Federated Cubic Regularized Newton Learning with Sparsification-amplified Differential Privacy

Wei Huo [a], Changxin Liu [b], Kemi Ding [c,1], Karl Henrik Johansson [b], Ling Shi [a]

[a] *Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong*

[b] *Division of Decision and Control Systems, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, and also with Digital Futures, SE-10044 Stockholm, Sweden*

[c] *School of System Design and Intelligent Manufacturing, Southern University of Science and Technology, Shenzhen, 518055, China*

Abstract

This paper investigates the use of the cubic-regularized Newton method within a federated learning framework while addressing two major concerns that commonly arise in federated learning: privacy leakage and communication bottleneck. We introduce a federated learning algorithm called Differentially Private Federated Cubic Regularized Newton (DP-FCRN). By leveraging second-order techniques, our algorithm achieves lower iteration complexity compared to first-order methods. We also incorporate noise perturbation during local computations to ensure privacy. Furthermore, we employ sparsification in uplink transmission, which not only reduces the communication costs but also amplifies the privacy guarantee. Specifically, this approach reduces the necessary noise intensity without compromising privacy protection. We analyze the convergence properties of our algorithm and establish the privacy guarantee. Finally, we validate the effectiveness of the proposed algorithm through experiments on a benchmark dataset.

Key words: Federated learning; Cubic regularized Newton method; Differential privacy; Communication sparsification

1 Introduction

As big data continues to grow and awareness of privacy issues increases, conventional centralized methods for optimizing model parameters encounter substantial challenges. To deal with these challenges, federated learning (FL) has become a promising approach. FL allows multiple computing devices to collaborate in a distributed paradigm, under the coordination of the central server, without the need to share their local data, to optimize a shared large model. FL has found wide applications such as robotics [1], finance [2], and autonomous driving [3].

The prevailing algorithm used in FL is based on stochastic gradient descent (SGD), usually called Fed-SGD [4]. In this approach, each client trains the local model on its dataset using SGD and then uploads the gradient to the central server. The server then averages all the local gradients and performs a gradient step. Although first-order methods like SGD are the currently preferred choice for FL, they usually suffer from slow convergence, which can impede the deployment of systems that require fast and efficient processing, such as autonomous vehicles where timely and accurate predictions are vital. Newton's technique, a second-order method, is renowned for its fast convergence in optimization. However, incorporating second-order methods into FL is highly challenging. The key obstacle lies in the non-linear nature of aggregating the solutions from local optimization problems for second-order approximation, unlike the straightforward gradient aggregation. This complexity is evident in recent algorithms like GIANT [5].

While aggregating the local Hessians is possible in theory, uploading the Hessian matrices at each round incurs significant communication costs. Moreover, even without matrix transmission, communication efficiency remains a critical bottleneck in FL. For instance, if clients are some mobile devices, they usually have

Email addresses: `whuoaa@connect.ust.hk` (Wei Huo), `changxin@kth.se` (Changxin Liu), `dingkm@sustech.edu.cn` (Kemi Ding), `kallej@kth.se` (Karl Henrik Johansson), `eesling@ust.hk` (Ling Shi).

[1] Corresponding author.

limited communication bandwidth. For example, the language model GPT-3 [6] has billions of parameters, and thus, it is impractical to transmit them directly. Traditional first-order optimization methods employ various techniques to improve communication efficiency, such as communication compression [7], event-based transmission [8, 9], and partial participation [10]. As for faster second-order approaches, Safaryan et al. [11] have recently introduced a family of federated Newton learning methods that facilitate general contractive compression operators for matrices and partial participation, thereby reducing communication costs. Liu et al. [12] developed a distributed Newton's method with improved communication efficiency and achieved the super-linear convergence result. Fabbro et al. [13] presented a Newton-type algorithm to accelerate FL and considered communication constraints.

In addition to slow convergence and communication costs in traditional FL, privacy leakage is another significant concern. That the data is stored locally on the client does not alone offer adequate privacy protection. For instance, recent inference attacks in [14–16] have demonstrated that sharing local model updates or gradients between clients and the server can result in privacy breaches. Sharing second-order information can also pose privacy concerns as it often encompasses the client's data. For example, Yin et al. [17] have shown that even the eigenvalues of the Hessian matrix can disclose some critical information from original input images. Thus, it is imperative to preserve privacy in FL. Differential privacy (DP), introduced by Dwork [18], has been the de facto standard among various privacy-preserving frameworks, primarily because of its effectiveness in data analysis tasks and strong privacy guarantee. In differentially private Fed-SGD, the gradient is typically augmented with Gaussian noise to achieve DP [19–22]. It is shown that due to the composition of DP, the required noise level at each iteration is always influenced by the number of iterations. Recently, Ganesh et al. [23] devised a differentially private optimization approach using second-order techniques, attaining (ε, δ)-DP and the utility loss $O(d/\varepsilon^2)$ for d dimensional model. This utility loss bound is optimal, i.e., the best achievable for differentially private optimization [24, 25], and their algorithm achieves faster convergence. However, this method is restricted to the centralized setting. In FL, ensuring DP for second-order optimization is a novel area of exploration. We should address challenges such as integrating noise perturbation and reducing communication concurrently in federated Newton learning for each client, and enhancing the overall balance among privacy, accuracy, and communication efficiency.

Motivated by the above observations, we aim to investigate federated Newton learning while jointly considering DP and communication issues in the algorithm design. Prior research predominantly considers DP and communication efficiency as separate entities [26, 27]. While some research has explored the joint trade-off among privacy, accuracy, and communication [28, 29], they tackled the communication and privacy in a cascaded fashion, i.e., their communication schemes do not directly impact privacy preservation. In contrast, our study investigates the interplay between communication and privacy guarantees. Although some recent studies have employed compression in uplink transmission to improve privacy [30, 31], these approaches are limited to first-order learning with slow convergence. Besides, Chen et al. [31] exclusively addressed central DP, which is less robust compared to privacy mechanisms at the client level. Specifically, we propose that each local machine runs a cubic regularized Newton method to update its model, with the addition of noise perturbation during the local computation process. In FL, since the local model update is usually sparse, we propose integrating perturbation with random sparsification to boost privacy preservation. Since sparsification remains only a subset of coordinates and sets values in other coordinates to zero, it reduces the sensitivity of updates to raw data, and results in lower privacy loss during each communication round. We show that the necessary noise intensity for DP is directly proportional to the number of transmitted coordinates. This implies that improved communication efficiency can reduce noise perturbation without compromising privacy. Furthermore, we illustrate that the complexity of our algorithm exhibits an exponential improvement compared to that of first-order methods. As the required noise intensity often depends on the number of iterations, this acceleration further reduces the noise intensity and enhances the balance between privacy and convergence trade-off.

The main contributions of our work are summarized as follows:

1) We develop a differentially private federated cubic regularized Newton learning algorithm (**Algorithm 1**), called DP-FCRN. By using second-order Newton methods, the proposed algorithm achieves rapid convergence. We exploit noise perturbation in local computations to guarantee privacy preservation (**Algorithm 2**). Moreover, we use sparsification to improve communication efficiency. Unlike previous studies that treat DP and communication efficiency as separate goals and neglect the impact of efficient communication on privacy [26–29], we use the inherent characteristic of sparsification to enhance privacy.
2) We demonstrate the impact of sparsification on the balance between utility and privacy. Specifically, we show that the required noise intensity is reduced by a sparsification ratio (**Theorem 1**), which means that the proposed algorithm can exploit sparsified transmission to reduce the magnitude of Gaussian noise. Moreover, we conduct a non-asymptotic analysis and obtain the utility loss and the corresponding complexity (**Theorem 2**). The utility loss is optimal and the iteration complexity for achieving the optimal utility

loss is improved over that of first-order methods.

3) We empirically evaluate our scheme on the benchmark dataset. The experiment results illustrate that our algorithm improves the model accuracy, and at the same time saves communication costs compared to Fed-SGD under the same DP guarantee.

The remainder of the paper is organized as follows. Preliminaries and the problem formulation are provided in Section 2. In Section 3, a federated cubic regularized Newton learning algorithm with sparsification-amplified DP is proposed. Then, details on the DP analysis are shown in Section 4 and the convergence analysis is presented in Section 5. In Section 6, numerical simulations are presented to illustrate the obtained results. Finally, the conclusion and future research directions are discussed in Section 7.

Notations: Let $\mathbb{R}^p$ and $\mathbb{R}^{p\times q}$ represent the set of p-dimensional vectors and $p\times q$-dimensional matrices, respectively. $I_p\in\mathbb{R}^{p\times p}$ represents a $p\times p$-dimensional identity matrix. With any positive integer, we denote $[d]$ as the set of integers $\{1,2,\ldots,d\}$. We use $[\cdot]_j$ to denote the j-th coordinate of a vector and j-th row of a matrix. Let c represent a set of integers, and we denote $[X]_c$ as a vector containing elements $[X]_j$ for $j\in c$ if X is a vector, and as a matrix with row vectors $[X]_j$ for $j\in c$ if X is a matrix. Let $\|\cdot\|$ be the ℓ_2-norm vector norm. For a convex and closed subset $\mathcal{X}\subseteq\mathbb{R}^d$, let $\Pi_{\mathcal{X}}:\mathbb{R}^d\to\mathcal{X}$ be the Euclidean projection operator, given by $\Pi_{\mathcal{X}}(x)=\arg\min_{y\in\mathcal{X}}\|y-x\|$. We use $\mathbb{P}\{\mathcal{A}\}$ to represent the probability of an event $\mathcal{A}$, and $\mathbb{E}[x]$ to be the expected value of a random variable x.

The notation $O(\cdot)$ is used to describe the asymptotic upper bound. Mathematically, $h(n)=O(g(n))$ if there exist positive constants C and n_0 such that $0\le h(n)\le Cg(n)$ for all $n\ge n_0$. Similarly, the notation $\Omega(\cdot)$ provides the asymptotic lower bound, i.e., $h(n)=\Omega(g(n))$ if there exist positive constants C and n_0 such that $0\le Cg(n)\le h(n)$ for all $n\ge n_0$. The notation $\tilde{O}(\cdot)$ is a variant of $O(\cdot)$ that ignores logarithmic factors, that is, $h(n)=\tilde{O}(g(n))$ is equivalent to $h(n)=O\left(g(n)log^k n\right)$ for some $k>0$. The notation $\Theta(\cdot)$ is defined as the tightest bound, i.e., $h(n)$ is said to be $\Theta(g(n))$ if $h(n)=O(g(n))$ and $h(n)=\Omega(g(n))$.

2 Preliminaries and Problem Formulation

This section introduces the fundamental setup of FL along with key concepts on Newton's methods with cubic regularization and DP. Subsequently, we outline the considered problem.

2.1 Basic Setup

We consider a federated setting with n clients and a central server. Each client $i\in[n]$ possesses a private local dataset $\zeta_i=\{\zeta_i^{(1)},\ldots,\zeta_i^{(m)}\}$ containing a finite set of m data samples. Moreover, each client has a private local cost function $f_i(x)=\frac{1}{m}\sum_{j=1}^m l(x,\zeta_i^{(j)})$, where $l(x,\zeta_i^{(j)})$ is the loss of model x over the data instance $\zeta_i^{(j)}$ for $j\in[m]$. With the coordination of the central server, all clients aim to train a global model x by solving the following problem while maintaining their data locally:

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

where $\mathcal{X}\subseteq\mathbb{R}^d$ is a convex and closed box constraint. Specifically, the model training process takes place locally on each client, and only the updates are sent to the server for aggregation and global updates. The optimal model parameter is defined as $x^*=\arg\min_{x\in\mathcal{X}}f(x)$.

We make the following assumptions on the loss function of the training model:

Assumption 1 *$l(\cdot,\zeta)$ is L_0-Lipschitz, L_1-smooth, and has L_2-Lipschitz Hessian for any ζ.*

Assumption 2 *$l(\cdot,\zeta)$ is μ-strongly convex for any ζ.*

Assumption 3 *$\mathcal{X}$ has a finite diameter D.*

From Assumptions 1 and 2, we infer that also $f_i(\cdot)$ and $f(\cdot)$ are μ-strongly convex, L_0-Lipschitz, L_1-smooth, and have L_2-Lipschitz Hessian.

2.2 Newton Methods with Cubic Regularization

Newton methods [32] iteratively minimize a quadratic approximation of the function $f(\cdot)$ as

$$\begin{aligned}x_{t+1}=\arg\min_{x\in\mathcal{X}}\Big\{&f(x_t)+\langle\nabla f(x_t),x-x_t\rangle\\&+\frac{1}{2}\left\langle\nabla^2 f(x_t)(x-x_t),x-x_t\right\rangle\Big\}.\end{aligned}\tag{2}$$

The Hessian matrix $\nabla^2 f(x_t)$ provides curvature information about $f(\cdot)$ at x_t. Newton's methods significantly improve the convergence speed of gradient descent by automatically adjusting the step size along each dimension based on the local curvature at each step.

The cubic regularized Newton method, initially introduced by Nesterov and Polyak [33], incorporates a second-order Taylor expansion with a cubic regularization term. In particular, the update is

$$x_{t+1} = \arg\min_{x\in\mathcal{X}} \Big\{ f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \frac{1}{2}\left\langle \nabla^2 f(x_t)(x-x_t), x - x_t \right\rangle + \frac{L_2}{6}\|x - x_t\|^3 \Big\}, \tag{3}$$

where L_2 is the Lipschitz Hessian constant in Assumption 1. The cubic upper bound of $f(x_t)$ in (3) serves as a universal upper bound regardless of the specific characteristics of the objective function. However, the function to minimize in each step of (3) does not have a closed-form solution and it is limited to a centralized single node setting, which our algorithm addresses in a federated setting as discussed in Section 3.

2.3 Threat Model and DP

Local datasets typically contain sensitive user information. If problem (1) is addressed in an insecure environment, the leakage of information could pose a threat to personal and property privacy. This paper considers the following adversary model [34]:

Definition 1 *(Adversary Model) Adversaries can be*

i) the honest-but-curious central server that follows the given protocol but may be curious about clients' private information and capable of inferring it according to the received messages.
ii) certain clients colluding among themselves or with the central server to deduce private information about other legal clients.
iii) an outside eavesdropper who can intercept all transmitted messages in the communication and training protocol but not actively destroy message transmissions.

Our considered adversary model is much stronger than some works that require a trusted third party [35,36].

DP is a privacy concept widely adopted for quantifying privacy risk. It is a characteristic of a randomized algorithm $\mathcal{A}$ where the presence or absence of an individual in a dataset cannot be determined from the output of $\mathcal{A}$ [18]. Here, we present the formal definition of DP within the context of FL.

Definition 2 *((ε,δ)-DP) The algorithm $\mathcal{A}$ is called (ε,δ)-DP, if for any neighboring dataset pair $\zeta = \cup_{i\in[n]}\zeta_i$ and $\zeta' = \cup_{i\in[n]}\zeta'_i$ that differ in one data instance and every measurable $\mathcal{O} \subseteq \mathrm{Range}(\mathcal{A})$ [2], the output distribution satisfies*

$$\mathbb{P}\{\mathcal{A}(\zeta) \in \mathcal{O}\} \le e^{\varepsilon}\mathbb{P}\{\mathcal{A}(\zeta') \in \mathcal{O}\} + \delta, \tag{4}$$

where the probability $\mathbb{P}\{\cdot\}$ is taken over the randomness of $\mathcal{A}$.

Definition 2 states that the output distributions of neighboring datasets exhibit small variation. The factor ε in (4) represents the upper bound of privacy loss by algorithm $\mathcal{A}$, and δ denotes the probability of breaking this bound. Therefore, a smaller ε corresponds to a stronger privacy guarantee. The Gaussian mechanism is one of the commonly employed techniques to achieve DP.

Lemma 1 *(Gaussian Mechanism [37]) A Gaussian mechanism $\mathcal{G}$ for a vector-valued computation $r : \zeta \to \mathbb{R}^d$ is obtained by computing the function r on the input data $\zeta_i \in \zeta$ and then adding random Gaussian noise perturbation $\nu \sim \mathcal{N}(0, \sigma^2 I_d)$ to the output, i.e,*

$$\mathcal{G} = r(\zeta) + \nu.$$

The Gaussian mechanism $\mathcal{G}$ is $\left(\frac{\sqrt{2\log(1.25/\delta)}\Delta}{\sigma}, \delta\right)$-DP for any neighboring dataset ζ and ζ', where Δ denotes the sensitivity of r, i.e., $\Delta = \sup_{\zeta,\zeta'}\|r(\zeta) - r(\zeta')\|$.

Lemma 1 indicates that achieving (ε,δ)-DP requires adjusting the noise intensity based on the privacy guarantee ε and δ, as well as the sensitivity Δ.

2.4 Problem Statement

This paper aims to answer the following questions:

(a) How can we develop a cubic regularized Newton algorithm for solving (1) in a federated setting?
(b) Can we explore the sparsification scheme to reduce communication costs while amplifying the privacy guarantee, i.e., achieving a smaller ε given σ or requiring a smaller σ given ε?
(c) What level of noise intensity, i.e., σ, is necessary to attain (ε,δ)-DP in the proposed algorithm?
(d) Is it possible to attain the best achievable utility loss under DP, i.e., $f(x_T) - f(x^*) = O(d/\varepsilon^2)$ with the output x_T? If achievable, what is the iteration complexity for achieving this optimal utility loss?

3 Main Algorithm

In this section, we present Algorithms 1 and 2 to answer problems (a) and (b) in Section 2.4.

In general, there are two approaches for integrating sparsification and privacy in FL: (1) perturb first, then spar-

[2] Range($\mathcal{A}$) denotes the set of all possible observation sequences under the algorithm $\mathcal{A}$.

Algorithm 1 DP-FCRN

```
Input: Clients' data ζ_1, ..., ζ_n, sparsification parameter k, DP parameters (ε, δ), and step size α.
Initialization: Model parameter x_0.
for t = 0, 1, ..., T − 1 do
    ▶ Server broadcasts
    Broadcast x_t to all clients
    ▶ Clients update and upload
    for each client i ∈ [n] in parallel do
        Sample ζ_{i,t} uniformly from {ζ_i^(1), ..., ζ_i^(m)} and compute the local estimate gradient ĝ_{i,t} = ∇l(x_t, ζ_{i,t}) and the local estimate Hessian Ĥ_{i,t} = ∇²l(x_t, ζ_{i,t})
        x_{i,t+1} = GMSolver(x_t, ĝ_{i,t}, Ĥ_{i,t}, τ, σ)
        y_{i,t} ← α(x_{i,t+1} − x_t) and upload S(y_{i,t}) to the server
    end for
    ▶ Server updates
    x_{t+1} = x_t + (1/n) Σ_{i∈I_t} S(y_{i,t})
end for
```

Algorithm 2 GMSolver

```
Input: Initialization θ_0, gradient g, Hessian H, the number of iterations τ, and the noise parameter σ.
for s = 0, 1, ..., τ − 1 do
    η_s = 2/(μ(s+2))
    grad_s = g + H(θ_s − θ_0) + (L_2/2)‖θ_s − θ_0‖(θ_s − θ_0)
    θ_{s+1} = Π_X [θ_s − η_s(grad_s + b_s)], where b_s ∼ N(0, σ² I_d)
end for
Return Σ_{s=0}^{τ−1} (2(s+1))/(τ(τ+1)) θ_s
```

sify, and (2) sparsify first, then perturb. The first approach is direct and adaptable since sparsification preserves DP and integrates smoothly with all current privacy mechanisms. However, in the second approach, perturbation may compromise the communication savings achieved through sparsification. Furthermore, empirical observations suggest that the first approach outperforms the second one in some scenarios [38]. Therefore, we adopt the first approach in this study.

As shown in Algorithm 1, during iteration t, the server broadcasts the parameter x_t to the clients. Then, client i randomly samples a data instance $\zeta_{i,t} \in \zeta_i$, estimates the local gradient $\hat{g}_{i,t} = \nabla l(x_t, \zeta_{i,t})$ and the local Hessian $\hat{H}_{i,t} = \nabla^2 l(x_t, \zeta_{i,t})$ using its local data to minimize a local cubic-regularized upper bound of its loss function, and then does the following update

$$x_{i,t+1} = \arg\min_{x \in \mathcal{X}} \left\{ f_i(x_t) + \langle \hat{g}_{i,t}, x - x_t \rangle + \frac{1}{2} \left\langle \hat{H}_{i,t}(x - x_t), x - x_t \right\rangle + \frac{L_2}{6} \|x - x_t\|^3 \right\}. \tag{5}$$

As there is no closed form for optimal solution to (5), the client instead employs the gradient descent method to compute $x_{i,t+1}$. To privately minimize the local cubic upper bound, Gaussian noise is added to perturb the gradient. This local solver utilizing the Gaussian mechanism is denoted GMSolver and is detailed in Algorithm 2.

Following the update of the local model parameter, each client uploads its model update $x_{i,t+1} - x_t$ to the server. To address the communication challenges in uplink transmissions, the random-k sparsifier is employed to reduce the size of the transmitted message by a factor of k/d [39]:

Definition 3 *(Random-k Sparsification): For $x \in \mathbb{R}^d$ and a parameter $k \in [d]$, the random-k sparsification operator is*

$$\mathcal{S}(x) := \frac{d}{k}(\xi_k \odot x),$$

where $\xi_k \in \{0,1\}^d$ is a uniformly random binary vector with k nonzero entries, i.e., $\|\xi_k\|_0 = k$ and $\odot$ represents the element-wise Hadamard product.

Integrating private GMSolver and random-k sparsification, the proposed algorithm simultaneously addresses privacy preservation and communication efficiency as depicted in Algorithm 1. A scaling factor $\alpha > 0$ is introduced for convergence analysis.

Remark 1 *As pointed out by Lacoste-Julien et al. [40], the output of Algorithm 2 can be computed online. Specifically, setting $z_0 = \theta_0$, and recursively defining $z_s = \rho_s \theta_s + (1 - \rho_s) z_{s-1}$ for $s \geq 1$, with $\rho_s = \frac{2}{s+1}$. It is a straightforward calculation to check that $z_\tau = \sum_{s=0}^{\tau-1} \frac{2s}{\tau(\tau+1)} \theta_s$.*

Remark 2 *Directly solving* (3) *requires substantial computational resources. Therefore, we leverage parallel cooperative solving by multiple clients to enhance learning efficiency. To address the absence of a closed-form solution in* (5)*, we propose a local training approach for its resolution. Privacy preservation is ensured through noise perturbation during local training, while sparsification in uplink transmission not only reduces communication costs but also enhances privacy protection.*

4 Privacy Analysis

In this section, we prove the privacy guarantee provided by Algorithm 1. To facilitate privacy analysis, we make the following assumption.

Assumption 4 *For any data sample $\zeta_i^{(j)} \in \zeta_i$ and $h \in [d]$, we have*

$$\left| \left[\nabla l(x, \zeta_i^{(j)}) \right]_h \right| \leq \frac{G_1}{\sqrt{d}}, \quad \left\| \left[\nabla^2 l(x, \zeta_i^{(j)}) \right]_h \right\| \leq \frac{G_2}{\sqrt{d}}$$

for any $x, v \in \mathcal{X}$ and $i \in [n]$.

Assumption 4 characterizes the sensitivity of each coordinate of gradient $\nabla l(x, \zeta_i^{(j)})$ and each row of Hessian $\nabla^2 l(x, \zeta_i^{(j)})$, and implies $\left\|\nabla l(x, \zeta_i^{(j)})\right\| \leq G_1$ and $\left\|\nabla^2 l(x, \zeta_i^{(j)})\right\|_2 \leq \left\|\nabla^2 l(x, \zeta_i^{(j)})\right\|_F \leq G_2$, which can be enforced by the gradient and Hessian clip techniques [23].

To analyze the interplay between the sparsification and privacy, let c_i^t denote the randomly selected coordinate set for client i at round t, i.e., $\mathcal{S}(\cdot) = \frac{d}{k}[\cdot]_{c_i^t}$. An important observation is that only the values in c_i^t are transmitted to the central server, i.e.,

$$\mathcal{S}(y_{i,t}) = \frac{d}{k}\left[\frac{x_{i,t+1} - x_{i,t}}{\alpha}\right]_{c_i^t} = \frac{d}{\alpha k}\left([x_{i,t+1}]_{c_i^t} - [x_{i,t}]_{c_i^t}\right).$$

The gradient update information is contained in $[x_{i,t+1}]_{c_i^t}$ and

$$[x_{i,t+1}]_{c_i^t} = \left[\sum_{s=0}^{\tau-1} \frac{2(s+1)}{\tau(\tau+1)}\theta_i^{t,s}\right]_{c_i^t} = \sum_{s=0}^{\tau-1} \frac{2(s+1)}{\tau(\tau+1)}[\theta_i^{t,s}]_{c_i^t},$$

where $\theta_i^{t,s}$ denotes the optimization variable used by client i at iteration s in Algorithm 2 and the communication round t in Algorithm 1. Based on step 4 in the GMSolver, we have

$$[\theta_i^{t,s+1}]_{c_i^t} = \left[\Pi_{\mathcal{X}}\left[\theta_i^{t,s} - \eta_s(\mathrm{grad}_i^{t,s} + b_i^{t,s})\right]\right]_{c_i^t},$$

where $\mathrm{grad}_i^{t,s}$ and $b_i^{t,s}$ are the gradient and noise used by client i at iteration s in GMSolver and the communication round t in Algorithm 1, respectively. Since projection into a box constraint does not influence the set of selected coordinators c_i^t, what matters in local computation is

$$\left[\theta_i^{t,s} - \eta_s(\mathrm{grad}_i^{t,s} + b_i^{t,s})\right]_{c_i^t} = [\theta_i^{t,s}]_{c_i^t} - \eta_s[\mathrm{grad}_i^{t,s} + b_i^{t,s}]_{c_i^t}.$$

According to the above analysis, we conclude that the crucial aspect of privacy protection lies in the sparsified noisy gradient update, which can be expressed as

$$[\mathrm{grad}_i^{t,s} + b_i^{t,s}]_{c_i^t} = [\mathrm{grad}_i^{t,s}]_{c_i^t} + [b_i^{t,s}]_{c_i^t}.$$

We observe that the sparsification makes Gaussian noises only perturb the values at coordinates within c_i^t. If noise is added only at the selected coordinates, the level of privacy remains the same. In other words, we ensure the same privacy level even when incorporating a diminished amount of additional noise, thereby enhancing the optimization accuracy. Subsequently, we only need to analyze the privacy budget of $[\mathrm{grad}_i^{t,s}]_{c_i^t}$ after adding noise $[b_i^{t,s}]_{c_i^t}$.

For client i, considering any two neighboring dataset ζ_i and ζ_i' of the same size m but with only one data sample different (e.g., $\zeta_i^{j_0}$ and $\zeta_i^{j_0}{}'$). Denote Δ as the ℓ_2-sensitivity of $[\mathrm{grad}_i^{t,s}]_{c_i^t}$, and we have

$$\begin{aligned}
&\Delta^2 \\
=&\max_{\zeta,\zeta'}\Big\| [\hat{g}_{i,t}]_{c_i^t} - [\hat{g}_{i,t}']_{c_i^t} + \left[\hat{H}_{i,t}(\theta_i^{t,s} - x_t)\right]_{c_i^t} \\
&\qquad - \left[\hat{H}_{i,t}'(\theta_i^{t,s} - x_t)\right]_{c_i^t}\Big\|^2 \\
=&\max_{\zeta,\zeta'}\Big\| \left[\nabla l(x_t, \zeta_i^{j_0}) - \nabla l(x_t, \zeta_i^{j_0}{}')\right]_{c_i^t} \\
&\qquad + \left[(\nabla^2 l(x_t, \zeta_i^{j_0}) - \nabla^2 l(x_t, \zeta_i^{j_0}{}'))(\theta_i^{t,s} - x_t)\right]_{c_i^t}\Big\|^2 \\
\leq&\frac{4k(G_1 + G_2 D)^2}{d},
\end{aligned} \tag{6}$$

where the last inequality holds from

$$\begin{aligned}
&\Big\| \left[\nabla l(x_t, \zeta_i^{j_0}) - \nabla l(x_t, \zeta_i^{j_0}{}')\right]_{c_i^t} \\
&\qquad + \left[(\nabla^2 l(x_t, \zeta_i^{j_0}) - \nabla^2 l(x_t, \zeta_i^{j_0}{}'))(\theta_i^{t,s} - x_t)\right]_{c_i^t}\Big\| \\
\leq&\left\|\left[\nabla l(x_t, \zeta_i^{j_0}) - \nabla l(x_t, \zeta_i^{j_0}{}')\right]_{c_i^t}\right\| \\
&+ \left\|\left[\nabla^2 l(x_t, \zeta_i^{j_0}) - \nabla^2 l(x_t, \zeta_i^{j_0}{}')\right]_{c_i^t}[\theta_i^{t,s} - x_t]_{c_i^t}\right\| \\
\leq&\frac{2\sqrt{k}G_1}{\sqrt{d}} + \frac{2\sqrt{k}G_2 D}{\sqrt{d}}.
\end{aligned}$$

Lemma 1 indicates that the required noise intensity for achieving (ε, δ)-DP relies on the sensitivity value. From (6), sparsification reduces the conventional sensitivity $2(G_1 + G_2 D)$ by a factor of $\sqrt{k/d}$. Consequently, sparsification decreases sensitivity, leading to a reduction in noise intensity. Theorem 1 states a sufficient condition for achieving (ε, δ)-DP based on the reduced sensitivity resulting from sparsification.

Theorem 1 *Suppose that Assumptions 1 and 3 are satisfied, and the random-k sparsifier with $k \leq d$ is used in Algorithm 1. Given parameters m, τ, $\varepsilon \in (0, 1]$, and $\delta_0 \in (0, 1]$, if*

$$\sigma^2 \geq \frac{80\tau T k \log(1.25/\delta_0)(G_1 + G_2 D)^2}{\varepsilon^2 m^2 d} \tag{7}$$

and $T \geq \frac{\varepsilon^2}{4\tau}$, then DP-FCRN is (ε, δ)-DP given certain constant $\delta \in (0, 1]$.

PROOF. The proof is provided in Appendix B.

Remark 3 *The required noise intensity is proportional to the sparsification ratio, k/d. Therefore, to achieve the same level of DP, the required noise under our algorithm can be reduced by decreasing k. In other words, the fewer transmitted bits, the less noise required for (ε, δ)-DP.*

5 Convergence Analysis

This section presents the convergence analysis of Algorithm 1. We begin by introducing some properties of random-k sparsification.

Lemma 2 *The random-k sparsification operator $\mathcal{S}(x)$ exhibits the following properties:*

$$\mathbb{E}[\mathcal{S}(x)] = x, \ \mathbb{E}\left[\|\mathcal{S}(x) - x\|^2\right] \leq \left(\frac{d}{k} - 1\right) \|x\|^2.$$

In each step of the algorithm, a global cubic upper bound function $\phi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for $f(w)$ is constructed as

$$\begin{aligned}
&\phi(v; w) \\
\triangleq& f(w) + \langle \nabla f(w), v - w \rangle + \frac{1}{2} \left\langle \nabla^2 f(w)(v - w), v - w \right\rangle \\
&+ \frac{M}{6} \|v - w\|^3, \quad \forall v \in \mathcal{X},
\end{aligned}$$

and local cubic upper bound functions $\phi_i : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for $f_i(w)$, $i \in \{1, 2, \ldots, n\}$, as

$$\begin{aligned}
&\phi_i(v; w) \\
\triangleq& f_i(w) + \langle \nabla f_i(w), v - w \rangle + \frac{1}{2} \left\langle \nabla^2 f_i(w)(v - w), v - w \right\rangle \\
&+ \frac{M}{6} \|v - w\|^3, \quad \forall v \in \mathcal{X}.
\end{aligned} \tag{8}$$

Algorithm 2 uses the typical SGD to solve (5) and the local cubic upper bound $\phi_i(x; x_t)$ is a strongly convex function. Therefore, we can obtain the suboptimality gap based on SGD analysis.

Lemma 3 *Suppose that Assumptions 1–3 hold. For every $\beta \in (0, 1)$, given parameters $\varepsilon \in (0, 1]$, $\delta_0 \in (0, 1]$, and $w \in \mathcal{X}$ the output of Algorithm 2, if we set the number of local iterations as*

$$\tau = \frac{(L_0 + L_1 D + M D^2/2)^2 \varepsilon^2 m^2}{k T \log(1/\delta_0)(G_1 + G_2 D)^2}, \tag{9}$$

and the noise as (7), then $\hat{v}$ satisfies

$$\begin{aligned}
&\mathbb{E}[\phi_{i,t}(\hat{v}; w)] - \min_{v \in \mathcal{X}} \phi_{i,t}(v; w) \\
=& O\left(\frac{k \log(1/\delta_0)(G_1 + G_2 D)^2 T}{\varepsilon^2 m^2 \mu}\right).
\end{aligned} \tag{10}$$

PROOF. The proof is provided in Appendix C.

Lemma 3 quantifies the suboptimal gap when solving (5) with Algorithm 2 for each client in every communication round. Based on this result, we are in a position to provide the convergence of DP-FCRN.

Theorem 2 *Suppose that Assumptions 1–3 hold and the random-k sparsifier with $k \leq d$ is used in Algorithm 1. Then for every $\beta \in (0, 1)$, given parameters m and $\varepsilon \in (0, 1]$, $\delta_0 \in (0, 1]$, by setting the number of local iterations as (9), the step size as $\alpha > 1$ and*

$$\alpha = O\left(\frac{k \log(1/\delta_0)(G_1 + G_2 D)^2 T}{\varepsilon^2 m^2 \mu (L_0 + L_1 D + M D^2/2) D}\right),$$

and the number of iterations in DP-FCRN to

$$\begin{aligned}
T = \Theta\Bigg(&\frac{\sqrt{L_2}(f(x_0) - f(x^*))^{\frac{1}{4}}}{\mu^{\frac{3}{4}}} \\
&+ \log\log\left(\frac{\varepsilon m}{\sqrt{k \log(1/\delta_0)}}\right)\Bigg),
\end{aligned}$$

then the output of DP-FCRN, that is, x_T, preserves (ε, δ)-DP and

$$\begin{aligned}
&\mathbb{E}[f(x_T)] - f(x^*) \\
\leq& \tilde{O}\left(\frac{k \log(1/\delta_0)(G_1 + G_2 D)^2}{\varepsilon^2 m^2 \mu} \cdot \frac{\sqrt{L_2}(f(x_0) - f(x^*))^{\frac{1}{4}}}{\mu^{\frac{3}{4}}}\right).
\end{aligned}$$

PROOF. The proof is provided in Appendix D.

Remark 4 *In existing DP algorithms for strongly convex functions, the best achievable bound for optimization error is $O\left(\frac{d}{\varepsilon^2}\right)$ [24, 25]. This indicates that the error bound derived in Theorem 2 is optimal w.r.t. the privacy loss ε. Furthermore, our result $O\left(\frac{k}{\varepsilon^2}\right)$ reduces the error bound by a factor k/d, attributed to sparsification. This result underscores how efficient communication better balances the trade-off between privacy and utility.*

Remark 5 *While DP-FCRN does not explicitly include a switching step, the proof of Theorem 2 indicates that DP-FCRN operates in two distinct phases. Initially, when x_t is distant from x^*, the convergence rate is $1/T^4$. Subsequently, as x_t approaches x^*, the algorithm transitions to the second phase with a convergence rate of $\exp(\exp(-T))$. In summary, leveraging second-order techniques in our algorithm significantly improves the oracle complexity compared to first-order methods [41].*

6 Numerical Evaluation

In this section, we evaluate the effectiveness of DP-FCRN with different compression levels and compare them to the first-order Fed-SGD with DP [42].

6.1 *Experimental Setup*

We test our algorithm on the benchmark datasets *epsilon* [43], which include 400,000 samples and 2,000 features for each sample. The data samples are evenly and randomly allocated among the $n = 40$ clients. The clients cooperatively solve the following logistic regression problem:

$$\min_{x \in \mathcal{X}} f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x),$$

where

$$f_i(x) = \frac{1}{m} \sum_{j=1}^{m} \log(1 + \exp(-b_j a_j^\top x)) + \frac{1}{2m} \|x\|^2,$$

$\mathcal{X} \subseteq [-0.5, 0.5]^d$, m is the number of samples in the local dataset, and $a_j \in \mathbb{R}^d$ and $b_j \in \{-1, 1\}$ are the data samples.

As DP parameters, we consider $\varepsilon \in \{0.4, 0.6, 0.8, 1\}$ and $\delta_0 = 0.01$. The random noise is generated according to (7) and the number of local iterations τ is determined by (9). In iteration t, client i processes one data point from ζ_i and the server updates x_t accordingly. Upon finishing processing the entire dataset, one epoch is completed. We conduct the algorithm for four epochs and repeat each experiment five times. We show the mean curve along with the region representing one standard deviation. The convergence performance of the algorithm is evaluated by training suboptimality and testing accuracy over iterations. Training suboptimality is calculated by $f(x_t) - f(x^*)$, where $f(x^*)$ is obtained using the LogisticSGD optimizer from scikit-learn [44]. Testing accuracy is determined by applying the logistic function to the entire dataset. It is calculated as the percentage of correct predictions out of the total number of predictions.

6.2 *Performance and Comparison with Fed-SGD*

By setting the privacy budget as $\varepsilon = 0.8$, we compare the convergence performance between first-order Fed-SGD with DP and Algorithm 1 with different choices of sparsification ratio $k/d \in \{0.08, 0.1, 0.2, 1\}$. Fig. 1 implies that DP-FCRN outperforms Fed-SGD with DP in terms of optimization accuracy and convergence speed. Moreover, employing a larger sparsification ratio k/d in

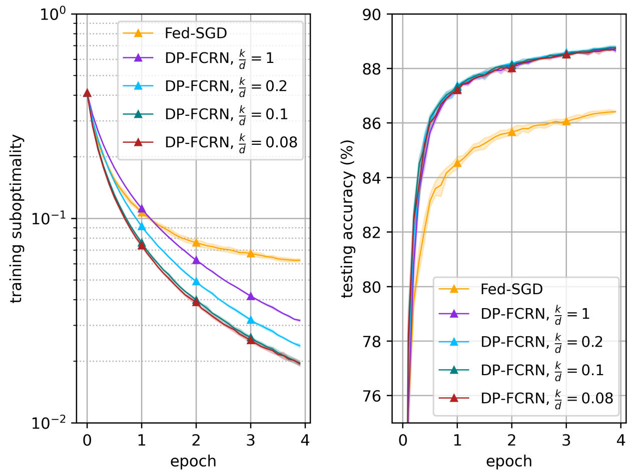

Fig. 1. Performance comparison between Fed-SGD with DP and DP-FCRN with $\varepsilon = 0.8$.

DP-FCRN results in worse training suboptimality, verifying Theorem 2. We find that keeping more coordinates in sparsification leads to more complete information transmission together with increased noise. The results shown in Fig. 1 indicate that, in certain settings, the benefit of noise reduction for convergence performance may outweigh the negative impacts arising from information completeness. On the other hand, there is no obvious difference in testing accuracy with different sparse ratios, which indicates that the performance under the proposed DP-FCRN does not deteriorate much while reducing the communication burden.

6.3 *Trade-off between Privacy and Utility*

Fig. 2 illustrates the trade-off between privacy and utility. It shows that when we increase the value of ε, i.e., relax the privacy requirement, the suboptimality will decrease across all the methods. Additionally, under a tighter DP requirement, i.e., smaller ε, the performance between DP-FCRN and Fed-SGD is more significant.

7 Conclusion and Future Work

This paper explores communication efficiency and differential privacy within federated second-order methods. We demonstrate that the inherent sparsification characteristic can bolster privacy protection. Moreover, employing second-order methods in a privacy setting can achieve the worst-case convergence guarantees and a faster convergence rate. Experiment results illustrate that our algorithm substantially outperforms first-order Fed-SGD in terms of utility loss.

There are several promising directions for future research. Firstly, investigating methods to reduce the computational complexity of federated second-order

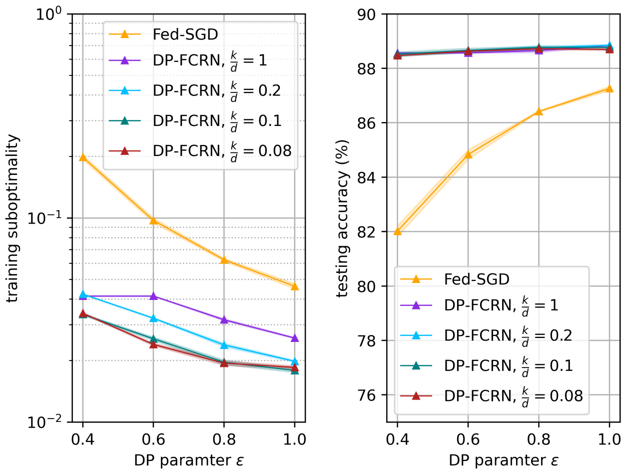

Fig. 2. Performance comparison between Fed-SGD and DP-FCRN under different DP parameters.

learning approaches is valuable. Additionally, integrating more general compression schemes and studying the privacy preservation and performance of non-convex cost functions are intriguing topics.

References

[1] Zhenyuan Yuan, Siyuan Xu, and Minghui Zhu. Federated reinforcement learning for robot motion planning with zero-shot generalization. *Automatica*, 166:111709, 2024.

[2] Yueyue Shi, Hengjie Song, and Jun Xu. Responsible and effective federated learning in financial services: A comprehensive survey. In *62nd IEEE Conference on Decision and Control*, pages 4229–4236, 2023.

[3] Anh Nguyen, Tuong Do, Minh Tran, Binh X Nguyen, Chien Duong, Tu Phan, Erman Tjiputra, and Quang D Tran. Deep federated learning for autonomous driving. In *IEEE Intelligent Vehicles Symposium*, pages 1824–1830, 2022.

[4] Nirupam Gupta, Thinh T Doan, and Nitin Vaidya. Byzantine fault-tolerance in federated local sgd under $2f$-redundancy. *IEEE Transactions on Control of Network Systems*, 10(4):1669–1681, 2023.

[5] Alessio Maritan, Ganesh Sharma, Luca Schenato, and Subhrakanti Dey. Network-GIANT: Fully distributed Newton-type optimization via harmonic Hessian consensus. In *IEEE Globecom Workshops*, pages 902–907, 2023.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[7] Peter Richtárik, Igor Sokolov, and Ilyas Fatkhullin. EF21: A new, simpler, theoretically better, and practically faster error feedback. *Advances in Neural Information Processing Systems*, 34:4384–4396, 2021.

[8] Xingkang He, Xinlei Yi, Yanlong Zhao, Karl Henrik Johansson, and Vijay Gupta. Asymptotic analysis of federated learning under event-triggered communication. *IEEE Transactions on Signal Processing*, 71:2654–2667, 2023.

[9] Wei Huo, Kam Fai Elvis Tsang, Yamin Yan, Karl Henrik Johansson, and Ling Shi. Distributed Nash equilibrium seeking with stochastic event-triggered mechanism. *Automatica*, 162:111486, 2024.

[10] Han Wang, Siddartha Marella, and James Anderson. Fedadmm: A federated primal-dual algorithm allowing partial participation. In *IEEE 61st Conference on Decision and Control*, pages 287–294, 2022.

[11] Mher Safaryan, Rustem Islamov, Xun Qian, and Peter Richtárik. FedNL: Making Newton-type methods applicable to federated learning. *arXiv preprint arXiv:2106.02969*, 2021.

[12] Huikang Liu, Jiaojiao Zhang, Anthony Man-Cho So, and Qing Ling. A communication-efficient decentralized Newton's method with provably faster convergence. *IEEE Transactions on Signal and Information Processing over Networks*, 9:427–441, 2023.

[13] Nicolò Dal Fabbro, Subhrakanti Dey, Michele Rossi, and Luca Schenato. SHED: A Newton-type algorithm for federated learning based on incremental Hessian eigenvector sharing. *Automatica*, 160:111460, 2024.

[14] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333, 2015.

[15] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in Neural Information Processing Systems*, 32, 2019.

[16] Lan Liu, Yi Wang, Gaoyang Liu, Kai Peng, and Chen Wang. Membership inference attacks against machine learning models via prediction sensitivity. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[17] Xiaoxia Yin, Brian WH Ng, Jing He, Yanchun Zhang, and Derek Abbott. Accurate image analysis of the retina using hessian matrix and binarisation of thresholded entropy with application of texture mapping. *PloS one*, 9(4):e95943, 2014.

[18] Cynthia Dwork. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12. Springer, 2006.

[19] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

[20] Stacey Truex, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. LDP-Fed: Federated learning with local differential privacy. In *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, pages 61–66, 2020.

[21] Antonious Girgis, Deepesh Data, Suhas Diggavi, Peter Kairouz, and Ananda Theertha Suresh. Shuffled model of differential privacy in federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2521–2529. PMLR, 2021.

[22] Jiaojiao Zhang, Dominik Fay, and Mikael Johansson. Dynamic privacy allocation for locally differentially private federated learning with composite objectives. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 9461–9465, 2024.

[23] Arun Ganesh, Mahdi Haghifam, Thomas Steinke, and Abhradeep Guha Thakurta. Faster differentially private convex optimization via second-order methods. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 464–473, 2014.

[25] Peter Kairouz, Ziyu Liu, and Thomas Steinke. The distributed discrete gaussian mechanism for federated learning with secure aggregation. In *International Conference on Machine Learning*, pages 5201–5212. PMLR, 2021.

[26] Zhize Li, Haoyu Zhao, Boyue Li, and Yuejie Chi. Soteriafl: A unified framework for private federated learning with communication compression. *Advances in Neural Information Processing Systems*, 35:4285–4300, 2022.

[27] Xin Zhang, Minghong Fang, Jia Liu, and Zhengyuan Zhu. Private and communication-efficient edge learning: a sparse differential gaussian-masking distributed SGD approach. In *Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, pages 261–270, 2020.

[28] Nima Mohammadi, Jianan Bai, Qiang Fan, Yifei Song, Yang Yi, and Lingjia Liu. Differential privacy meets federated learning under communication constraints. *IEEE Internet of Things Journal*, 9(22):22204–22219, 2021.

[29] Wei-Ning Chen, Christopher A Choquette-Choo, and Peter Kairouz. Communication efficient federated learning with secure aggregation and differential privacy. In *NeurIPS Workshop Privacy in Machine Learning*, 2021.

[30] Rui Hu, Yuanxiong Guo, and Yanmin Gong. Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy. *IEEE Transactions on Mobile Computing*, 2023.

[31] Wei-Ning Chen, Dan Song, Ayfer Ozgur, and Peter Kairouz. Privacy amplification via compression: Achieving the optimal privacy-accuracy-communication trade-off in distributed mean estimation. *Advances in Neural Information Processing Systems*, 36, 2024.

[32] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[33] Yurii Nesterov and Boris T Polyak. Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[34] Lingjuan Lyu, Han Yu, and Qiang Yang. Threats to federated learning: A survey, 2020.

[35] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[36] Meng Hao, Hongwei Li, Xizhao Luo, Guowen Xu, Haomiao Yang, and Sen Liu. Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Transactions on Industrial Informatics*, 16(10):6532–6542, 2019.

[37] Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403. PMLR, 2018.

[38] Jiahao Ding, Guannan Liang, Jinbo Bi, and Miao Pan. Differentially private and communication efficient collaborative learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7219–7227, 2021.

[39] Zhize Li and Peter Richtárik. Canita: Faster rates for distributed convex optimization with communication compression. *Advances in Neural Information Processing Systems*, 34:13770–13781, 2021.

[40] Simon Lacoste-Julien, Mark Schmidt, and Francis Bach. A simpler approach to obtaining an $O(1/t)$ convergence rate for the projected stochastic subgradient method. *arXiv preprint arXiv:1212.2002*, 2012.

[41] Changxin Liu, Karl H Johansson, and Yang Shi. Distributed empirical risk minimization with differential privacy. *Automatica*, 162:111514, 2024.

[42] Andrew Lowy and Meisam Razaviyayn. Private federated learning without a trusted server: Optimal algorithms for convex losses, 2023.

[43] Sören Sonnenburg, Gunnar Rätsch, Christin Schäfer, and Bernhard Schölkopf. Large scale multiple kernel learning. *The Journal of Machine Learning Research*, 7:1531–1565, 2006.

[44] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[45] Thomas Steinke. Composition of differential privacy & privacy amplification by subsampling. *arXiv preprint arXiv:2210.00597*, 2022.

Appendix

A Supporting Lemmas

The following lemma provide some useful properties of $\phi_i(v; w)$ [23].

Lemma 4 *For any $i \in \{1, 2, \dots, n\}$, ϕ_i defined in (8) has the following properties:*

1) *For any $M \geq 0$ and $w, v \in \mathcal{X}, v \neq w$, there is*

$$\begin{aligned}&\nabla_v^2 \phi_i(v; w)\\ =&\nabla^2 f_i(w) + \frac{M}{2}\|v - w\| I_d + \frac{M}{2\|v-w\|}(v-w)(v-w)^T.\end{aligned}$$

Therefore, $\nabla_v^2 \phi_i(v; w) \succeq \lambda_{\min}(\nabla^2 f_i(w)) I_d + M\|v - w\| I_d$.

2) *For any $M \geq L_2$, and $v, w \in \mathcal{X}$,*

$$f_i(v) \leq \phi_i(v; w).$$

3) *For any $M \geq 0$ and $v, w \in \mathcal{X}$,*

$$\phi_i(v; w) \leq f_i(v) + \frac{M + L_2}{6}\|v - w\|^3.$$

It can be verified that $\phi(v; w) = \frac{1}{n}\sum_{i=1}^n \phi_i(v; w)$. Therefore, we can obtain similar properties between ϕ and f as in Lemma 4.

Lemma 5 *For a sequence $\{q_t\}_{t \geq 0}$ where $q_t \geq 1$ for all $t \geq 0$, if*

$$q_{t+1} \leq q_t - \frac{1}{3} q_t^{\frac{3}{4}},$$

then

$$q_t \le \left[q_0^{\frac{1}{4}} - \frac{t}{12}\right]^4, \ \forall t \ge 0. \tag{A.1}$$

PROOF. We prove Lemma 5 by induction. For $t = 0$, inequality (A.1) is trivially true. Suppose (A.1) holds for $t = k$, i.e.,

$$q_k \le \left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^4.$$

Since the function $x - \frac{1}{3}x^{\frac{3}{4}}$ is increasing w.r.t. x, we have

$$\begin{aligned} q_{k+1} \le & q_k - \frac{1}{3}q_k^{\frac{3}{4}} \\ \le & \left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^4 - \frac{1}{3}\left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^3. \end{aligned}$$

To prove (A.1) holds true for $t = k+1$, we need to show

$$\left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^4 - \frac{1}{3}\left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^3 \le \left[q_0^{\frac{1}{4}} - \frac{k+1}{12}\right]^4. \tag{A.2}$$

Using the equality $a^4 - b^4 = (a-b)(a^3 + a^2b + ab^2 + b^3)$, inequality (A.2) is equivalent to

$$\begin{aligned} \frac{1}{3}\left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^3 \ge & \left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^4 - \left[q_0^{\frac{1}{4}} - \frac{k+1}{12}\right]^4 \\ = & \frac{1}{12}\Bigg[\left[q_0^{\frac{1}{4}} - \frac{k+1}{12}\right]^3 \\ & + \left[q_0^{\frac{1}{4}} - \frac{k+1}{12}\right]^2\left[q_0^{\frac{1}{4}} - \frac{k}{12}\right] \\ & + \left[q_0^{\frac{1}{4}} - \frac{k+1}{12}\right]\left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^2 \\ & + \left[q_0^{\frac{1}{4}} - \frac{k}{12}\right]^3\Bigg], \end{aligned}$$

which holds true since $q_0^{\frac{1}{4}} - \frac{k+1}{12} \le q_0^{\frac{1}{4}} - \frac{k}{12}$. Thus, (A.2) is established, completing the induction proof of Lemma 5.

Lemma 6 *[23] Let $b_0 > 0$ and define the sequence $a_{t+1} \le b_0 + \frac{1}{2}a_t^{\frac{3}{2}}$ where $a_0 \le \frac{16}{9}$. Then, after $T = \Theta(\log\log(\frac{1}{b}))$, we have $a_T = O(b_0)$.*

B Proof of Theorem 1

We first present some relevant properties of DP for privacy analysis.

Lemma 7 *(Privacy for Subsampling [45]) Suppose $\mathcal{G}$ is an (ε, δ)-DP mechanism. Consider $\texttt{Sample}_{r_1,r_2} : \mathcal{D}^{r_1} \to \mathcal{D}^{r_2}$ as the subsampling manipulation. Given a dataset belonging to $\mathcal{D}^{r_1}$ as an input, this subsampling manipulation selects a subset of $r_2 \le r_1$ elements from the input dataset uniformly at random. For the following mechanism*

$$\mathcal{G} \circ \texttt{Sample}_{r_1,r_2}(D),$$

where $D \in \mathcal{D}^{r_1}$. Then the mechanism $\mathcal{G} \circ \texttt{Sample}_{r_1,r_2}$ is (ε', δ')-DP for $\varepsilon' = \log(1 + r_2(e^{\varepsilon} - 1)/r_1)$ and $\delta' = r_2\delta/r_1$.

Lemma 8 *(Composition of DP [45]) If each of T randomized algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_T$ is $(\varepsilon_i, \delta_i)$-DP with $\varepsilon_i \in (0, 0.9]$ and $\delta_i \in (0, 1]$, then $\mathcal{A}$ with $\mathcal{A}(\cdot) = (\mathcal{A}_1(\cdot), \ldots, \mathcal{A}_T(\cdot))$ is $(\tilde{\varepsilon}, \tilde{\delta})$-DP with*

$$\tilde{\varepsilon} = \sqrt{\sum_{t=1}^{T} 2\varepsilon_t^2 \log\left(e + \frac{\sqrt{\sum_{t=1}^{T}\varepsilon_t^2}}{\hat{\delta}}\right)} + \sum_{t=1}^{T}\varepsilon_t^2$$

and

$$\tilde{\delta} = 1 - (1 - \hat{\delta})\prod_{t=1}^{T}(1 - \delta_t)$$

for any $\hat{\delta} \in (0, 1]$.

We first analyze DP at each local computation. The Gaussian noise injected to each coordinate in $[\mathrm{grad}_i^{t,s}]_{c_i^t}$ is generated from $\mathcal{N}(0, \sigma^2)$. Then based on Lemma 1, every local iteration in GMSolver preserves $(\varepsilon_s, \delta_0)$-DP for each sampled data $\zeta_{i,t}$ with

$$\varepsilon_s = \frac{2\sqrt{k\log(1.25/\delta_0)}(G_1 + G_2 D)}{\sigma\sqrt{d}}$$

for any $\delta_0 \in [0, 1]$.

Based on Lemma 7, each local iteration of GMSolver preserves $(\varepsilon_s', \delta_0/m)$-DP for client i's local dataset ζ_i, where

$$\varepsilon_s' = \log\left(1 + \frac{e^{\varepsilon_s} - 1}{m}\right) \le \frac{2\varepsilon_s}{m}.$$

According to the conditions on T and σ shown in Theorem 1, we have

$$\varepsilon_s'^2 \le \frac{16k\log(1.25/\delta_0)(L_0 + L_1 D)^2}{\sigma^2 m^2 d} \le \frac{\varepsilon^2}{5\tau T} \le 0.8.$$

Therefore, we have $\varepsilon_s \le 0.9$ and

$$\sum_{s=1}^{\tau T}\varepsilon_s^2 \le \frac{1}{5}\sum_{s=1}^{\tau T}\varepsilon^2 \le 1 \tag{B.1}$$

for the given $\varepsilon \in (0, 1]$.

Then we analyze DP after T iterations. After performing T communication rounds, client i conducts $T\tau$ iterations of local computation. Therefore, using Lemma 8, we obtain DP-FCRN obtains $(\tilde{\varepsilon}, \tilde{\delta})$-DP with

$$\tilde{\varepsilon} = \sqrt{\sum_{s=1}^{\tau T} 2\varepsilon_s^2 \log\left(e + \frac{\sqrt{\sum_{s=1}^{\tau T} \varepsilon_s^2}}{\tilde{\delta}}\right)} + \sum_{s=1}^{\tau T} \varepsilon_s^2$$

and $\tilde{\delta} = 1 - (1-\delta')(1-\delta_0/m)^{\tau T}$ for any $\delta' \in (0,1]$. Furthermore, there is

$$\begin{aligned}
\tilde{\varepsilon} =& \sqrt{\sum_{s=1}^{\tau T} 2\varepsilon_s^2 \log\left(e + \frac{\sqrt{\sum_{s=1}^{\tau T} \varepsilon_s^2}}{\tilde{\delta}}\right)} + \frac{1}{5}\varepsilon^2 \\
\leq& \sqrt{3\sum_{s=1}^{\tau T} \varepsilon_s^2} + \frac{1}{5}\varepsilon \\
\leq& \sqrt{\frac{3}{5}\varepsilon^2} + \frac{1}{5}\varepsilon \\
\leq& \varepsilon,
\end{aligned}$$

where the second inequality holds from (B.1). If we set $\delta' = \sqrt{\sum_{s=1}^{\tau T} \varepsilon_s^2}$ and $\delta = \tilde{\delta}$, the we have DP-FCRN preserves (ε, δ)-DP.

C Proof of Lemma 3

We can write a stochastic estimate of $\phi_i(v; w)$ as follows:

$$\begin{aligned}
&\hat{\phi}_i(v; w) \\
\triangleq& f(w) + \langle \hat{g}_i, v - w\rangle + \frac{1}{2}\left\langle \hat{H}_i(v-w), v-w \right\rangle \\
&+ \frac{M}{6}\|v-w\|^3,
\end{aligned}$$

where $\hat{g}_i$ and $\hat{H}_{i,t}$ are stochastic estimates of $\nabla f_i(w)$ and $\nabla^2 f_i(w)$. According to Algorithm 1, we find that grad_s is a stochastic gradient of $\nabla_{\theta_s}\phi(\theta_s, \theta_0)$. Based on the non-expansive property of the projection operator, we have

$$\begin{aligned}
&\mathbb{E}\left[\|\theta_{s+1} - \theta_*\|^2 | \mathcal{F}_s\right] \\
\leq& \|\theta_s - \theta_*\|^2 + \eta_s^2 \mathbb{E}\left[\|\text{grad}_s + b_s\|^2 | \mathcal{F}_s\right] \\
&- 2\eta_s \left\langle \nabla_{\theta_s}\phi_i(\theta_s; \theta_0), \theta_s - \theta_0\right\rangle \\
\leq& \|\theta_s - \theta_*\|^2 + \eta_s^2 \mathbb{E}\left[\|\text{grad}_s + b_s\|^2 | \mathcal{F}_s\right] \\
&- 2\eta_s \left[\phi_i(\theta_s; \theta_0) - \phi_i(\theta^*; \theta_0) + \frac{\mu}{2}\|\theta_s; \theta_0\|^2\right],
\end{aligned}$$

where the last inequality holds from the μ-strong convexity of $\nabla\phi_i$. By arranging the inequality, we have

$$\begin{aligned}
&\mathbb{E}[\phi_i(\theta_s; \theta_0)] - \phi_i(\theta^*; \theta_0) \\
\leq& \frac{\eta_s(L^2 + \sigma^2 d)}{2} + \left(\frac{1}{2\eta_s} - \frac{\mu}{2}\right)\mathbb{E}[\|\theta_s - \theta^*\|^2] \\
&- \frac{1}{2\eta_s}\mathbb{E}[\|\theta_{s+1} - \theta^*\|^2], \qquad \text{(C.1)}
\end{aligned}$$

where $L = L_0 + L_1 D + \frac{M}{2}D^2$. With $\eta_s = \frac{2}{\mu(s+2)}$ and multiplying the (C.1) by $s+1$, we obtain

$$\begin{aligned}
&(s+1)\left(\mathbb{E}[\phi_i(\theta_s; \theta_0)] - \phi_i(\theta^*; \theta_0)\right) \\
\leq& \frac{(s+1)(L^2 + \sigma^2 d)}{\mu(s+2)} - \frac{\mu(s+2)(s+1)}{4}\mathbb{E}[\|\theta_{s+1} - \theta^*\|^2] \\
&+ \left(\frac{\mu(s+2)(s+1)}{4} - \frac{\mu(s+1)}{2}\right)\mathbb{E}[\|\theta_s - \theta^*\|^2] \\
\leq& \frac{L^2 + \sigma^2 d}{\mu} + \frac{\mu}{4}\Big[s(s+1)\mathbb{E}\left[\|\theta_s - \theta^*\|^2\right] \\
&- (s+1)(s+2)\mathbb{E}\left[\|\theta_{s+1} - \theta^*\|^2\right]\Big].
\end{aligned}$$

By summing from $s = 0$ to $s = \tau$ of these s-weighted inequalities, we have

$$\begin{aligned}
&\sum_{s=0}^{\tau-1}(s+1)\left(\mathbb{E}[\phi_i(\theta_s; \theta_0)] - \phi_i(\theta^*; \theta_0)\right) \\
\leq& \frac{\tau(L^2 + \sigma^2 d)}{\mu} - \frac{\mu}{4}\tau(\tau+1)\mathbb{E}\left[\|\theta_\tau - \theta^*\|^2\right].
\end{aligned}$$

Thus,

$$\begin{aligned}
&\mathbb{E}\left[\phi_i\left(\frac{2}{\tau(\tau+1)}\sum_{s=0}^{\tau-1}(s+1)\theta_s; \theta_0\right)\right] - \phi_i(\theta^*; \theta_0) \\
\leq& \frac{2(L^2 + \sigma^2 d)}{\mu(\tau+1)}.
\end{aligned}$$

Therefore, after the local computation of Algorithm 2, the suboptimality gap is given by

$$O\left(\frac{L^2 + \sigma^2 d}{\mu\tau}\right).$$

Putting the value of σ in (7) obtains:

$$O\left(\left(\frac{L^2}{\mu\tau} + \frac{kT\log(1/\delta_0)(G_1 + G_2 D)^2}{\varepsilon^2 m^2 \mu}\right)\right).$$

Then, by setting the number of local iterations to $\tau = \frac{L^2\varepsilon^2 m^2}{kT\log(1/\delta_0)(G_1+G_2D)^2}$, we obtain that the subotimality is given by (10).

D Proof of Theorem 2

Using 2) in Lemma 4, we can write

$$
\begin{aligned}
&\mathbb{E}[f(x_{t+1})] - f(x^*) \\
\leq&\mathbb{E}[\phi(x_{t+1}; x_t)] - f(x^*) \\
=&\mathbb{E}\left[\phi(x_{t+1}; x_t) - \frac{1}{n}\sum_{i=1}^{n} \phi_i(x_{i,t+1}; x_t) + \frac{1}{n}\sum_{i=1}^{n} \phi_i(x_{i,t+1}; x_t)\right] \\
&- \frac{1}{n}\sum_{i=1}^{n} \min_{x^{(i)}\in\mathcal{X}} \phi_i(x^{(i)}; x_t) + \frac{1}{n}\sum_{i=1}^{n} \min_{x^{(i)}\in\mathcal{X}} \phi_i(x^{(i)}; x_t) \\
&- f(x^*) \\
\leq&\frac{1}{n}\sum_{i=1}^{n}\left[\mathbb{E}[\phi_i(x_{i,t+1}; x_t)] - \min_{x^{(i)}\in\mathcal{X}} \phi_i(x^{(i)}; x_t)\right] \\
&+ \mathbb{E}\left[\phi(x_{t+1}; x_t) - \frac{1}{n}\sum_{i=1}^{n} \phi_i(x_{i,t+1}; x_t)\right] \\
&+ \left[\min_{x\in\mathcal{X}} \phi(x; x_t) - f(x^*)\right], \qquad \text{(D.1)}
\end{aligned}
$$

where the last inequality uses the fact that

$$\frac{1}{n}\sum_{i=1}^{n} \min_{x^{(i)}\in\mathcal{X}} \phi_i(x; x_t) \leq \min_{x\in\mathcal{X}} \phi(x; x_t).$$

Since $\mathcal{X}$ is a closed and convex set and $\phi(x; x_t)$ is a strongly convex function w.r.t. x, we conclude that there exists a unique $x^*_{t+1} = \arg\min_{x\in\mathcal{X}} \phi(x; x_t)$.

At each t, we obtain an approximate minimizer of $\phi_i(x; x_t)$ based on the GMSolver:

$$
\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n}\left[\mathbb{E}[\phi_i(x_{i,t+1}; x_t)] - \min_{x^{(i)}\in\mathcal{X}} \phi_i(x^{(i)}; x_t)\right] \\
\leq&O\left(\frac{k\log(1/\delta_0)(G_1 + G_2 D)^2 T}{\varepsilon^2 m^2 \mu}\right) \triangleq \Gamma_1.
\end{aligned}
$$

Lemma 3 provides the performance guarantee of the GMSolver and shows that at each step of Algorithm 1, the optimization error in minimizing $\phi_i(x^{(i)}; x_t)$ is less than Γ_1.

For the second term of (D.1), we obtain

$$
\begin{aligned}
&\mathbb{E}\left[\phi(x_{t+1}; x_t) - \frac{1}{n}\sum_{i=1}^{n} \phi_i(x_{i,t+1}; x_t)\right] \\
=&\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[\phi_i(x_{t+1}; x_t) - \phi_i(x_{i,t+1}; x_t)\right] \\
\leq&\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[\nabla_{x_{t+1}}\phi_i(x_{t+1}; x_t)(x_{t+1} - x_{i,t+1})] \\
=&\frac{1}{n}\sum_{i=1}^{n} \nabla_{x_{t+1}}\phi_i(x_{t+1}; x_t)\mathbb{E}[(x_{t+1} - x_t) - (x_{i,t+1} - x_t)] \\
=&\frac{1}{n}\sum_{i=1}^{n} \nabla_{x_{t+1}}\phi_i(x_{t+1}; x_t)\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} \mathcal{S}_i^t(y_{i,t}) - \frac{y_{i,t}}{\alpha}\right] \\
=&\frac{1}{n}\sum_{i=1}^{n} \nabla_{x_{t+1}}\phi_i(x_{t+1}; x_t)\left(\frac{1}{n}\sum_{i=1}^{n} y_{i,t} - \frac{y_{i,t}}{\alpha}\right) \\
\leq&\frac{1}{n}\sum_{i=1}^{n} L\left(\frac{1}{n}\sum_{i=1}^{n} \|y_{i,t}\| + \|y_{i,t}\|\right) \\
\leq&2\alpha LD, \qquad \text{(D.2)}
\end{aligned}
$$

where the first inequality follows from the Lipschitz continuous of ϕ_i, the third equality holds from steps 8 and 9 in Algorithm 1, and the second inequality holds from $\alpha > 1$. Putting

$$\alpha = O\left(\frac{\Gamma_1}{LD}\right)$$

into (D.2), we have

$$\phi(x_{t+1}; x_t) - \frac{1}{n}\sum_{i=1}^{n} \phi_i(x_{i,t+1}; x_t) \leq O(\Gamma_1).$$

Then, we provide an upper bound on the last term of (D.1). We obtain the following relationship by 3) in Lemma 4.

$$
\begin{aligned}
&\min_{x\in\mathcal{X}} \phi(x; x_t) - f(x^*) \\
\leq&\min_{x\in\mathcal{X}}\left[f(x) + \frac{M + L_2}{6}\|x - x_t\|^3 - f(x^*)\right].
\end{aligned}
$$

Since $\mathcal{X}$ is a convex set and $x_t, x^* \in \mathcal{X}$, for all $\eta \in [0, 1]$, $(1-\eta)x_t + \eta x^* \in \mathcal{X}$. Therefore,

$$
\begin{aligned}
&\min_{x\in\mathcal{X}}\left[f(x_t) + \frac{M + L_2}{6}\|x - x_t\|^3 - f(x^*)\right] \\
\leq&\min_{\eta_t\in[0,1]}\left[f\left((1-\eta_t)x_t + \eta_t x^*\right) + \eta_t^3\frac{M + L_2}{6}\|x_t - x^*\|^3\right. \\
&\left.- f(x^*)\right].
\end{aligned}
$$

By the convexity of f, we have $f((1-\eta_t)x_t + \eta_t x^*) \leq f(x_t) - \eta_t\left(f(x_t) - f(x^*)\right)$. Also, strong convexity implies that $\|x_{t+1} - x^*\|^3 \leq \left[\frac{2}{\mu}(f(x_t) - f(x^*)\right]^{\frac{3}{2}}$. Thus,

$$
\begin{aligned}
&\min_{x\in\mathcal{X}} \phi(x; x_t) - f(x^*) \\
\leq&\min_{\eta_t\in[0,1]}\left\{f(x_t) - f(x^*) - \eta_t(f(x_t) - f(x^*))\right. \\
&\left.+ \eta_t^3\frac{M + L_2}{6}\left[\frac{2}{\mu}(f(x_t) - f(x^*)\right]^{\frac{3}{2}}\right\}. \qquad \text{(D.3)}
\end{aligned}
$$

Let $\lambda = \left(\frac{3}{M+L_2}\right)^2 \left(\frac{\mu}{2}\right)^3$ and $u_t = \lambda^{-1}\big(f(x_t) - f(x^*)\big)$. Based on (D.3), we can rephrase (D.1) as

$$u_{t+1} \leq \lambda^{-1}\Gamma_1 + \min_{\eta_t \in [0,1]} \left(u_t - \eta_t u_t + \frac{1}{2}\eta_t^3 u_t^{\frac{3}{2}} \right). \quad \text{(D.4)}$$

Denote $\eta_t^* = \arg\min_{\eta_t \in [0,1]} \left(u_t - \eta_t u_t + \frac{1}{2}\eta_t^3 u_t^{\frac{3}{2}} \right)$, we have that $\eta_t = \min\left\{ \sqrt{\frac{2}{3\sqrt{u_t}}}, 1 \right\}$.

We have two convergence cases according to different choices of η^*.

Phase I: If $u_t \geq \frac{4}{9}$, then $\eta^* = \sqrt{\frac{2}{3\sqrt{u_t}}}$. The iteration (D.4) will become

$$u_{t+1} \leq \lambda^{-1}\Gamma_1 + u_t - \left(\frac{2}{3}\right)^{\frac{3}{2}} u_t^{\frac{3}{4}}.$$

Phase II: If $u_t < \frac{4}{9}$, then $\eta^* = 1$. The iteration (D.4) will be given by

$$u_{t+1} \leq \lambda^{-1}\Gamma_1 + \frac{1}{2}u_t^{\frac{3}{2}}.$$

Assume that $u_0 \geq \frac{4}{9}$. In the following analysis, we will show that, $\{u_t\}_{t\in[T]}$ is a decreasing sequence. Therefore, we can conclude that there exists a time step $T_1 > 0$, such that $u_t < \frac{4}{9}$ for $t \geq T_1$. Subsequently, for $t \geq T_1$, there will be $\eta_t^* = 1$.

For the convergence of Phase I, inspired by Nesterov and Polyak [33], we let $\tilde{u}_{t+1} = \frac{9}{4}u_t$, and assume $u_t \geq \frac{3\Gamma_1}{\lambda}$. Then, there is $\tilde{u}_{t+1} \geq 1$ and the evolution of Phase I becomes:

$$\begin{aligned}\tilde{u}_{t+1} \leq& \frac{9\Gamma_1}{4\lambda} + \tilde{u}_t - \frac{2}{3}\tilde{u}_t^{\frac{3}{4}} \\ \leq& \tilde{u}_t - \frac{1}{3}\tilde{u}_t^{\frac{3}{4}}, \end{aligned} \quad \text{(D.5)}$$

where the last inequality holds from $\frac{9\Gamma_1}{4\lambda} \leq \frac{\tilde{u}_t^{\frac{3}{4}}}{3}$. According to Lemma 5, we have

$$\tilde{u}_t \leq \left[\tilde{u}_0^{\frac{1}{4}} - \frac{t}{12} \right]^4, \quad \text{(D.6)}$$

which indicates

$$\frac{9u_t}{4} \leq \left[\left(\frac{9u_0}{4}\right)^{\frac{1}{4}} - \frac{t}{12} \right]^4.$$

To make $u_{T_1^*} < \frac{4}{9}$, there is

$$\frac{9u_{T_1^*}}{4} \leq \left[\left(\frac{9u_0}{4}\right)^{\frac{1}{4}} - \frac{T_1^*}{12} \right]^4 \leq \frac{4}{9},$$

which implies that

$$T_1^* = O\left(\frac{\sqrt{M+L_2}(f(x_0) - f(x^*))^{\frac{1}{4}}}{\mu^{\frac{3}{4}}} \right). \quad \text{(D.7)}$$

Therefore, after T_1^* iterations, we enter Phase II.

For the convergence analysis of phase II, the evolution is given by

$$u_{t+1} \leq \lambda^{-1}\Gamma_1 + \frac{1}{2}u_t^{\frac{3}{2}}.$$

We define another sequence $\{w_t\}_{t\geq 0}$, with $w_0 = u_0$, $w_{t+1} = \frac{3}{4}(w_t)^{\frac{3}{2}}$. By induction, we derive for every $t \geq 0$ where $\lambda^{-1}\Gamma_1 \leq \frac{1}{4}w_t^{\frac{3}{2}}$, there is $w_{t+1} \geq u_{t+1}$. Then, we can write

$$w_{t+1} = \frac{3}{4}w_t^{\frac{3}{2}}, \; \frac{9}{16}w_{t+1} = \left(\frac{9}{16}w_t\right)^{\frac{3}{2}}.$$

Therefore, we obtain that $\log(\frac{9}{16}w_t) = (\frac{3}{2})^t \log(\frac{9}{16}w_0)$. We want to find T such that $\lambda^{-1}\Gamma_1 \leq \frac{1}{4}w_T^{\frac{3}{2}} \leq 2\lambda^{-1}\Gamma_1$, i.e., $\frac{2}{3}\log(\frac{27}{16}\lambda^{-1}\Gamma_1) \leq \log(\frac{9}{16}w_T) \leq \frac{2}{3}\log(\frac{27}{8}\lambda^{-1}\Gamma_1)$. Hence, we obtain $T = \Theta\left(\log\left(\log\left(\frac{\lambda}{\Gamma_1}\right)\right)\right)$. As a result, there is $w_{T+1} = O(\lambda^{-1}\Gamma_1)$ and $u_{T+1} \leq w_{T+1} = O(\lambda^{-1}\Gamma_1)$. Therefore, in Phase II, with the number of iterations

$$T_2^* = \tilde{\Theta}\left(\log\log\left(\frac{\varepsilon m}{\sqrt{k\log(1/\delta_0)}} \right) \right), \quad \text{(D.8)}$$

we obtain the best optimization error.

In summary, the optimization error is given by

$$\begin{aligned}& f(x_T) - f(x^*) \\ =& O\left(\frac{k\log(1/\delta_0)(G_1 + G_2 D)^2}{\varepsilon^2 m^2 \mu} \cdot T \right),\end{aligned}$$

where $T = T_1^* + T_2^*$.