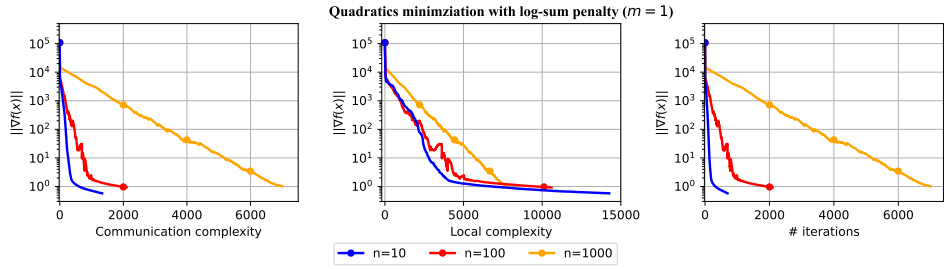

Figure E.6: Comparisons of I-CGM-RG-SAGA under different n_m for solving the quadratic minimization problems with non-convex log-sum penalty.

I-CGM-RG-SAGA remains largely unaffected, as ASS is only used during initialization. This result further confirms the advantage of I-CGM-RG-SAGA in settings where full synchronization is costly.

Ratio $\frac{n}{m}$. Finally, we examine how the ratio $\frac{n}{m}$ influences the performance of our method. Theoretically, both the communication and local complexities scale with $\sqrt{n_m}\delta_m F^0/\varepsilon^2$. We fix $m = 1$ and vary $n \in \{10, 100, 1000\}$. The datasets are generated in a consistent manner so that the values of δ and $L_{\max}$ remain approximately unchanged. We set $\lambda = \sqrt{n_m}\delta \approx 5\sqrt{n_m}$, $\beta = \frac{1}{n_m}$ and $p = \frac{\lambda}{\lambda + L_{\max}} \approx \frac{\lambda}{\lambda+100}$ with $n_m = n$. From Figure E.6, we observe that increasing n_m indeed leads to higher communication complexity. However, the growth is moderate: the additional cost scales by roughly $\sqrt{100}/\sqrt{10} = \sqrt{1000}/\sqrt{100} \approx 3$ rather than linearly $100/10 = 100/10 = 10$, confirming that the dependence is on $\sqrt{n_m}$ instead of n_m.

E.2 Logistic regression with nonconvex regularizer.

For both datasets, we set $p = 0.1$ in Local GD for CGM-RG methods and ScaffNew, and use $K = 10$ local steps for the other algorithms. We select the best local learning rate for each method from $\{0.1, 0.2, 0.5, 1.0\}$ for Mushroom and $\{0.002, 0.001, 0.0005\}$ for Duke. For proximal-point methods, we choose the best λ from $\{10, 1, 0.1, 0.01\}$ on both datasets. We use $\beta = \frac{m}{n}$ for both I-CGM-RG methods.

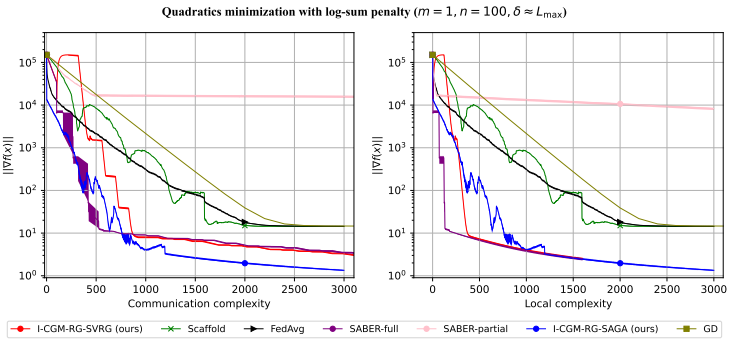

Figure E.7: Comparisons of different algorithms for solving the quadratic minimization problems with non-convex log-sum penalty when $\delta \approx L_{\max}$ where the thickness reflects the error bar.

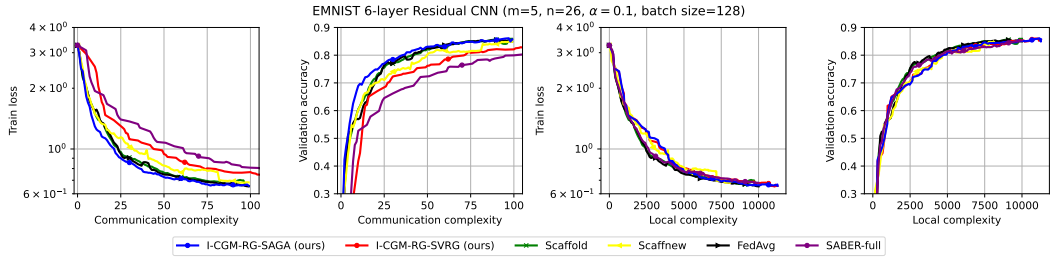

Figure E.8: Comparisons of different algorithms on the EMNIST dataset using a 6-layer residual CNN.

E.3 EMNIST with Residual CNN

We now extend our study to neural network training. Specifically, we train a 6-layer Residual CNN on the EMNIST dataset [49], which consists of a collection of 26 letter classes. We use $n = 26$ and $m = 5 \approx \sqrt{n}$, and split the dataset according to the Dirichlet distribution with $\alpha = 0.1$ (the smaller the α, the higher the heterogeneity, $\alpha = 0.1$ is highly heterogeneous). We use a batch size of 128 for computing both the local stochastic gradient and the control variates. For all the methods that use control variates, including I-CGM-RG, SCAFFOLD, SABER and SCAFFNEW, we add a damping factor q in front of the control variate to enhance their empirical performance, i.e., on line 5 of Algorithm 8, we use $\mathbf{y}_{k+1}^t = \arg\min_{\mathbf{y}\in\mathbb{R}^d} \{f_1(\mathbf{y}_k^t) + \langle \mathbf{g}_1(\mathbf{y}_k^t) + q(\mathbf{g}^t - \mathbf{g}_1(\mathbf{x}^t)), \mathbf{y} - \mathbf{y}_k^t\rangle + \frac{\eta}{2}\left\|\mathbf{y} - \mathbf{y}_k^t\right\|^2 + \frac{\lambda}{2}||\mathbf{y} - \mathbf{x}^t||^2\}$, where $q \in (0, 1]$ is a tuned parameter and $\mathbf{g}_1$ is the stochastic mini-batch gradient of ∇f_1. This approach is suggested by [50]. We report the best local stepsize $\frac{1}{\eta}$ among $\{0.05, 0.02, 0.01, 0.001\}$ and the best λ among $\{0.001, 0.01, 0.1, 1\}$. The final choices of the parameters can be found in Table E.1. The convergence behaviours can be found in Figure E.8. The best validation accuracy can be found in Table E.2, where I-CGM-RG-SAGA performs the best.

optimizers	hyper-parameters used for multi-classification tasks
I-CGM-RG-SAGA	$\frac{1}{\eta} = 0.02, \lambda = 0.01, p = 0.01, \beta = 0.2, q = 0.001, t_0 = 0$
I-CGM-RG-SVRG	$\frac{1}{\eta} = 0.02, \lambda = 0.01, p = 0.01, \beta = 0.2, q = 0.001$
SCAFFOLD [20]	$\frac{1}{\eta} = 0.02, K = 100, q = 0.001$
FEDAVG [1]	$\frac{1}{\eta} = 0.02, K = 100$
SCAFFNEW [51]	$\frac{1}{\eta} = 0.02, p = 0.01, q = 0.001$
SABER [12]	$\frac{1}{\eta} = 0.02, \lambda = 0.01, p = 0.01, \beta = 0.2, q = 0.001$

Table E.1: Hyper-parameters of the considered optimizers used in the multi-classification task for the EMNIST dataset.

Optimizers	I-CGM-RG-SAGA	I-CGM-RG-SVRG	SABER-FULL	SCAFFOLD	SCAFFNEW	FEDAVG
Accuracy	**86.2**	86.0	85.3	85.9	84.9	85.6

Table E.2: Comparisons of validation accuracy for different optimizers used in the multi-classification task for the EMNIST dataset within 100 outer iterations.

E.4 CIFAR10 with ResNet18

We now consider multi-class classification tasks with CIFAR10 [52] using ResNet18 [53]. We use $n = 10$ and $m = 3 \approx \sqrt{n}$, and split the dataset according to the Dirichlet distribution with $\alpha = 0.1$ (highly heterogeneous). We use a batch size of 128 for computing both the local stochastic gradient and the control variates $\mathbf{m}$. We report the best local stepsize $\frac{1}{\eta}$ among $\{0.1, 0.05, 0.01, 0.001\}$ and the best λ among $\{0.001, 0.01, 0.1, 1\}$. The final choices of the parameters can be found in Table E.3. The convergence behaviours can be found in Figure E.9. The best validation accuracy can be found in Table E.4.

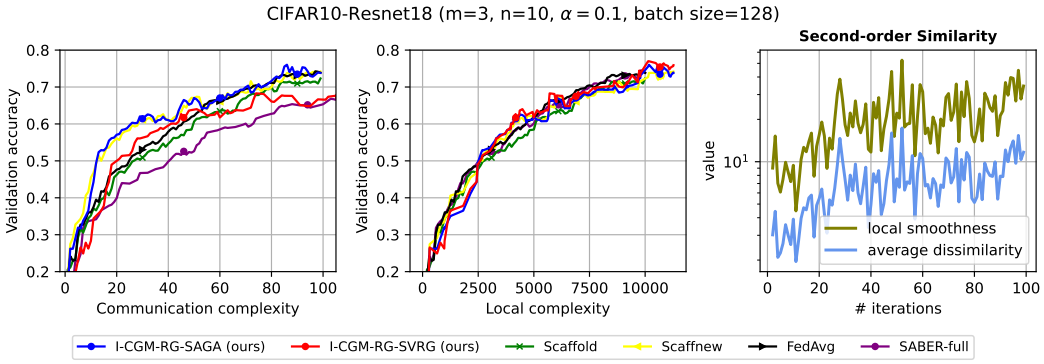

Figure E.9: Comparisons of different algorithms on the CIFAR10 dataset using ResNet18.

optimizers	hyper-parameters used for multi-classification tasks
I-CGM-RG-SAGA	$\frac{1}{\eta} = 0.05$, $\lambda = 0.01$, $p = 0.01$, $\beta = 0.2$, $q = 0.001$, $t_0 = 0$
I-CGM-RG-SVRG	$\frac{1}{\eta} = 0.05$, $\lambda = 0.01$, $p = 0.01$, $\beta = 0.2$, $q = 0.001$
SCAFFOLD [20]	$\frac{1}{\eta} = 0.05$, $K = 100$, $q = 0.001$
FEDAVG [1]	$\frac{1}{\eta} = 0.05$, $K = 100$
SCAFFNEW [51]	$\frac{1}{\eta} = 0.05$, $p = 0.01$, $q = 0.001$
SABER [12]	$\frac{1}{\eta} = 0.05$, $\lambda = 0.01$, $p = 0.01$, $\beta = 0.2$, $q = 0.001$

Table E.3: Hyper-parameters of the considered optimizers used in the multi-classification task for the CIFAR10 dataset.

Optimizers	I-CGM-RG-SAGA	I-CGM-RG-SVRG	SABER-FULL	SCAFFOLD	SCAFFNEW	FEDAVG
Accuracy	76.1	**77.0**	74.5	72.3	74.2	74.3

Table E.4: Comparisons of validation accuracy for different optimizers used in the multi-classification task for the CIFAR10 dataset within 100 outer iterations.