Communication Efficient Differentially Private Federated Learning Using Second Order Information

Mounssif Krouka
University of Oulu
mounssif.krouka@oulu.fi

Antti Koskela
Nokia Bell Labs
antti.h.koskela@nokia-bell-labs.com

Tejas Kulkarni
Nokia Bell Labs
tejas.kulkarni@nokia-bell-labs.com

Abstract

Training machine learning models with differential privacy (DP) is commonly done using first-order methods such as DP-SGD. In the non-private setting, second-order methods try to mitigate the slow convergence of the first-order methods. The existing DP methods that use second-order information still provide faster convergence, however they cannot be easily turned into federated learning (FL) algorithms without an excessive communication cost required by the exchange of the Hessian or feature covariance information between the nodes and the server. In this paper we propose DP-FedNew, a DP method for FL that uses second-order information and results in per-iteration communication cost similar to first-order methods such as DP Federated Averaging.

Keywords

Differential Privacy, Federated Learning, Communication-Efficiency, ADMM, Newton's Method, Second-Order Optimization Methods, Distributed Optimization

1 Introduction

Commonly used federated learning methods use first-order information, i.e., gradients of the loss function, to perform model updates. However, they often suffer from slow convergence. In the literature, second-order methods which incorporate second-order information via Hessians or covariance matrices in addition to gradients are often reported to have faster convergence properties compared to first-order methods.

The goal of this work is to speed-up differentially private federated convex optimization with the help of second-order information in a communication efficient manner, without transmitting Hessian or covariance matrices. Private convex optimization has recently gained further interest due to applications in private fine-tuning of large neural networks.

In the context of FL [23], combining secure aggregation with DP [19, 22, 42] reduces the trustworthiness assumptions on a central server. Specifically, when the DP noise in the model updates is additive and the model updates are sums of user-wise updates, DP perturbations can be offloaded to clients to obtain the global model under cryptographic guarantees [41] in addition to DP's usual statistical privacy guarantees.

With the performance gap between private and non-private training shrinking rapidly, communication costs in FL can easily become a bottleneck in adoption of DP secure aggregation protocols. Several works including [8–11, 42] employ the tools from compression/sketching/quantization literature to design communication efficient DP perturbations mechanisms for distributed mean estimation compatible with secure aggregation.

In this work, we take an optimization perspective and rely on off-the-shelf DP secure aggregation primitives. In private non-FL setting, methods that use second-order information have been recently developed for private convex learning problems (e.g., private fine-tuning the last layer of a neural network [34]) and show impressive improvements in increasing the privacy-utility tradeoffs. Mehta et al. [34] consider a method called DP-FC where the DP gradients are preconditioned with the noisy feature covariance matrix that is released only once. The work by Ganesh et al. [17] gives a private second-order method with rigorous convergence analysis, with utility bounds matching the lower bounds of private empirical risk minimization Bassily et al. [5].

Unfortunately, neither of these methods seem to be easy to transfer to the FL setting. For d_x features, a distributed version of [34] requires a one time aggregation of a noisy covariance matrix of size $O(d_x^2)$ from users. The $O(d_x^2)$ term can still dominate in the total communication cost when $d_x \gg T$ (training length T). The method by Ganesh et al. [17] does not seem to be easy to transfer to the FL setting due to the inverse of a non-private Hessian in the model update. It is the main goal of this work to fill this gap in the private FL literature. We demonstrate that it is possible to save communication bandwidth with the help of a method that avoids transmitting second-order matrices however yet converges faster to models with optimal privacy-utility trade-offs.

Our Contributions. This paper proposes a DP optimization method for convex problems in FL that leverages the benefits of fast convergence of second-order methods and the communication cost of first-order methods. In particular, we build upon the work of Elgabli et al. [16] where the Newton update step is approximated using one alternating direction method of multipliers (ADMM) [7] step. The main contributions of this work are summarized as follows.

- To the best of our knowledge, for convex problems, we propose the first DP optimization method in the context of FL that uses second-order information via Hessians and has a model size communication cost. Consequently, we combine the fast convergence of second-order methods and the low communication overhead of first-order methods, while providing the formal DP guarantees for the training and final model.
- We give accurate privacy analysis for both user-level and record-level in FL setting for our algorithm, referred to as DP-FedNew, in a way that the privacy guarantees are the

Proceedings on Privacy Enhancing Technologies 2025(1), 584–612
© 2025 Copyright held by the owner/author(s).
https://doi.org/10.56553/popets-2025-0032

same as the privacy guarantees of a composed Gaussian mechanism [3].

- We carry out comprehensive experiments where we show that our proposed algorithm copes with various privacy budget values and excels in terms of test accuracy, outperforming both the baseline methods. To mitigate the excessive compute and memory requirements for use cases with large size features, we suggest a variant of DP-FedNew where we replace the Hessians with certain approximations that use the feature covariance matrices and demonstrate higher performance compared to the first-order baseline using different benchmark datasets.
- We provide an asymptotic convergence analysis for the proposed method that shows its stability.

2 Background

2.1 Differential Privacy

We first shortly review the required definitions and results on differential privacy.

An input dataset containing N data points is denoted as $D = (x_1, \dots, x_N) \in \mathcal{D}$, where $\mathcal{D}$ denotes the set of datasets of all sizes. We say that two datasets D and D' are neighbors if we get one by adding or removing one element to/from the other (denoted $D \sim D'$). We say that a mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{O}$ is (ε, δ)-DP if the output distributions for neighboring datasets are always (ε, δ)-indistinguishable.

Definition 1. *Let $\varepsilon \geq 0$ and $\delta \in [0,1]$. Mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{O}$ is (ε, δ)-DP if for every pair of neighboring datasets D, D', every measurable set $E \subset \mathcal{O}$,*

$$\mathbb{P}(\mathcal{M}(D) \in E) \leq \mathrm{e}^{\varepsilon} \mathbb{P}(\mathcal{M}(D') \in E) + \delta.$$

We call $\mathcal{M}$ tightly (ε, δ)-DP, if there does not exist $\delta' < \delta$ such that $\mathcal{M}$ is (ε, δ')-DP.

We refer to Definition (1) as *record-level DP.* In case we have n users and x_i's correspond to the whole local dataset owned by user i, $i \in [n]$, we call the corresponding DP definition *user-level DP.* [32, 33].

In this work, we provide an (ε, δ)-analysis for our methods using the hockey-stick divergence. This way, we are able to get optimal privacy parameters for the mechanisms we consider and in particular we obtain lower bounds than using, e.g., the Rényi differential privacy (RDP) [35] which is a commonly used alternative.

When analyzing iterative DP-FL training methods, we model them as adaptive compositions such that the adversary has a view on the output of all intermediate outputs. This means that we analyze mechanisms of the form

$$\begin{aligned} \mathcal{M}^{(T)}(D) = \big(&\mathcal{M}_1(D), \mathcal{M}_2(\mathcal{M}_1(D), D), \dots, \\ &\mathcal{M}_T(\mathcal{M}_1(D), \dots, \mathcal{M}_{T-1}(D), D)\big). \end{aligned} \tag{2.1}$$

In the methods we consider, each $\mathcal{M}_i$, $i \in [T]$, will correspond to a Gaussian mechanism with a given sensitivity and noise scale and thus the privacy analysis is equivalent to that of the Gaussian mechanism.

Lemma 2. *Consider an adaptive composition of T Gaussian mechanisms, each with L_2-sensitivity Δ and noise scale parameter σ. The adaptive composition is (ε, δ)-DP for*

$$\delta(\varepsilon) = \Phi\left(-\frac{\varepsilon\sigma}{\sqrt{T}\cdot\Delta} + \frac{\sqrt{T}\cdot\Delta}{2\sigma}\right) - e^{\varepsilon}\Phi\left(-\frac{\varepsilon\sigma}{\sqrt{T}\cdot\Delta} - \frac{\sqrt{T}\cdot\Delta}{2\sigma}\right).$$

2.2 Alternating Direction Method of Multipliers

Alternating direction method of multipliers represents a wide class of distributed optimization methods to minimize loss functions that are non-separable across users and have a potentially non-smooth regularizer. Many settings consider n compute workers with a local convex objective function $f_i : \mathbb{R}^d \to \mathbb{R}, \forall i \in [n]$ and dataset, communicating with the central orchestrating server. The goal for the server is to learn the global parameter $y \in \mathbb{R}^d$ solving the following problem,

$$\min_y \frac{1}{n} \sum_{i=1}^n f_i(y) + \gamma \cdot r(y),$$

where r is a non-smooth and convex regularizer function with strength $\gamma > 0$. In order to make the loss function separable across nodes, it is reformulated as

$$\min_{y, \{y_i\}_{i=1}^n} \frac{1}{n} \sum_{i=1}^n f_i(y_i) + \gamma \cdot r(y), \text{ s.t. } y_i = y, \forall i \in [n]$$
.

This constrained optimization problem is then solved by another class of algorithms called *augmented Lagrangian methods* which replace the original problem with a series of unconstrained problems, additionally *augmented* with a quadratic penalty term. As we will see in the next Section, this introduces the so called *primal* and the *dual* variables. Clients and server locally update these variables to minimize the original objective function.

Private ADMM. ADMM-based methods are often viewed as an alternative to standard gradient based optimizers (e.g., Federated averaging) and exhibit similar privacy vulnerabilities. There is a long line of work on ADMM under DP including [14, 21, 29, 38] under both centralized and decentralized settings, however we are not aware of any work that considers optimizing using second-order information while also ensuring model-sized communication cost.

3 FedNew

Let n denote the number of users and $f_i(\theta)$ the empirical loss of user i, $i \in [n]$, where $\theta \in \mathbb{R}^d$ denotes the model parameters. We consider the minimization problem

$$\min_{\theta \in \mathbb{R}^d} f(\theta) := \sum_{i=1}^n f_i(\theta).$$

The Newton iteration which is the basis for most of the second-order methods, is given as

$$\theta^{k+1} = \theta^k - \left(\sum_{i=1}^n \nabla^2 f_i(\theta^k)\right)^{-1} \sum_{i=1}^n \nabla f_i(\theta^k).$$

The same Newton iteration has been considered in the centralized case by Ganesh et al. [17]. However, extending their method to FL scenario in a communication efficient way is far from immediate since each user needs to transmit both local Hessian and gradients to take one global step. Therefore, we consider as a starting point the single pass ADMM-method called FedNew as given in Elgabli et al. [16]. This method has only $O(d)$ user-wise communication cost per iteration.

The update

$$(\sum_{i=1}^{n} \nabla^2 f_i(\theta^k))^{-1} \sum_{i=1}^{n} \nabla f_i(\theta^k)$$

in the Newton iteration is approximated such that the ADMM algorithm is applied to the augmented Lagrangian

$$\begin{aligned}\mathcal{L}(\{y_i, \lambda_i\}_{i=1}^n, y) = \sum_{i=1}^{n} y_i^T \left(\frac{1}{2}(H_i^k + \alpha I) y_i - y_i^T g_i^k \right) \\ + \sum_{i=1}^{n} \langle \lambda_i, y_i - y \rangle + \frac{\rho}{2} \sum_{i=1}^{n} \|y_i - y\|_2^2,\end{aligned}$$

where $H_i^k = \nabla^2 f_i(\theta^k)$, $g_i^k = \nabla f_i(\theta^k)$, and $\{y_i, \lambda_i\}_{i=1}^n$ denote the primal and dual variables for user i. The global model parameters at iteration k are θ^k. We refer to [16] for more details on the derivation of the FedNew method, and simply list here the resulting algorithm.

Let T denote the total number of training iterations. Also, denote the primal and dual variables of user i, $i \in [n]$, at iteration k, $k \in [T]$, as y_i^k and λ_i^k, respectively, and the global primal and dual variables at iteration k as y^k and λ^k. Then, the FedNew algorithm is described by the following steps.

(1) At user i, at round k, the update of the primal variable is obtained from the local minimization problem

$$\begin{aligned} y_i^k = \arg\min_y \Big[\tfrac{1}{2} y^T (H_i^k + \alpha I) y + \langle \lambda_i^{k-1}, y - y^{k-1} \rangle \\ - y^T g_i^k + \tfrac{\rho}{2} \|y - y^{k-1}\|_2^2 \Big] \end{aligned} \tag{3.1}$$

for which the solution can be written as

$$y_i^k = \big(H_i^k + (\alpha + \rho) I_d\big)^{-1} \big(g_i^k - \lambda_i^{k-1} + \rho y^{k-1}\big). \tag{3.2}$$

(2) The primal variable update at the server is obtained by solving the problem

$$y^k = \arg\min_y \Big[\sum_{i=1}^{n} \langle \lambda_i^{k-1}, y_i^k - y \rangle + \tfrac{\rho}{2} \sum_{i=1}^{n} \|y - y_i^k\|_2^2 \Big]$$

which gives the solution

$$y^k = \tfrac{1}{n} \sum_{i=1}^{n} (y_i^k + \tfrac{1}{\rho} \lambda_i^{k-1}). \tag{3.3}$$

(3) The dual variables are updated locally:

$$\lambda_i^k = \lambda_i^{k-1} + \rho (y_i^k - y^k). \tag{3.4}$$

Also, since

$$\sum_{i=1}^{n} \lambda_i^k = 0,$$

the update (3.3) can be written as an average of the primal variables:

$$y^k = \tfrac{1}{n} \sum_{i=1}^{n} y_i^k.$$

(4) The global model parameters are updated as

$$\theta^{k+1} = \theta^k - \eta \cdot y^k,$$

where $\eta > 0$ denotes the learning rate hyperparameter.

In case of convergence of the local iterations (repeating steps 1 to 3 until convergence), the following conditions are satisfied by the FedNew iteration for all $i \in [n]$ [see 16, and the references therein]:

$$\begin{aligned} & y_i^*(\theta^k) = y^*(\theta^k), \\ & (H_i^k + \alpha I) y_i^*(\theta^k) - g_i^k + \lambda_i^*(\theta^k) = 0, \end{aligned} \tag{3.5}$$

where $y_i^*(\theta^k)$ and $\lambda_i^*(\theta^k)$ denote the optimal values of y_i^k and λ_i^k, respectively, at iteration k, i.e., the results of running the ADMM steps until the end at the iteration k.

4 DP-FedNew

From the differential privacy perspective, the FedNew iteration poses several challenges. First of all, instead of only limiting the sensitivity of the gradients $\nabla f_i(\theta)$ by clipping and adding normally distributed noise as in DP-SGD [2], we need to consider the potential privacy leakage via the Hessians $\nabla^2 f_i(\theta)$ which are data-dependent. Second, naive differentially private FL approaches suffer from a very high communication cost due to the communication of the Hessians.

We next describe the required modifications to FedNew to make it differentially private. In FedNew, the only part where the data is used, is in the update of the primal variable y. At user i, at round k, the primal variable y_i^k is updated as

$$y_i^k = (H_i^k + \alpha I + \rho I)^{-1} (g_i^k - \lambda_i^{k-1} + \rho y^{k-1}). \tag{4.1}$$

Here the global primal variable y^{k-1} and the dual variable λ_i^{k-1} are results of previous iterations and therefore do not incur additional per-iteration privacy cost. The only data-dependent objects are the gradients $g_i^k = \nabla f_i(\theta^k)$ which are functions of data and the previous iterations primal variables $y^{k-1}, y^{k-2}, \ldots$.

We consider separately the user and record-level DP versions of DP-FedNew. In both cases, the only modification to the non-private FedNew algorithm happens in the update of local primal variables. We use the additive Gaussian noise, however, remark that our algorithm is compatible with any suitable DP secure aggregation primitive (e.g. [10, 22]) closed under summation and other noise distributions could be considered.

4.1 DP-FedNew with User-level Privacy

For user-level privacy, we need to hide users i whole contribution. We can obtain this simply by clipping the user-wise updates and adding normally distributed noise [similar user-level algorithms considered, e.g., in 30, 37]. This means that we simply replace the local update (3.2) in FedNew by the pseudocode of Algorithm 1, where we clip $\widehat{y}_i^k$ with some constant $C > 0$ and add normally distributed noise with covariance $C^2 \frac{\sigma^2 I_d}{n}$, $\sigma > 0$, to the resulting clipped update $\mathrm{clip}_C(\widehat{y}_i^k)$, where the clipping function is defined for vectors and matrices as

$$\mathrm{clip}_C(\widehat{y}_i^k) = \begin{cases} \widehat{y}_i^k, & \text{if} \quad \|\widehat{y}_i^k\|_F \le C, \\ C \cdot \frac{\widehat{y}_i^k}{\|\widehat{y}_i^k\|_F}, & \text{else.} \end{cases}$$

From the differentially privacy point of view, at each iteration we release only the noisy sum of the local updates,

$$\mathcal{M}(D) \sim \sum_{i=1}^{n} \left(\mathrm{clip}_C(\widehat{y}_i^k) + \mathcal{N}(0, \frac{C^2\sigma^2}{n} I_d)\right). \tag{4.2}$$

The noisy local primal variable update is distributed as,

$$\begin{aligned} y^k = \frac{1}{n}\sum_{i=1}^{n} \widetilde{y}_i^k &\sim \frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{clip}_C(\widehat{y}_i^k) + \mathcal{N}(0, \frac{C^2\sigma^2}{n} I_d)\right) \\ &\sim \frac{1}{n}\sum_{i=1}^{n} \mathrm{clip}_C(\widehat{y}_i^k) + \mathcal{N}(0, C^2\sigma^2 I_d). \end{aligned} \tag{4.3}$$

The local noise of scale $\frac{\sigma^2}{n}$ is not sufficient to derive the final privacy guarantee. The final privacy guarantee hinges on the underlying secure aggregation protocol, which privately sums the perturbed local primal variables. Note that this is a standard assumption in all protocols that combine secure aggregation and DP.

Algorithm 1 User-level DP-FedNew algorithm.

```
1:  Input: clipping constant $C > 0$, number of global steps $T$, noise
    scale $\sigma > 0$, regularizers $\rho > 0, \alpha > 0$, public server learning
    rate $\eta > 0$, initial parameter $\theta_0$.
2:  Initialize $\{\lambda_i^0\}_{i\in[n]}$ and $y^0$ to zero.
3:  for iteration $k = 1, \dots, T$ do
4:      for user $i = 1, \dots, n$ do
5:          Compute the non-DP update of the primal variable:
                $\widehat{y}_i^k = (H_i^k + (\alpha+\rho)I_d)^{-1}(g_i^k - \lambda_i^{k-1} + \rho y^{k-1})$.
6:          Clip and perturb the primal variable:
                $\widetilde{y}_i^k \leftarrow \mathrm{clip}_C(\widehat{y}_i^k) + E_i, \quad E_i \sim \mathcal{N}\left(0, \frac{C^2\sigma^2}{n} I_d\right)$.
7:          Share $\widetilde{y}_i^k$ with server.
8:      end for
9:      Server: $y^k = \frac{1}{n}\sum_{i=1}^{n} \widetilde{y}_i^k$.
10:     Server: $\theta^{k+1} = \theta^k - \eta \cdot y^k$ and share with users.
11:     for user $i = 1, \dots, n$ do
12:         Recover $y^k$ from $\theta^{k+1}, \theta^k$ and $\eta$.
13:         Update dual variable:
                $\lambda_i^k = \lambda_i^{k-1} + \rho(\widetilde{y}_i^k - y^k)$.
14:     end for
15: end for
```

4.2 DP-FedNew with Record-Level Privacy

In the record-level case we also have additive noise, and for the privacy guarantees we need to bound the sensitivity of y_i^k w.r.t. changes of data elements. Consider two neighbouring local datasets such that one of them has one more data element than the other one. Denote the Hessian of this additional data element x' by H' and it's gradient by g'.

Suppose the user i has the dataset D_i. Then, the data-dependent function that needs to be randomized is

$$F(D_i, \alpha, \rho, \lambda_i^{k-1}, y^{k-1}) = (H_i^k + \alpha I + \rho I)^{-1}(g_i^k - \lambda_i^{k-1} + \rho y^{k-1}).$$

Here α and ρ are pre-defined constant, and λ_i^{k-1} and y^{k-1} are auxiliary variables that are outputs of previous iterations. H_i^k stands for the Hessian and g_i^k for the gradient of user i.

As we are considering an additive-noise mechanism, we need to bound the sensitivity of the user-wise contributions w.r.t. to addition and removal of data elements. We have the following result which justifies our record-level clipping procedure described in Algorithm 2.

Lemma 3. *Let D_i' and D_i be neighboring datasets such that $D_i' = D_i \cup \{x'\}$ for some data-element x'. Let Δ_i be defined as*

$$\Delta_i := F(D_i', \alpha, \rho, \lambda_i^{k-1}, y^{k-1}) - F(D_i, \alpha, \rho, \lambda_i^{k-1}, y^{k-1}).$$

Assume

$$\begin{aligned} \|\nabla^2 f(x, \theta^k)\|_2 &\le \Delta_H \quad \text{for all } x \in D_i, \\ \|\nabla f(x, \theta^k)\|_2 &\le C_1 \quad \text{for all } x \in D_i, \\ \|g_i^k - \lambda_i^{k-1} + \rho y^{k-1}\| &\le C_2, \end{aligned} \tag{4.4}$$

and $\gamma = \alpha + \rho > \frac{\Delta_H}{|D_i|}$. Then, we have: $\|\Delta_i\|_2 \le \frac{1}{\gamma \cdot |D_i|} \cdot C_1 + \frac{\Delta_H}{\gamma^2 \cdot |D_i| - \gamma \cdot \Delta_H} \cdot C_2$, where $|D_i|$ is the size of the local dataset D_i.

The record-level DP precedure is described by Algorithm 2. Clipping the local gradient and Hessian may not be sufficient to bound the sensitivity of g_{sum} on line 6 due to the additive factor $-\lambda_i^{k-1} + \rho y^{k-1}$. Therefore, we find a scalar ξ to tightly bound $\|g_{sum}\|_2 \le C_2$ using the following analytical formula.

Lemma 4. *Let $a, b \in \mathbb{R}^n$ and $C > 0$. If we set*

$$\xi = \frac{-2\langle a, \frac{b}{\|b\|_2}\rangle + \sqrt{4\langle a, \frac{b}{\|b\|_2}\rangle^2 + 4(C^2 - \|a\|_2^2)}}{2\|b\|_2},$$

we have that $\|a + \xi \cdot b\|_2 = C$.

4.3 Memory Efficient Hessian Approximation

In each step, our algorithm requires each client to compute local Hessian with d^2 elements for each private sample. The computation and memory requirement of this order easily become prohibitive. For example, considering a logistic regression model with feature dimension $d_x = 512$ and output dimension $c = 10$, assuming 4 bytes is reserved for each parameter, then for a client with local 256 local records the per-example Hessian matrices would occupy in total $5120 \cdot 5120 \cdot 256 \cdot 4$ bytes equalling $\approx$ 27GBs of memory. Sequential Hessian computation is an option, but we could lose the benefit of vectorized computations offered by modern accelerators (e.g. GPUs) and incur an unacceptable increase in the run time. This motivates to search for appropriate approximations of the Hessian.

In our experiments and convergence analysis we focus on generalized linear models such as the logistic regression. This class of problems has recently turned out an attractive approach for private fine-tuning of large models pre-trained using public data [see, e.g., 15, 34]. Therein, a well-justified approximation of the Hessian using the covariance matrix of the feature vectors can be derived as follows [34]. Assume the loss function is of the form

$$f\big((x, y), \theta\big) = \ell(\theta^T x - y)$$

Algorithm 2 Record-level DP-FedNew algorithm.

```
1:  Input: clipping constants $\Delta_H, C_1, C_2 > 0$, number of global steps $T$, noise scale $\sigma > 0$, regularizers $\rho > 0, \alpha > 0$, public server learning rate $\eta > 0$, initial parameter $\theta_0$.
2:  Initialize $\{\lambda_i^0\}_{i \in [n]}$ and $y^0$ to zero.
3:  for iteration $k = 1, \dots, T$ do
4:      for user $i = 1, \dots, n$ do
5:          $H_i^k = \frac{1}{|D_i|} \sum_{x \in D_i} \mathrm{clip}_{\Delta_H}\left(\nabla^2 f(x, \theta^k)\right)$
            $g_i^k = \frac{1}{|D_i|} \sum_{x \in D_i} \mathrm{clip}_{C_1}\left(\nabla f(x, \theta^k)\right)$
6:          Scale the auxiliary variables with $C_2$:
            $g_{\mathrm{sum}} = g_i^k - \lambda_i^{k-1} + \rho y^{k-1}.$
7:          if $\|g_{\mathrm{sum}}\|_2 > C_2$ then
8:              $g_{\mathrm{sum}} = g_i^k + \xi \cdot (-\lambda_i^{k-1} + \rho y^{k-1}),$
                where the scalar $\xi$ is chosen using Lemma 4 such that
                $\|g_i^k + \xi \cdot (-\lambda_i^{k-1} + \rho y^{k-1})\|_2 \le C_2.$
9:          end if
10:         $\gamma = \alpha + \rho$
11:         Compute the non-DP update of the primal variable:
            $\widehat{y}_i^k = (H_i^k + \gamma I_d)^{-1} g_{\mathrm{sum}}.$
12:         Clip and perturb the primal variable:
            $\widetilde{y}_i^k \leftarrow \widehat{y}_i^k + E_i^k, E_i^k \sim \mathcal{N}(0, \frac{C^2\sigma^2}{n} I_d),$
            where
            $C = \frac{C_1}{\gamma \cdot |D_i|} + \frac{\Delta_H \cdot C_2}{\gamma^2 \cdot |D_i| - \gamma \cdot \Delta_H}.$
13:         Share $\widetilde{y}_i^k$ with server.
14:     end for
15:     Server: $y^k = \frac{1}{n} \sum_{i=1}^n \widetilde{y}_i^k$.
16:     Server: $\theta^{k+1} = \theta^k - \eta \cdot y^k$ and share with users.
17:     for user $i = 1, \dots, n$ do
18:         Recover $y^k$ from $\theta^{k+1}, \theta^k$, and $\eta$.
19:         Update dual variable: $\lambda_i^k = \lambda_i^{k-1} + \rho(\widetilde{y}_i^k - y^k)$.
20:     end for
21: end for
```

for some twice differentiable function ℓ, where $x \in \mathbb{R}^{d_x}$, $y \in \mathbb{R}^c$, $\theta \in \mathbb{R}^{d_x \times c}$. Then, for a $(d_x \cdot c)$-dimensional vectorized parameter vector, the Hessian is of the form

$$H((x,y),\theta) = \ell''(\theta^T x - y) \otimes xx^T,$$

where the Hessian $\ell''(\theta^T x - y)$ is a $c \times c$ matrix and where $\otimes$ denotes the Kronecker product. I.e., $H((x,y),\theta)$ is a $(d_x c \times d_x c)$ block matrix, where the ith $d_x \times d_x$ diagonal block equals $[\ell''(\theta^T x - y)]_{ii} xx^T$. In the method with a feature covariance approximation we simply carry out the approximation

$$H((x,y),\theta) \approx I_c \otimes xx^T$$

and then when the variables and gradients are expressed in matrix form, we can replace the scaled Hessian times the gradient product with the product

$$\left(\tfrac{1}{|D_i|} X_i X_i^T + \gamma I\right)^{-1} g_{\mathrm{sum}},$$

where then the gradient mean $g_{\mathrm{sum}} \in \mathbb{R}^{d_x \times c}$. This reduces both the compute and memory requirement considerably. In the above example, instead of memory requirement of 27GBs, we only need to allocate $256 \cdot 512 \cdot 512 \cdot 4$ bytes equalling ≈ 270MBs of memory for a parallel computation of the covariance matrix. Such proxy does not affect the privacy nor our convergence analysis and our experiments show that we still outperform the first order baseline. We call the resulting method DP FedNew Feature Covariance method (DP-FedNew-FC).

Remark 5. *Note that we assume for the privacy analysis the same scaling $1/|D_i|$ for the average per-example Hessians and gradients of user i both in case the averages originated from the datasets X and X'. Thus, strictly speaking, as we use the add/remove neighborhood relation of datasets, the privacy analysis holds in case we have some fixed scaling $1/|D_i|$ for every user $i \in [n]$ regardless of the input dataset. We could drop this assumption by considering the substitute neighborhood relation. This would simply double the sensitivity of the Gaussian mechanism used for the privacy analysis and increase the ε-values.*

5 Convergence Analysis

The analysis of FedNew as given in [16] is an asymptotic analysis such that the primal variables y_i^k get closer to the optimal primal variables $y_i^{k,*}$ which would be the result of running the inner loop until convergence. I.e., they show that

$$\lim_{k \to \infty} \|y_i^k - y^{k,*}\|_2^2 = 0. \quad (5.1)$$

With $y^{k,*}$'s the outer loop corresponds to a single step of Newton's iteration.

In case of DP-noise perturbed local updates, we cannot have the asymptotic convergence of the form (5.1). However, under the assumptions of the analysis by Elgabli et al. [16] and some weak assumptions related to the DP version of the algorithm, we obtain an asymptotic limit of the form

$$\lim_{k \to \infty} \|y^k - y^{k,*}\|_2^2 = O\left(\frac{d\sigma^2}{n}\right) = \widetilde{O}\left(\frac{d}{n\widetilde{\varepsilon}^2}\right),$$

where $y^k = \sum_{i=1}^n \widetilde{y}_i^k$ is a sum of the local noisy updates. and $\widetilde{\varepsilon}$ corresponds to the privacy guarantee of releasing the global result of a single iteration. The additional noise then matches the amount of local noise that one has, e.g., in DP gradient descent.

5.1 Convergence Analysis for DP-FedNew

For the convergence analysis of DP-FedNew, we assume that the clipping constants are chosen such that no clipping happens during the iteration. This is a natural assumption, e.g., in case the gradients are Lipschitz, the per-example Hessian is bounded in Frobenius norm (plausible assumption, e.g., for generalized linear models) and the auxiliary terms in the local update (4.1), i.e., the terms

$$g_i^k - \lambda_i^{k-1} + \rho y^{k-1},$$

ß $\in [n]$, stay bounded along the iteration. Then, the noisy update can be written as

$$\widetilde{y}_i^k = (H_i^k + \alpha I + \rho I)^{-1}(g_i^k - \lambda_i^{k-1} + \rho y^{k-1}) + E_i^k, \qquad (5.2)$$

where $E_i^k \sim \mathcal{N}(0, \frac{C^2\sigma^2}{n} I_d)$ and C is the sensitivity parameter, i.e., either the clipping constant in case of user level algorithm or then the parameter given by Lemma 3 in case of record-level algorithm. Without loss of generality of our results, we also assume that $C = 1$.

The following auxiliary result applies for the noisy update rule (5.2) and is central in our analysis.

Lemma 6. *Consider one iteration of DP-FedNew. Assume the per-example approximations of the Hessian at user i at iteration k, H_i^k, is positive semidefinite. Denote by $\lambda^{*,k}$ and $y^{k,*}$ the dual variables that are the results of running the non-DP FedNew inner iteration until the end (given the results of the DP iterations from $k-1$ iterations). Denote the dual residual $s^k := \rho(y^k - y^{k-1})$. We have:*

$$\mathbb{E}\langle \lambda_i^k + s^k - \lambda^{k,*}, \widetilde{y}_i^k - y^{k,*}\rangle \leq -\alpha \mathbb{E}\|y^{k,*} - \widetilde{y}_i^k\|^2 + \sigma^2 \cdot \mathrm{Trace}(H_i^k) + \sigma^2 \cdot (\alpha + \rho) \cdot d,$$

where the expectation is taken over the randomness of E_i^k, the noise added by the user i at iteration k.

In the following result, we define a certain Lyapunov function for the DP-FedNew algorithm, and by using the auxiliary Lemma 23, we obtain a stochastic inequality which leads us to the main result.

Lemma 7. *Let the Lyapunov function V_k be defined as*

$$V_k := \frac{1}{\rho}\sum_{i=1}^n \|\lambda_i^k - \lambda^{k,*}\|_2^2 + 2\beta_1 \sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|_2^2 + \rho n \|y^k - y^{k,*}\|_2^2 + 2\rho n \|y^k - y^{k-1}\|_2^2,$$

where $\widetilde{y}_i^k$ denotes the noisy update (5.2) *Denote $\widetilde{V}_k = \mathbb{E}V_k$, where the expectation is taken over all additive noises up to iteration k. Then, $\widetilde{V}_k$ satisfies*

$$\widetilde{V}_k \leq \widetilde{V}_{k-1} - \beta_2 \mathbb{E}\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|^2 + \sigma^2 \cdot (\alpha + \rho) \cdot d$$

for some constant $\beta_2 > 0$, where the expectation is taken over the noise added at iteration k.

Our main convergence result is of qualitative nature and states that the DP version inherits the stability of FedNew in a sense that the added DP noise does not make the solution to diverge from the non-private iterations. The result follows from Lemma 7.

Theorem 8. *Let $\sigma > 0$. For all $k \in \mathbb{Z}$, there exists $\ell > k$ such that*

$$\mathbb{E}\|y^\ell - y^{\ell,*}\|^2 \leq \frac{\sigma^2 \cdot (\alpha + \rho) \cdot d}{n\beta_2}$$

where the expectation is taken over the noise added at iteration ℓ.

6 Experimental Results

6.1 Experimental Setting and Baselines

Baselines. Noble et al. [36] proposed a DP variant of the Scaffold [24] method designed specifically to tackle data heterogeneity. Similarly to popular FedAVG [31], in Scaffold each client running performs multiple local updates before sending their model updates to the server. Each user and server maintains a control variable for drift correction. During each local step, subtraction of the local and global control variates is added to the perturbed gradients to counter local models drifting away from the global due to heterogeneity in their data distributions. Algorithms 3 and 4 of the Appendix outline record and user level full-batch versions of the DP-Scaffold methods. Experiments in Noble et al. [36] show that DP-Scaffold performs no worse than DP-FedAVG even on IID datasets. Moreover, we can recover DP-FedAVG by removing the control variables. Therefore, we use it as our main baseline (proxy for DP-FedAVG) in our evaluations on both IID and non-IID datasets. We use the *warm start* variant of DP-Scaffold in which the local control variates $\{c_i^0\}_{i=1}^n$ are initialized to the perturbed gradients in the first step. In each global iteration, DP-Scaffold requires clients and server to exchange the parameter and control variable information, making the communication cost $2 \times d$. Additionally, it only uses the first-order information for model update.

In the non-FL setting, Mehta et al. [34] proposed the DP-FC method which involves pre-multiplying the noisy full batch mean gradients with the inverse of a shifted noisy feature covariance matrix. We present federated versions of DP-FedFC in Algorithms 5 and 6. DP-FedFC gives a strong baseline however it requires server aggregating the *global* noisy feature covariance matrix from clients [1]. On the other hand, clients running DP-FedNew compute primal variables only with their local Hessians. Finally, we consider a distributed version of DP-GD (coined DP-FedGD) as another first-order baseline method because it has the same privacy and communication cost as DP-FedNew. Record- and user-level DP versions of the method are depicted in Algorithms 7 and 8 of the Appendix.

Differences to the Experimental Setting of [36]. DP-Scaffold is also applicable to non-convex problems and [36] consider training a 2-layer neural network from scratch for various classification experiments. We consider training only a single linear layer due to our focus on convex problems. For a fair comparison with DP-FedNew, we consider full batch variants for all baselines, i.e., all clients participate in each round. This means none of the methods benefit from privacy amplification due to client and record subsampling. Similar to DP-Scaffold, both DP-FedNew and DP-FedFC can be modified to perform multiple local client-side updates with client or record sampling. However, an exploration of these variants is left for future work.

IID Datasets. In our transfer learning experiments, we finetune the last layer on CIFAR10 [26], EMNIST [13], and FashionMnist [43]. Towards this purpose, we extract features of sizes 64 and 2048 from the last layers of pre-trained Resnet models listed in Table 2. Additionally, we train binary classification models from scratch on the enhanced Adult dataset from LIBSVM [6] repository.

Non-IID Datasets. We pick two non-IID classification datasets used in [36]. The first one is a synthetic dataset with feature dimension $d_x = 60$ drawn from a generative model proposed in [27] which allows us to adjust heterogeneity between local distributions

[1] Global noisy feature covariance matrix aggregation costs one or more communication rounds. In the user-level DP variant, server needs the second round to share the aggregated noisy covariance matrix back with the clients, making the cost $2d_x^2 + d \cdot T$. However, in record-level DP version, this second round can be saved by pushing the preconditioning to server.

Table 1: Dataset and model description.

datasets	CIFAR10	FashionMNIST	EMNIST	synthetic	Federated EMNIST	Adult
classes	10	10	10	10	47	2
N	50k	60k	240k	50k	90240	24800
$\|D_i\|$ (for Section 6.2 and 6.3)	$\frac{50k}{500} = 100$	$\frac{60k}{500} = 120$	$\frac{240k}{500} = 480$	$\frac{50k}{500} = 100$	$\frac{90240}{47} \approx 1920$	$\frac{24800}{500} \approx 50$
test dataset size	10k	10k	10k	10k	22560	6200
distribution	IID	IID	IID	non-IID	non-IID	IID
trained from scratch	No	No	No	Yes	Yes	Yes
layer sizes	64×10, 2048×10	64×10, 2048×10	64×10, 2048×10	60×5	100×47	123×2

Table 2: Method for extracting IID features.

Linear layer size	64×10	2048×10
Pretraining architecture	ResNet44	ResNet50
Pretraining weights	CIFAR100	Imagenet

Table 3: Our proposed methods DP-FedNew (Algorithm 1 and 3) and DP-FedNew-FC have both $O(d)$ communication and use second-order information about the loss functions. Neither of the baseline methods DP-Scaffold (Algorithm 3), DP-FedGD (Algorithm 7 and 8) and DP-FedFC (Algorithm 5 and 6) do not have both of these properties. Here d, d_x denote the dimension of the model and features. T is the number of training iterations.

method	comm. cost	uses 2nd order info.	performs local updates
DP-FedNew	$d \times T$	yes	no
DP-FedNew-FC	$d \times T$	yes	no
DP-FedFC	$d_x^2 + d \times T$	yes	no
DP-Scaffold	$2d \times T$	no	yes
DP-FedGD	$d \times T$	no	no

with hyperparameters $\alpha \geq 0$ and $\beta \geq 0$. Higher values of α, β indicate more heterogeneity. As in the experiments of [36], we fix $\alpha = 5$ and $\beta = 5$ which gives the most heterogeneous setting of [36]. The second dataset is the balanced version of EMNIST [12], with 47 classes (digits and letters). We consider the extreme scenario in which the training dataset is divided among 47 clients, with each client holding all records of exactly one class. Following [36], we call this dataset *Federated* EMNIST. With principal component analysis, we reduce the original dimensionality to 100. For experiments in Section 6.2 and 6.3, training data is divided among 500 clients such that each client has roughly an equal sized dataset. The dataset sizes are mentioned in Table 1.

Tasks. In experiments on IID data, we train the linear layers of size 64×10 and 2048×10. For reasons of space, all figures for experiments with layers of size 2048×10 are moved to the Appendix. We minimize the cross-entropy loss in all experiments. In each plot, we show the average test performance of the model obtained after performing a hyperparameter search for each method. Test accuracies are averaged across 5 independent runs. The best hyperparameters are selected based on the average test accuracy of the last 20 epochs. The hyperparameter grids used for the hyperparameter tuning are specified in Table 7.

Implementation. We generate the non-IID datasets using the code released with the DP-Scaffold paper [36]. The extreme distribution for the Federated EMNIST dataset can be obtained by setting the similarity hyperparameter to 0 in their script. We implement our training simulations with PyTorch 2.0 and rely on *vmap* calls for speeding up our per-example gradient and Hessian computations. For scalability, we tune the hyperparameters with Ray Tune [28] on a dedicated multi-GPU cluster.

Privacy Accounting. For each value of ε, we obtain the lowest σ through a binary search on the expression given by Lemma 2 in the Appendix. Due to M number of local updates, we account for $T \cdot M$ steps for DP-Scaffold instead of T steps. We do not consider the privacy cost of tuning towards the final DP guarantee for simplicity. We fix δ to $\frac{1}{N}$, where N is the total number of data points, in all experiments.

Remark 9. *As mentioned in Section 6.1, DP-FedFC is a strong baseline due to its higher communication complexity. For a fair evaluation and readability, we move the experiments comparing the epoch wise accuracy DP-FedFC to Section J.*

6.2 Experiments on IID Datasets

For Figure 1, we train linear layers of size 64×10 (image datasets) and 123×2 (Adult dataset) for each method, and compare the performance of DP-FedNew, DP-Scaffold, and DP-FedGD. We tune only the learning rate η for DP-FedGD, and additionally tune the number of local steps M for DP-Scaffold, constants Δ_H, α, and ρ for DP-FedNew. Figure 1 plots the mean test accuracy for the best model obtained after hyperparameter tuning for all methods at several epochs. Thanks to Hessian information, we observe that typically DP-FedNew converges the fastest for all datasets and for all ϵ levels. DP-Scaffold struggles specially at low ε regimes, due to much higher value of σ compared to DP-FedGD and DP-FedNew. In some cases however, it slightly outperforms DP-FedGD at higher ε values benefiting from the local steps.

To verify the impact of increased learning capacity and better features, Figure 5 in the Appendix shows the same quantities, but for layer sizes 2048×10. We cannot compute the per-example Hessians because of our computational limitations. Therefore, we use this figure as an opportunity to demonstrate the benefits of a variant where we carry out a Hessian approximation (DP-FedNew-FC) which is discussed in Section 4.3. The main observation across the board is that the overall accuracies at higher ε's are higher compared to Figure 1. In this case, the accuracy gap between DP-FedGD and DP-FedNew-FC shrinks, infact the curves overlap at lower ε's. For EMNIST and FashionMNIST, DP-FedNew provides similar final accuracies as DP-FedGD, but at much lower ε (e.g. $\varepsilon = 0.5$ vs. 8). DP-FedNew-FC still dominates the high utility regimes, and has the fastest convergence. In both figures, DP-Scaffold initially suffers from a *slow start* issue, and is generally outperformed by DP-FedGD at lower ε. Note that the slow start behavior has also been observed in [36] plots. Figure 3 shows comparisons for user-level DP, and it only support the previous observations. To comply user-DP, we clip and perturb the difference between previous global parameter θ^{k-1} and the final local model y_i^M, which is a standard [44] way to make FedAVG user-DP. Full exploration of other variants is outside the current scope.

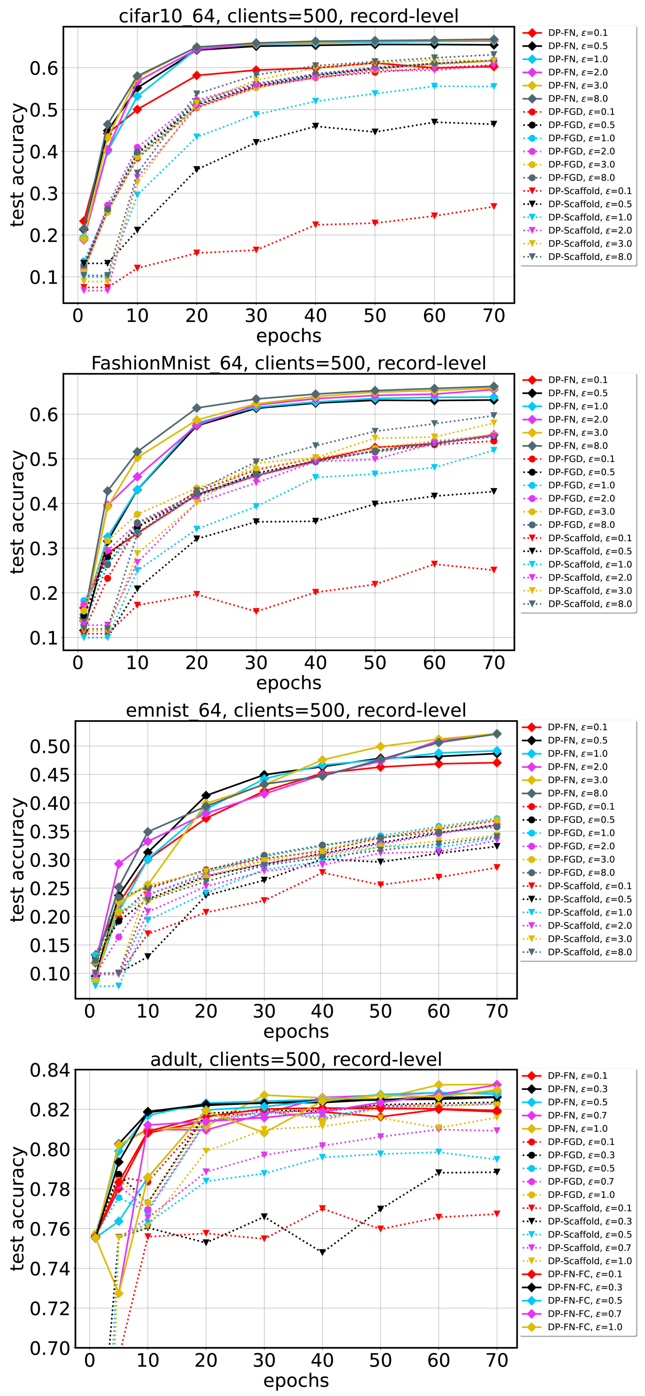
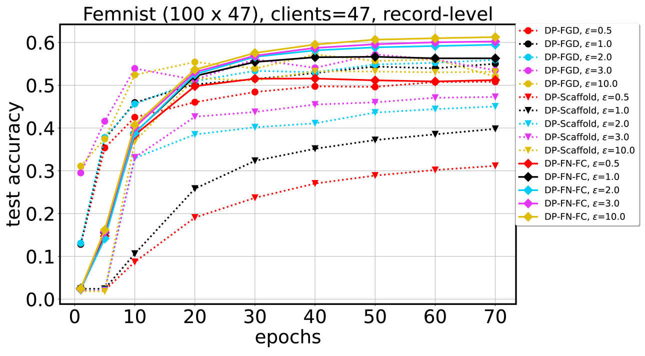


Figure 1: IID data: Record-level results for DP-FedNew (DP-FN in plots), DP-FedGD (DP-FGD in plots), and DP-Scaffold. For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size for Adult dataset is 123×2. For other datasets, it is 64×10. Check Figures 9 and 10 for comparison with DP-FedFC.

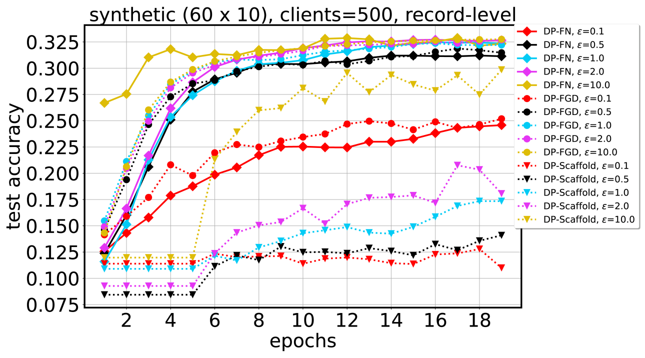
6.3 Non-IID Datasets

For experiments on non-IID data, we train the models from scratch. DP-Scaffold involves client performing multiple local updates. Similarly, DP-FedNew requires the inverse Hessian and gradient product for local primal variable computation. DP-Scaffold tries to correct the client model drifts with control variables. Although DP-FedNew has not been tailor made for heterogeneous datasets, we observe that the objective function 3.1 includes the term $||y - y^{k-1}||_2^2$ *by design*, owing to the use of augmented Lagrangian multipliers. This quadratic penalty aims to resist the deviations in local and global primal variables. We note that similar terms can also be found in other non-private first-order optimizers (e.g. FedProx [27]) designed for data-heterogeneous FL. Figures 2 and 6 (in Appendix) compare all three solutions on Federated EMNIST and synthetic datasets for record- and client-level DP. Note that in DP-FedGD, client's role is to only transmit the perturbed gradients to server. Computing gradients and taking their mean is a permutation invariant operation, and remains unaffected by changes in the client data distribution.

With the quadratic penalty and Hessian information, we can see in Figure 2a that DP-FedNew-FC performs at par with DP-FedGD for low ε's and outperforms it for higher ε even when each client



(a) Federated EMNIST



(b) Synthetic data

Figure 2: Non-IID data: Record-level results for DP-FedNew (DP-FN in plots) and DP-FedGD (DP-FGD in plots). For each ϵ, we plot the average test accuracies of the best model obtained after hyperparameter tuning. The model sizes are 100×47 and 60×10. Check Figure 6

has a dataset for a single label. We have similar results for the synthetic dataset in Figure 2, with DP-FedNew loosing to DP-FedGD by small margins for low ε's. However, curves nearly overlap for higher ε.

In Figure 6, we use $\varepsilon \geq 1$ to ensure the stability of all methods. In Figure 6a, DP-FedGD suffers from slow convergence, possibly caused by the bias injected during clipping and coarser nature of guarantee. To limit the contribution of each user, we also tried another variant in which per-example gradients are clipped, but did not observe a huge performance gain. We did not inpsect this issue further because neither DP-FedNew nor DP-Scaffold have an explicit bias-aware clipping mechanism. Bias correction is a topic of separate discussion. We note that DP-Fed(S)GD performs poorly on this dataset, even in [36] for non-convex problems. Despite twice the communication cost, DP-Scaffold continues to be inferior to DP-FedNew for all ε's except $\varepsilon = 10$, probably due to lack of second-order information. Similar remarks can be made about Figure 6b in Appendix.

6.4 Impact of Varying Local Dataset Sizes

We have fixed $|D_i| = 500$ in previous experiments. Now we would like to check the performance consistency of all methods across the client dataset sizes. We also compare with DP-FedFC in this Section. The size of D_i reduces with increase in n. However, the product $n \times |D_i|$ remains the same. The reduction in $|D_i|$ increases the sensitivity bound for DP-FedNew, which in turn increases the amount of noise to be injected into primal variables. Moreover, higher dissimilarity among local primal variables caused by smaller (and potentially imbalanced) datasets can also effectively increase the squared penalty term in the objective function 3.1, leading to reduced model performance. Similarly, for smaller datasets, client models after taking the local steps in DP-Scaffold are likely to experience higher drifts since each client's data is less likely to be a representation of the overall data distribution.

Tables 4 and 5 compare the mean accuracy at 70th epoch for the best model obtained after hyperparameter tuning for several ε's and different number of clients for IID FashionMNIST dataset and non-IID Federated EMNIST. This time, we also include DP-FedFC. For Federated EMNIST, each client holds data of atmost 2 classes. We tune the learning rate η and two clipping constants C_c, C_g for DP-FedFC. Section G in the Appendix includes tables for FashionMNIST, CIFAR10, Adult, and EMNIST datasets.

For DP-FedNew, DP-FedNew-FC, DP-FedFC, and DP-Scaffold, we observe the expected accuracy reduction as we increase n for $\varepsilon = 0.1$. DP-FedGD remains relatively unaffected by the variations in the dataset sizes, possibly because client's job is to only share the perturbed gradients with server, and the number of gradient evaluations stay the same. However, accuracies for DP-SCAFFOLD are still much worse than both DP-FedNew and DP-FedNew-FC even at $\varepsilon = 10$. The main conclusion for FashionMNIST is that DP-FedNew or DP-FedNew-FC generally are the most accurate methods for $\varepsilon < 0.7$, but get outperformed by DP-FedFC for $\varepsilon \geq 0.7$. For non-IID Federated EMNIST on the other hand, DP-FedNew-FC excels even for larger ε's. DP-FedFC's inferior run on non-IID data can be explained by the fact that heterogeneity induced in the data division or data generation process also changes the shape of the local covariance matrices. The sum of (noisy) local feature covariance matrices aggregated at server could differ a lot from the actual global covariance matrix. We compare user-level DP-FedNew and DP-FedFC in Appendix Figure 8. The summary is that DP-FedNew is overall the most accurate method for user-level DP.

DP-FedFC's higher accuracy for record-level DP comes at increased communication cost (under secure aggregation), which is $d_x^2 + d \cdot T$ for T training steps. At this communication cost, DP-FedNew (and DP-FedNew-FC) can perform $\frac{d_x \times d_x}{d} = \frac{d_x \times d_x}{d_x \times c} = \lceil \frac{d_x}{c} \rceil$ additional steps if required. The factor $\lceil \frac{d_x}{c} \rceil$ can be large when $d_x \gg c$. For example, while training a model of size 2048×10 (Tables 9, 10, and 11), the number of bits required to transmit a covariance matrix of size 2048×2048 is sufficient for DP-FedNew-FC to perform 205 global steps. We remind that DP-FedNew-FC only uses the local covariance matrices for primal variable computations.

6.5 Effect of Damping Local Hessians in DP-FedNew

Local primal variable computation requires adding a damping factor $\gamma = \alpha + \rho$ to each Hessian H_i in Equation 3.2. Our sensitivity result in Lemma 3 is also conditioned on the assumption $\gamma > \frac{\Delta_H}{|D_i|}$.

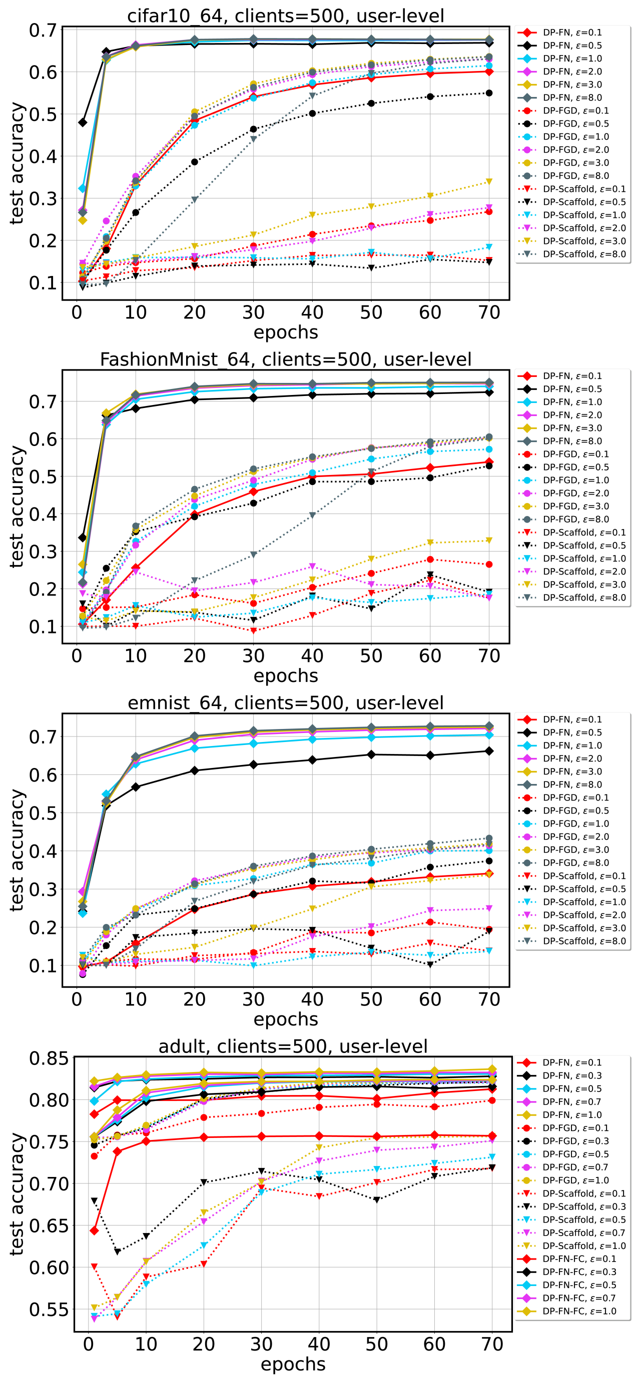


Figure 3: IID data: User-level results for DP-FedNew (DP-FN in plots), DP-Scaffold, and DP-FedGD (DP-FGD in plots). For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size for Adult dataset is 123×2. For other datasets, it is 64×10. Check Figures 7 and 8 for comparison with DP-FedFC.

Specifically, we use the fact that adding $\gamma > 0$ to a positive semidefinite matrix H_i increases its eigen values by at least γ. The value of regularizer α is typically small, however ρ, which controls the influence of primal/dual variables from previous iteration in the current one can be quite high. If ρ is much larger compared to the largest eigen value of clipped H_i^k, the factor γ dictates the magnitudes of the spectrum of $H_i^k + (\alpha + \rho)I_d$. This effect can deteriorate the second-order information present in the clipped and damped Hessian. To be able to attribute DP-FedNew's strong experimental success to the Hessian information, the condition $\gamma > \frac{\Delta_H}{|D_i|}$ should be satisfied even for moderately low values of ρ.

Table 6 assesses how DP-FedNew responds to changes in ρ and γ for various dataset sizes for $\varepsilon = 1$ for FashionMNIST. For several n's we show the accuracies of the top-3 most accurate models at various epochs along with their mean maximum eigenvalues. The columns for each n are sorted according to the accuracy in the 70th epoch. For FashionMNIST, we observe that the most frequent ρ value is 1, and $\alpha < 1$. Table 12 in Appendix reports the same quantities for EMNIST. For EMNIST, the best ρ is 1, and best α's are once again less than 1. This supports the hypothesis that DP-FedNew is not too sensitive to the choice of ρ, and the prerequisite $\gamma > \frac{\Delta_H}{|D_i|}$ can be satisfied easily.

6.6 Computation vs. Communication Cost

Figure 4 compares the mean total time taken for all client and server-side computations when a training global model for 70 epochs. Note that DP-Scaffold's runtimes vary a lot across ε's because we also tune the number of local steps. Due to costly Hessian computations, DP-FedNew's run time scales up more quickly than the first-order alternatives with the local dataset size. Therefore, for stress testing, we divide IID datasets among only 50 clients compared to 500 earlier so that each client has 10X more records. While DP-FedNew is computationally the most expensive method, we check in the first two plots that it achieves the final accuracy of DP-FedGD and DP-Scaffold in less than $\frac{1}{4}$th of its total run time, and before the 15th epoch. This means that we can save both privacy budget and communication by running DP-FedNew for a much lower number of epochs. The highlight of Figure 4 is the DP-FedNew-FC method, which attains accuracy comparable to DP-FedNew within a small fraction of its run time, making it a viable alternative to DP-FedNew when Hessians computations are infeasible. Each user in this case can pre-compute and cache the local covariance matrix.

7 Concluding Remarks

We have proposed the first DP FL optimization method with model-sized communication overhead that uses the curvature information via the feature covariance matrix or via the Hessian matrix of the loss function. Our approach nicely complements an orthogonal line of research dedicated to the development of resource efficient DP primitives for secure aggregation.

For convex classification problems, our experiments show that we can achieve better privacy-utility tradeoffs by incorporating second-order information into optimizer than the first order counterparts. DP-FedNewton outperforms the first-order alternatives nearly all ε's on iid datasets; and specially in high accuracy regimes even for challenging non-iid data distributions due to inbuilt drift resistance. While DP-FedNew is computationally more expensive, we can save on both communication and privacy budget compared to the competition by running it for much lower number of epochs. When Hessians cannot be computed, their computation and memory wise cheaper approximation can also exhibit better privacy-utility tradeoffs than the alternatives. DP-FedNew also appears to be the least fluctuating method, even for low ε's.

The main limitation of DP-fedNew is the memory cost required for holding the per-example Hessians, which arises due to our assumption of the bounded matrix norm in the sensitivity analysis. It may be possible to relax this assumption by appropriately clipping the features. We can then use PyTorch/Jax utilities (e.g., [1, 40]) to efficiently compute inverse Hessian gradient products without explicitly computing the Hessians or their inverses.

Ability to sample clients and records is another practical functionality that DP-Fednew is missing. Here various subsampling amplification results could be considered to lower per iteration privacy cost. Another interesting avenue of future work is to check if the proposed method can be extended for non-convex tasks, such as training a neural network with multiple layers.

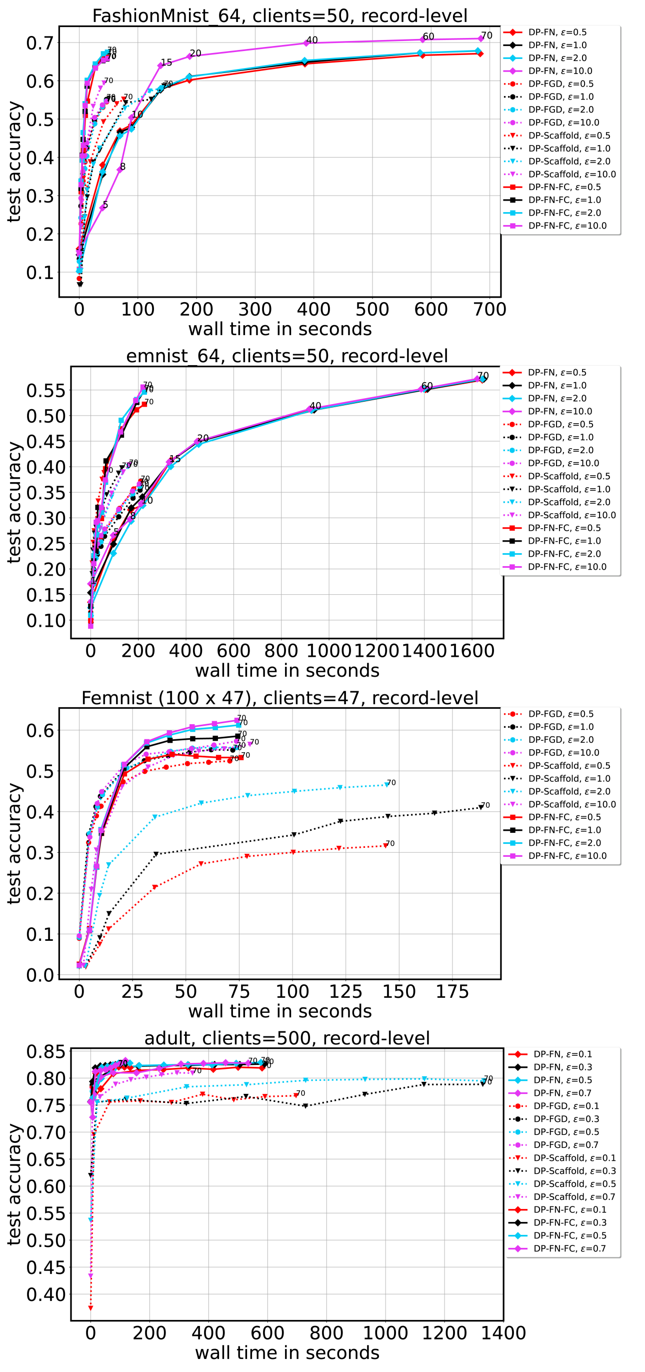


Figure 4: Timing plots: Record-level results for DP-FedNew (DP-FN in plots), DP-FedNew-FC (DP-FN-FC in plots), DP-FedGD (DP-FGD in plots), and DP-Scaffold. The X axis shows the mean computational time required for training a global model for 70 epochs. For readability, only some of the points are annotated with the epoch number. For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning.

Table 4: Effect of varying the local dataset size. The numbers in the ε columns report the mean test accuracy after the final 70th epoch for the best model trained on IID FashionMNIST dataset obtained after tuning the hyperparameters mentioned in Table 7. The layer sizes are 64×10. Note that DP-FedFC achieves higher accuracy at the cost of higher communication.

n	$\lvert D_i \rvert$	method	$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 8$	$\epsilon = 10$
20	3000	DP-FedNew	0.638	0.676	0.683	0.684	0.686	0.690	0.692	0.710	0.713
		DP-FedNew-FC	0.643	0.678	0.676	0.683	0.683	0.687	0.683	0.685	0.688
		DP-Scaffold	0.497	0.519	0.471	0.541	0.592	0.531	0.527	0.515	0.510
		DP-FedFC	0.645	0.672	0.675	0.704	0.715	0.724	0.724	0.736	0.739
		DP-FedGD	0.555	0.565	0.569	0.575	0.568	0.563	0.567	0.566	0.568
50	1200	DP-FedNew	0.643	0.679	0.683	0.687	0.689	0.689	0.690	0.708	0.714
		DP-FedNew-FC	0.642	0.670	0.684	0.684	0.682	0.686	0.685	0.685	0.685
		DP-Scaffold	0.427	0.543	0.570	0.586	0.601	0.603	0.611	0.611	0.611
		DP-FedFC	0.635	0.668	0.671	0.704	0.717	0.724	0.725	0.739	0.739
		DP-FedGD	0.561	0.571	0.572	0.571	0.572	0.575	0.571	0.569	0.568
100	600	DP-FedNew	0.608	0.675	0.686	0.686	0.687	0.691	0.691	0.710	0.713
		DP-FedNew-FC	0.604	0.673	0.678	0.684	0.687	0.681	0.684	0.683	0.687
		DP-Scaffold	0.346	0.494	0.549	0.575	0.595	0.605	0.607	0.615	0.615
		DP-FedFC	0.638	0.667	0.672	0.704	0.716	0.723	0.725	0.739	0.738
		DP-FedGD	0.561	0.569	0.574	0.578	0.578	0.569	0.577	0.572	0.569
250	240	DP-FedNew	0.601	0.679	0.683	0.687	0.687	0.690	0.691	0.708	0.716
		DP-FedNew-FC	0.583	0.678	0.681	0.681	0.686	0.682	0.684	0.684	0.685
		DP-Scaffold	0.340	0.452	0.451	0.511	0.483	0.493	0.482	0.513	0.492
		DP-FedFC	0.641	0.667	0.675	0.705	0.717	0.723	0.725	0.735	0.739
		DP-FedGD	0.574	0.568	0.577	0.575	0.576	0.572	0.568	0.569	0.569
500	120	DP-FedNew	0.575	0.654	0.682	0.688	0.688	0.690	0.691	0.708	0.713
		DP-FedNew-FC	0.574	0.651	0.681	0.687	0.687	0.686	0.687	0.683	0.688
		DP-Scaffold	0.331	0.418	0.464	0.470	0.531	0.585	0.597	0.607	0.606
		DP-FedFC	0.634	0.667	0.672	0.705	0.716	0.723	0.725	0.739	0.740
		DP-FedGD	0.560	0.571	0.572	0.569	0.568	0.578	0.579	0.574	0.575

Table 5: Effect of varying the local dataset size. The numbers in the ε columns report the mean test accuracy after the final 70th epoch for the best model trained on non-IID Federated EMNIST dataset obtained after tuning the hyperparameters mentioned in Table 7. The layer sizes are 100×47. Note that DP-FedFC achieves higher accuracy at the cost of higher communication.

			$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 8$	$\epsilon = 10$
n	$\lvert D_i \rvert$	method									
47	1920	DP-FedNew-FC	$\underline{0.390}$	$\underline{0.489}$	$\underline{0.527}$	$\underline{0.556}$	$\underline{0.582}$	$\underline{0.613}$	$\underline{0.622}$	$\underline{0.628}$	$\underline{0.629}$
		DP-Scaffold	0.135	0.270	0.329	0.375	0.416	0.483	0.512	0.552	0.558
		DP-FedFC	$\underline{0.390}$	$\underline{0.489}$	$\underline{0.527}$	0.550	0.569	0.545	0.567	0.574	0.595
		DP-FedGD	0.389	0.488	0.524	0.545	0.533	0.565	0.564	0.576	0.569
100	893	DP-FedNew-FC	$\underline{0.383}$	0.481	$\underline{0.548}$	$\underline{0.552}$	$\underline{0.584}$	$\underline{0.610}$	$\underline{0.619}$	$\underline{0.626}$	$\underline{0.627}$
		DP-Scaffold	0.085	0.217	0.281	0.322	0.361	0.439	0.469	0.493	0.492
		DP-FedFC	0.382	$\underline{0.491}$	0.521	0.548	0.565	0.560	0.559	0.585	0.580
		DP-FedGD	$\underline{0.383}$	0.482	0.519	0.541	0.558	0.562	0.562	0.574	0.564
250	359	DP-FedNew-FC	0.376	$\underline{0.491}$	$\underline{0.533}$	$\underline{0.557}$	$\underline{0.582}$	$\underline{0.614}$	$\underline{0.621}$	$\underline{0.628}$	$\underline{0.631}$
		DP-Scaffold	0.067	0.148	0.207	0.256	0.293	0.347	0.365	0.372	0.370
		DP-FedFC	0.384	0.481	0.526	0.555	0.566	0.577	0.560	0.550	0.587
		DP-FedGD	$\underline{0.385}$	0.484	0.525	0.545	0.553	0.557	0.550	0.548	0.544
500	179	DP-FedNew-FC	0.384	0.462	$\underline{0.539}$	$\underline{0.568}$	$\underline{0.593}$	$\underline{0.620}$	$\underline{0.631}$	$\underline{0.639}$	0.649
		DP-Scaffold	0.053	0.122	0.157	0.192	0.205	0.234	0.243	0.253	0.249
		DP-FedFC	$\underline{0.401}$	0.495	$\underline{0.539}$	0.561	0.575	0.584	0.587	0.550	$\underline{0.669}$
		DP-FedGD	0.391	$\underline{0.498}$	0.534	0.547	0.555	0.573	0.558	0.609	0.558
1000	89	DP-FedNew-FC	0.377	0.456	0.497	0.542	$\underline{0.584}$	$\underline{0.615}$	$\underline{0.626}$	$\underline{0.631}$	$\underline{0.633}$
		DP-Scaffold	0.044	0.078	0.097	0.107	0.115	0.120	0.124	0.126	0.128
		DP-FedFC	$\underline{0.390}$	0.485	0.526	$\underline{0.553}$	0.571	0.573	0.573	0.542	0.570
		DP-FedGD	0.389	$\underline{0.487}$	$\underline{0.530}$	0.548	0.555	0.581	0.596	0.573	0.566

Table 6: Effect of ρ and α while varying the local dataset sizes. The numbers in the columns $\{1, 10, 30, 50, 70\}$ report the mean test accuracies in the correspond epochs for the top-3 most accurate models trained on IID FashionMNIST dataset for $\varepsilon = 1$. The layer size is 64×10. The value in the columns $\bar{\lambda}^{j}_{max}$ is computed by averging the maximum eigen value of local Hessians from all clients at epoch j. We tune the hyperparameters $\alpha = \{0.01, 0.1, 1\}$, and $\rho = \{0.01, 0.1, 1, 10, 100\}$. $\eta = C_1 = C_2 = \Delta_H = 1$.

					1	10	30	50	70	$\bar{\lambda}^{1}_{max}$	$\bar{\lambda}^{10}_{max}$	$\bar{\lambda}^{30}_{max}$	$\bar{\lambda}^{50}_{max}$	$\bar{\lambda}^{70}_{max}$
n	$\lvert D_i \rvert$	η	ρ	α										
20	3000	1	1	0.01	0.138	0.484	0.632	0.665	0.680	0.584	0.304	0.206	0.176	0.172
				0.10	0.104	0.446	0.621	0.649	0.664	0.663	0.323	0.225	0.215	0.209
			0.1	0.10	0.118	0.511	0.631	0.649	0.657	0.605	0.406	0.220	0.219	0.221
50	1200	1	1	0.01	0.162	0.518	0.634	0.663	0.679	0.345	0.290	0.207	0.174	0.173
				0.10	0.193	0.496	0.627	0.650	0.663	0.345	0.286	0.227	0.215	0.211
			0.1	0.10	0.234	0.502	0.624	0.644	0.659	0.676	0.381	0.237	0.227	0.214
100	600	1	1	0.01	0.137	0.477	0.633	0.665	0.679	0.632	0.301	0.210	0.179	0.174
				0.10	0.162	0.473	0.620	0.645	0.660	0.599	0.306	0.229	0.218	0.213
			0.1	0.10	0.115	0.535	0.615	0.645	0.652	0.595	0.330	0.235	0.227	0.227
500	120	1	1	0.01	0.116	0.463	0.633	0.663	0.678	0.621	0.321	0.227	0.195	0.186
				0.10	0.093	0.502	0.625	0.650	0.665	0.568	0.318	0.248	0.236	0.231
			0.1	0.10	0.148	0.530	0.626	0.650	0.657	0.504	0.308	0.250	0.242	0.235

Acknowledgments

We thank the anonymous reviewers for their constructive comments and suggestions. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

[1] Alexander B. A. and Matthew Johnson. 2024. The Autodiff Cookbook. Accessed on 2024-04-17.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 308–318.

[3] Borja Balle and Yu-Xiang Wang. 2018. Improving the Gaussian Mechanism for Differential Privacy: Analytical Calibration and Optimal Denoising. In *International Conference on Machine Learning*. 394–403.

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

[5] Raef Bassily, Adam Smith, and Abhradeep Thakurta. 2014. Private Empirical Risk Minimization: Efficient Algorithms and Tight Error Bounds. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. 464–473.

[6] Barry Becker and Ronny Kohavi. 1996. Adult Dataset. UCI Machine Learning Repository.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.* 3, 1 (2011), 122 pages.

[8] Wei-Ning Chen, Christopher A. Choquette-Choo, Peter Kairouz, and Ananda Theertha Suresh. 2022. The Fundamental Price of Secure Aggregation in Differentially Private Federated Learning. In *International Conference on Machine Learning, ICML 2022 (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 3056–3089.

[9] Wei-Ning Chen, Peter Kairouz, and Ayfer Özgür. 2023. Breaking the Communication-Privacy-Accuracy Trilemma. *IEEE Trans. Inf. Theory* 69, 2 (2023), 1261–1281.

[10] Wei-Ning Chen, Ayfer Özgür, and Peter Kairouz. 2022. The Poisson Binomial Mechanism for Unbiased Federated Learning with Secure Aggregation. In *International Conference on Machine Learning, ICML 2022 (Proceedings of Machine Learning Research, Vol. 162)*. PMLR, 3490–3506.

[11] Wei-Ning Chen, Dan Song, Ayfer Özgür, and Peter Kairouz. 2023. Privacy Amplification via Compression: Achieving the Optimal Privacy-Accuracy-Communication Trade-off in Distributed Mean Estimation. *Advances in Neural Information Processing Systems* 36 (2023).

[12] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. EMNIST: an extension of MNIST to handwritten letters.

[13] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*. 2921–2926.

[14] Edwige Cyffers, Aurelien Bellet, and Debabrota Basu. 2023. From noisy fixed-point iterations to private ADMM for centralized and federated learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*.

[15] Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. 2022. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650* (2022).

[16] Anis Elgabli, Chaouki Ben Issaid, Amrit Singh Bedi, Ketan Rajawat, Mehdi Bennis, and Vaneet Aggarwal. 2022. FedNew: A communication-efficient and privacy-preserving Newton-type method for federated learning. In *International Conference on Machine Learning*. PMLR, 5861–5877.

[17] Arun Ganesh, Mahdi Haghifam, Thomas Steinke, and Abhradeep Thakurta. 2023. Faster Differentially Private Convex Optimization via Second-Order Methods. *arXiv preprint arXiv:2305.13209* (2023).

[18] Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. 2021. Numerical Composition of Differential Privacy. In *Advances in Neural Information Processing Systems*.

[19] Florian Hartmann and Peter Kairouz. 2023. Distributed Differential Privacy for Large-Scale Data Analysis. Blog.

[20] Roger A Horn and Charles R Johnson. 2012. *Matrix analysis*. Cambridge university press.

[21] Zonghao Huang, Rui Hu, Yuanxiong Guo, Eric Chan-Tin, and Yanmin Gong. 2020. DP-ADMM: ADMM-Based Distributed Learning With Differential Privacy. *Trans. Info. For. Sec.* (jan 2020), 1002–1012.

[22] Peter Kairouz, Ziyu Liu, and Thomas Steinke. 2021. The Distributed Discrete Gaussian Mechanism for Federated Learning with Secure Aggregation. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021 (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 5201–5212.

[23] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.

[24] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *Proceedings of the 37th International Conference on Machine Learning*.

[25] Antti Koskela, Joonas Jälkö, Lukas Prediger, and Antti Honkela. 2021. Tight Differential Privacy for Discrete-Valued Mechanisms and for the Subsampled Gaussian Mechanism Using FFT. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 3358–3366.

[26] Alex Krizhevsky and Geoffrey Hinton. 2009. Learning Multiple Layers of Features from Tiny Images. *University of Toronto* (2009).

[27] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. In *Proceedings of Machine Learning and Systems 2020, MLSys 2020*. mlsys.org.

[28] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. 2018. Tune: A Research Platform for Distributed Model Selection and Training. *arXiv preprint arXiv:1807.05118* (2018).

[29] Yuanyuan Liu, Jiacheng Geng, Fanhua Shang, Weixin An, Hongying Liu, Qi Zhu, and Wei Feng. 2022. Laplacian Smoothing Stochastic ADMMs With Differential Privacy Guarantees. *IEEE Trans. Inf. Forensics Secur.* (2022).

[30] Brendan McMahan, Galen Andrew, Ilya Mironov, Nicolas Papernot, Peter Kairouz, Steve Chien, and Úlfar Erlingsson. 2018. A General Approach to Adding Differential Privacy to Iterative Training Procedures. *NeurIPS 2018 workshop on Privacy Preserving Machine Learning (PPML)* (2018).

[31] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS (Proceedings of Machine Learning Research)*.

[32] H. Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. 2018. A General Approach to Adding Differential Privacy to Iterative Training Procedures. *CoRR* abs/1812.06210 (2018).

[33] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. In *6th International Conference on Learning Representations, ICLR 2018,*.

[34] Harsh Mehta, Walid Krichene, Abhradeep Guha Thakurta, Alexey Kurakin, and Ashok Cutkosky. 2023. Differentially Private Image Classification from Features. *Transactions on Machine Learning Research* (2023).

[35] Ilya Mironov. 2017. Rényi Differential Privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. 263–275.

[36] Maxence Noble, Aurélien Bellet, and Dieuleveut Aymeric. 2022. Differentially Private Federated Learning on Heterogeneous Data. In *25th International Conference on Artificial Intelligence and Statistics*. PMLR.

[37] Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Guha Thakurta. 2023. How to dp-fy ml: A practical guide to machine learning with differential privacy. *Journal of Artificial Intelligence Research* 77 (2023), 1113–1201.

[38] Minseok Ryu and Kibaek Kim. 2022. Differentially Private Federated Learning via Inexact ADMM with Multiple Local Updates. *CoRR* abs/2202.09409 (2022).

[39] David M Sommer, Sebastian Meiser, and Esfandiar Mohammadi. 2019. Privacy loss classes: The central limit theorem in differential privacy. *Proceedings on Privacy Enhancing Technologies* 2019, 2 (2019), 245–269.

[40] PyTorch Team. 2024. torch.autograd.functional.hvp.

[41] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. 2019. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on artificial intelligence and security*. 1–11.

[42] Enayat Ullah, Christopher A. Choquette-Choo, Peter Kairouz, and Sewoong Oh. 2023. Private Federated Learning with Autotuned Compression. In *International Conference on Machine Learning, ICML 2023 (Proceedings of Machine Learning Research, Vol. 202)*. PMLR, 34668–34708.

[43] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. Technical Report.

[44] Xinwei Zhang, Xiangyi Chen, Min-Fong Hong, Zhiwei Steven Wu, and Jinfeng Yi. 2021. Understanding Clipping for Federated Learning: Convergence and Client-Level Differential Privacy. In *International Conference on Machine Learning*.

[45] Yuqing Zhu, Jinshuo Dong, and Yu-Xiang Wang. 2022. Optimal Accounting of Differential Privacy via Characteristic Function. *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics* (2022).

A Detailed Description of DP-Scaffold, DP-FedFC, and DP-FedGD Algorithms

Algorithm 3 Record-level full-batch DP-Scaffold (based on the DP-Scaffold method of [36])

```
1:  Input: dataset $D = \{D_i\}_{i=1}^n$, noise level $\sigma$, clipping constant $C$, training length $T$, number of local steps $M$, local and global learning rates $\eta_l$ and $\eta_g$, initial $c_i^0$.
2:  for iteration $k = 1, \dots, T$ do
3:      Server: Send $(\theta^{k-1}, c^{k-1})$ to all users.
4:      for user $i = 1, \dots, n$ do
5:          Initialize model: $y_i^0 = \theta^{k-1}$.
6:          for local step $m = 1, \dots, M$ do
7:              Clients: Add DP noise to local gradients:
                $\tilde{g}_i^m = \frac{1}{|D_i|} \sum_{x \in D_i} \mathrm{clip}_C\big(\nabla f(x, \theta^m)\big) + \frac{2C}{|D_i|} \mathcal{N}(0, \sigma^2)$
8:              $y_i^m = y_i^{m-1} - \eta_l(\tilde{g}_i^m - c_i^{k-1} + c^{k-1})$
9:          end for
10:         Update user control variables:
            $\tilde{c}_i^k = c_i^{k-1} - c^{k-1} + \frac{(\theta^{k-1} - y_i^M)}{M\eta_l}$
11:         $(\Delta y_i^k, \Delta c_i^k) = (y_i^M - \theta^{k-1}, \tilde{c}_i^k - c_i^{k-1})$
12:         Share $(\Delta y_i^k, \Delta c_i^k)$ with server.
            $c_i^k = \tilde{c}_i^k$
13:     end for
14:     Server: $(\Delta\theta^k, \Delta c^k) = \frac{1}{n} \sum_{i=1}^n (\Delta y_i^k, \Delta c_i^k)$
15:     Server: Global model update, $\theta^k = \theta^{k-1} + \eta_g \Delta\theta^k$.
16:     Server: Update global control variable, $c^k = c^{k-1} + \Delta c^k$.
17: end for
```

Algorithm 4 User-level full-batch DP-Scaffold (based on the DP-Scaffold method of [36])

```
1:  Input: dataset $D = \{D_i\}_{i=1}^n$, noise level $\sigma$, clipping constant $C$, training length $T$, number of local steps $M$, local and global learning rates $\eta_l$ and $\eta_g$, initial $c_i^0$.
2:  for iteration $k = 1, \dots, T$ do
3:      Server: Send $(\theta^{k-1}, c^{k-1})$ to all users.
4:      for user $i = 1, \dots, n$ do
5:          Initialize model: $y_i^0 = \theta^{k-1}$.
6:          for local step $m = 1, \dots, M$ do
7:              Clients: Add DP noise to local gradients:
                $g_i^m = \frac{1}{|D_i|} \sum_{x \in D_i} \nabla f(x, \theta^m)$
8:              $y_i^m = y_i^{m-1} - \eta_l(g_i^m - c_i^{k-1} + c^{k-1})$
9:          end for
10:         $\tilde{\zeta}_i = \mathrm{clip}_C(\theta^{k-1} - y_i^M) + \mathcal{N}(0, \frac{C^2\sigma^2 I_d}{n})$
11:         Update user control variables:
            $\tilde{c}_i^k = c_i^{k-1} - c^{k-1} + \frac{\tilde{\zeta}_i}{M\eta_l}$
12:         $(\Delta y_i^k, \Delta c_i^k) = (-\tilde{\zeta}_i, \tilde{c}_i^k - c_i^{k-1})$
13:         Share $(\Delta y_i^k, \Delta c_i^k)$ with server.
            $c_i^k = \tilde{c}_i^k$
14:     end for
15:     Server: $(\Delta\theta^k, \Delta c^k) = \frac{1}{n} \sum_{i=1}^n (\Delta y_i^k, \Delta c_i^k)$
16:     Server: Global model update, $\theta^k = \theta^{k-1} + \eta_g \Delta\theta^k$.
17:     Server: Update global control variable, $c^k = c^{k-1} + \Delta c^k$.
18: end for
```

Algorithm 5 Record-level DP-FedFC Algorithm (based on the DP-FC method of [34])

```
1: Input: dataset $D = \{D_i\}_{i=1}^n$, noise levels $\sigma_c, \sigma_g$, clipping constants $C_c, C_g$, training length $T$, learning rate $\eta$, regularization parameter $\gamma$.
2: for user $i = 1, \dots, n$ do
3:     Clip local user inputs: $\widetilde{D}_i = \left[\text{clip}_{C_c}(x_1) \quad \dots \quad \text{clip}_{C_c}(x_{|D_i|})\right]$.
4:     Compute local noisy feature covariance matrix: $C_i = \widetilde{D}_i \widetilde{D}_i^T + E_i, E_i \sim \mathcal{N}\left(0, \frac{I_d (C_c^2 \sigma_c)^2}{n}\right)$
5:     Share $C_i$ with server.
6: end for
7: Server aggregates $\{C_i\}_{i=1}^n$, computes the global noisy convariance matrix $C = \frac{\sum_i^n C_i}{n} + \gamma I_d$.
8: for iteration $k = 1, \dots, T$ do
9:     for user $i = 1, \dots, n$ do
10:        Clip and perturb local gradients:
           $$u_i^k = \frac{\sum_{x \in D_i} \text{clip}_{C_g}\left(\nabla f(\theta^k, x)\right) + E_i^k}{|D_i|}, E_i^k \sim \mathcal{N}\left(0, \frac{I_d (C_g \sigma_g)^2}{n}\right).$$
11:        Share local update $u_i^k$ with server.
12:    end for
13:    Server aggregates $\{u_i^k\}_{i=1}^n$, computes the global noisy update $U^k = C^{-1} \cdot \left(\frac{\sum_{i=1}^n u_i^k}{n}\right)$.
14:    Update model parameters and share with users:
           $$\theta^{k+1} = \theta^k - \eta \cdot U^k.$$
15: end for
```

Algorithm 6 User-level DP-FedFC (based on the DP-FC method of [34])

```
1: Input: dataset $D = \{D_i\}_{i=1}^n$, noise levels $\sigma_c, \sigma_g$, clipping constants $C_c, C_g$, training length $T$, learning rate $\eta$, regularization parameter $\gamma$.
2: for user $i = 1, \dots, n$ do
3:     Clip local covariance matrices: $\tilde{C}_i = \text{clip}_{C_c}(D_i . D_i^T)$.
4:     Compute local noisy feature covariance matrix: $C_i = \tilde{C}_i + E_i, E_i \sim \mathcal{N}\left(0, \frac{I_d (C_c \sigma_c)^2}{n}\right)$
5:     Share $C_i$ with server.
6: end for
7: Server aggregates $\{C_i\}_{i=1}^n$, computes global noisy convariance matrix $C = \frac{\sum_i^n C_i}{n} + \gamma I_d$, and shares it back with clients.
8: for iteration $k = 1, \dots, T$ do
9:     for user $i = 1, \dots, n$ do
10:        Compute and average local gradients:
           $$g_i^k = \frac{\sum_{x \in D_i} \nabla f(\theta^k, x)}{|D_i|}$$
           .
11:        Clip and perturb local updates multiplied with a preconditioner:
           $$u_i^k = \text{clip}_{C_g}(C^{-1} \cdot g_i^k) + E_i^k, E_i^k \sim \mathcal{N}\left(0, \frac{I_d (C_g \sigma_g)^2}{n}\right).$$
12:        Share noisy update $u_i^k$ with server.
13:    end for
14:    Server aggregates $\{u_i^k\}_{i=1}^n$, and computes global noisy update $U^k = \frac{\sum_{i=1}^n u_i^k}{n}$.
15:    Update model parameters and share with users: $\theta^{k+1} = \theta^k - \eta \cdot U^k$.
16: end for
```

Algorithm 7 Record-level DP-FedGD Algorithm (Distributed Version of Full-Batch DP-SGD)

```
1: Input: dataset $D = \{D_i\}_{i=1}^n$, noise levels $\sigma_g$, clipping constants $C_g$, training length $T$, learning rate $\eta$.
2: for iteration $k = 1, \ldots, T$ do
3:     for user $i = 1, \ldots, n$ do
4:         Clip and perturb the avg. of local gradients:
           $$u_i^k = \frac{\sum_{x \in D_i} \text{clip}_{C_g}\left(\nabla f(\theta^k, x)\right) + E_i^k}{|D_i|}, \quad E_i^k \sim \mathcal{N}\Big(0, \frac{I_d (C_g \sigma_g)^2}{n}\Big).$$
5:         Share $u_i^k$ with server.
6:     end for
7:     Server aggregates $\{u_i^k\}_{i=1}^n$, computes the global noisy update $U^k = \frac{\sum_{i=1}^n u_i^k}{n}$.
8:     Update model parameters and share with users:
           $$\theta^{k+1} = \theta^k - \eta \cdot U^k.$$
9: end for
```

Algorithm 8 User-level DP-FedGD Algorithm (Distributed Version of Full-Batch DP-SGD)

```
1: Input: dataset $D = \{D_i\}_{i=1}^n$, noise levels $\sigma_g$, clipping constants $C_g$, training length $T$, learning rate $\eta$.
2: for iteration $k = 1, \ldots, T$ do
3:     for user $i = 1, \ldots, n$ do
4:         Compute and average local gradients:
           $$g_i^k = \frac{\sum_{x \in D_i} \nabla f(\theta^k, x)}{|D_i|}.$$
5:         Clip and perturb local updates:
           $$u_i^k = \text{clip}_{C_g}(g_i^k) + E_i^k, \quad E_i^k \sim \mathcal{N}\Big(0, \frac{I_d (C_g \sigma_g)^2}{n}\Big).$$
6:         Share noisy update $u_i^k$ with server.
7:     end for
8:     Server aggregates $\{u_i^k\}_{i=1}^n$, and computes global noisy update $U^k = \frac{\sum_{i=1}^n u_i^k}{n}$.
9:     Update model parameters and share with users: $\theta^{k+1} = \theta^k - \eta \cdot U^k$.
10: end for
```

B Tables of Hyperparameter Grids Used in the Experiments

Table 7 depicts the hyperparameter grids used for the hyperparameter tuning in the experiments.

C Details on the Privacy Analysis

In this work, we provide an accurate (ε, δ)-analysis for our methods using the hockey-stick divergence. This way, we are able to get optimal privacy parameters for a given sensitivity analysis of the data-dependent functions and in particular we obtain lower bounds than using, e.g., the Rényi differential privacy (RDP) [35] which is a commonly used alternative.

We next shortly describe the mathematical results needed for obtaining accurate (ε, δ)-DP bounds using the hockey-stick divergence.... The (ε, δ)-DP as defined in 1 can be characterized using the hockey-stick divergence as follows. For $\alpha > 0$ the hockey-stick divergence H_α from a distribution P to a distribution Q is defined as

$$H_\alpha(P||Q) = \int [P(t) - \alpha \cdot Q(t)]_+ \, \mathrm{d}t,$$

where for $t \in \mathbb{R}$, $[t]_+ = \max\{0, t\}$. Tight (ε, δ)-values for a given mechanism can be obtained using the hockey-stick-divergence:

Lemma 10 (Zhu et al. 45). *For a given $\varepsilon \geq 0$, tight $\delta(\varepsilon)$ is given by the expression*

$$\delta(\varepsilon) = \max_{D \sim D'} H_{\mathrm{e}^\varepsilon}(\mathcal{M}(D)||\mathcal{M}(D')).$$

Thus, if we can bound the divergence $H_{\mathrm{e}^\varepsilon}(\mathcal{M}(D)||\mathcal{M}(D'))$ accurately, we also obtain accurate $\delta(\varepsilon)$-bounds. To this end we need to consider so-called dominating pairs of distributions:

Definition 11 (Zhu et al. 45). *A pair of distributions (P, Q) is a* dominating pair *of distributions for mechanism $\mathcal{M}(D)$ if for all neighboring datasets D and D' and for all $\alpha > 0$,*

$$H_\alpha(\mathcal{M}(D)||\mathcal{M}(D')) \leq H_\alpha(P||Q).$$

If the equality holds for all α for some D, D', then (P, Q) is a tightly dominating pair of distributions.

When analyzing iterative DP-FL training methods, we model them as adaptive compositions such that the adversary has a view on the output of all intermediate outputs. This means that we analyze mechanisms of the form

$$\begin{aligned}\mathcal{M}^{(T)}(D) = \big(&\mathcal{M}_1(D), \mathcal{M}_2(\mathcal{M}_1(D), D), \ldots, \\ &\mathcal{M}_T(\mathcal{M}_1(D), \ldots, \mathcal{M}_{T-1}(D), D)\big).\end{aligned} \tag{C.1}$$

In the methods we propose, each $\mathcal{M}_i$, $i \in [T]$, will correspond to a Gaussian mechanism with a given sensitivity and noise scale.

We get upper bounds for adaptive compositions using the dominating pairs of distributions as follows:

Theorem 12 (Zhu et al. 45). *If (P, Q) dominates $\mathcal{M}$ and (P', Q') dominates $\mathcal{M}'$, then $(P \times P', Q \times Q')$ dominates the adaptive composition $\mathcal{M} \circ \mathcal{M}'$.*

To convert the hockey-stick divergence from $P \times P'$ to $Q \times Q'$ into an efficiently computable form, we consider so called privacy loss random variables.

Definition 13. *Let P and Q be probability density functions. We define the privacy loss random variable (PRV) $\omega_{P/Q}$ as*

$$\omega_{P/Q} = \log \frac{P(t)}{Q(t)}, \quad t \sim P(t).$$

PRVs can be utilized for obtaining accurate privacy guarantees via the following result.

Theorem 14 (Gopi et al. 18). *The $\delta(\varepsilon)$-bounds can be represented using the following representation that involves the PRV:*

$$H_{\mathrm{e}^\varepsilon}(P||Q) = \mathop{\mathbb{E}}_{s \sim \omega_{P/Q}} \left[1 - \mathrm{e}^{\varepsilon - s}\right]_+. \tag{C.2}$$

Moreover, if $\omega_{P/Q}$ is the PRV for the pair of distributions (P, Q) and $\omega_{P'/Q'}$ the PRV for the pair of distributions (P', Q'), then the PRV for the pair of distributions $(P \times P', Q \times Q')$ is given by $\omega_{P/Q} + \omega_{P'/Q'}$.

Given a dominating pair of distributions (P, Q) for a mechanism $\mathcal{M}$, 14 is all that is needed for obtaining (ε, δ)-bounds for $\mathcal{M}$. In some cases, such as in the case of the Gaussian mechanism, this expression leads to analytical bounds Balle and Wang [see, e.g., 4]. In the general case, Fast Fourier Technique-based methods [18, 25] can be used to numerically evaluate the convolutions appearing when summing the PRVs and evaluating the expression C.2.

In this work, the methods we propose are based on additive Gaussian noise and the privacy analysis is equivalent to that of the Gaussian mechanism.

Hockey-stick divergence between two Gaussians. Let $x_0, x_1 \in \mathbb{R}$, $\sigma \geq 0$, and let P be the density function of $\mathcal{N}(x_0, \sigma^2)$ and Q the density function of $\mathcal{N}(x_1, \sigma^2)$. Then, the PRV $\omega_{P/Q}$ is distributed as [Lemma 11 by 39]

$$\omega_{P/Q} \sim \mathcal{N}\left(\frac{(x_0 - x_1)^2}{2\sigma^2}, \frac{(x_0 - x_1)^2}{\sigma^2}\right). \tag{C.3}$$

Thus, in particular: $H_\alpha(P||Q) = H_\alpha(Q||P)$ for all $\alpha > 0$. Plugging in PLD $\omega_{P/Q}$ to the expression (C.2), we find that for all $\varepsilon \geq 0$, the hockey-stick divergence $H_{\mathrm{e}^\varepsilon}(P||Q)$ is given by the expression

$$\delta(\varepsilon) = \Phi\left(-\frac{\varepsilon\sigma}{\Delta} + \frac{\Delta}{2\sigma}\right) - e^\varepsilon \Phi\left(-\frac{\varepsilon\sigma}{\Delta} - \frac{\Delta}{2\sigma}\right), \tag{C.4}$$

where Φ denotes the CDF of the standard univariate Gaussian distribution and $\Delta = |x_0 - x_1|$. This formula was originally given by Balle and Wang [4].

If $\mathcal{M}$ is of the form $\mathcal{M}(D) = f(D) + Z$, where $f : \mathcal{D}^N \to \mathbb{R}^d$ and $Z \sim \mathcal{N}(0, \sigma^2 I_d)$, and $\Delta = \max_{D \simeq D'} \|f(D) - f(D')\|_2$ gives the L_2-sensitivity, then for $x_0 = 0, x_1 = \Delta$, (P, Q) of the above form gives a tightly dominating pair of distributions for $\mathcal{M}$ [45]. Subsequently, by Theorem 14, $\mathcal{M}$ is (ε, δ)-DP for $\delta(\varepsilon)$ given by (C.4).

It also directly follows from Theorem 14 and the form of the PRV (C.3) that the PRV for the adaptive composition of T Gaussian mechanisms is given by $\omega_{P/Q} \sim \mathcal{N}\left(\frac{T \cdot \Delta^2}{2\sigma^2}, \frac{T \cdot \Delta^2}{\sigma^2}\right)$ and we obtain the following expression.

Lemma 15. *Consider an adaptive composition of T Gaussian mechanisms, each with L_2-sensitivity Δ and noise scale parameter σ. The adaptive composition is (ε, δ)-DP for*

$$\delta(\varepsilon) = \Phi\left(-\frac{\varepsilon\sigma}{\sqrt{T} \cdot \Delta} + \frac{\sqrt{T} \cdot \Delta}{2\sigma}\right) - e^\varepsilon \Phi\left(-\frac{\varepsilon\sigma}{\sqrt{T} \cdot \Delta} - \frac{\sqrt{T} \cdot \Delta}{2\sigma}\right).$$

Table 7: Hyperparameter grids.

method	Hyperparameter	alternatives	privacy level	grid size
DP-FedGD	η	$\{0.001, 0.01, 0.1, 1, 10\}$	user/record	15
	C_g	{0.1,1,2}	user/record	
DP-Scaffold	η_l	$\{0.001, 0.01, 0.1, 1, 10\}$	user/record	30
	C	{0.1,1}	user/record	
	M	{5,10,20}	user/record	
	η_g	1	user/record	
DP-FedNew/DP-FedNew-FC	α	$\{0.01, 0.1, 1\}$	user/record	96
	ρ	$\{0.01, 0.1, 1, 10\}$	user/record	
	η	$\{0.01, 0.1, 1, 10\}$	user/record	
	C, Δ_H	$\{0.1, 1\}$	user/record	
	C_1, C_2	1	record	
DP-FedFC	η	$\{0.001, 0.01, 0.1, 1, 10\}$	user/record	20
	C_c	{0.1,1}	user/record	
	C_g	{0.1,1}	user/record	
	γ	0.001	user/record	

D Proof of Lemma 3 (Record-Level Sensitivity Bound)

For the proof of Lemma 3 we first need the following auxiliary lemma:

Lemma 16. *Suppose $A, B \in \mathbb{R}^{d\times d}$ are positive definite, $\|\cdot\|$ is a matrix norm and $\|A-B\|\|A^{-1}\| < 1$. Then*

$$\|A^{-1} - B^{-1}\| \leq \frac{\|A-B\|\|A^{-1}\|^2}{1-\|A-B\|\|A^{-1}\|}.$$

Proof. The proof can be found in Section 5.8 of [20] (see also Lemma C.4 by Ganesh et al. [17]). □

Lemma 17. *Let Δ_i be defined as in (3). Let Δ_H be an upper bound for the norm of H, i.e., an upper bound for the norm of data-sample-wise Hessian. Assume*

$$\begin{aligned} \|\nabla^2 f(x,\theta^k)\|_F &\leq \Delta_H \quad \text{for all } x \in D_i, \\ \|\nabla f(x,\theta^k)\|_2 &\leq C_1 \quad \text{for all } x \in D_i, \\ \|g_i^k - \lambda_i^{k-1} + \rho y^{k-1}\| &\leq C_2, \end{aligned} \tag{D.1}$$

and $\gamma = \alpha + \rho > \frac{\Delta_H}{|D_i|}$. Then, we have:

$$\|\Delta_i\|_2 \leq \frac{1}{\gamma\cdot|D_i|}\cdot C_1 + \frac{\Delta_H}{\gamma^2\cdot|D_i| - \gamma\cdot\Delta_H}\cdot C_2,$$

where $|D_i|$ is the size of the local dataset D_i.

Proof. For ease of notation, consider the function

$$f(D_i,\gamma,\theta) = (H_i^k + \gamma I)^{-1}(g_i^k + \theta),$$

where $\gamma > 0$ is a constant and θ stands for auxiliary variables. We need to bound the 2-norm of

$$\Delta_i = f(D_i', \gamma, \theta) - f(D_i, \gamma, \theta),$$

where $D_i' = D_i \bigcup \{x'\}$ for some data-element x'. Adding and subtracting $(H_i^k + H' + \gamma I)^{-1}(g_i^k + \theta)$ to Δ_i, we have:

$$\begin{aligned} \Delta_i &= (H_i^k + H' + \gamma I)^{-1}(g_i^k + g' + \theta) - (H_i^k + \gamma I)^{-1}(g_i^k + \theta) \\ &= (H_i^k + H' + \gamma I)^{-1}(g_i^k + g' + \theta) - (H_i^k + H' + \gamma I)^{-1}(g_i^k + \theta) \\ &+ (H_i^k + H' + \gamma I)^{-1}(g_i^k + \theta) - (H_i^k + \gamma I)^{-1}(g_i^k + \theta) \\ &= (H_i^k + H' + \gamma I)^{-1} g' \\ &\quad + \left((H_i^k + H' + \gamma I)^{-1} - (H_i^k + \gamma I)^{-1}\right)(g_i^k + \theta). \end{aligned} \tag{D.2}$$

For the first term on the right-hand side of (D.2) we use the following fact: if a matrix A is positive definite with smallest eigenvalue $\lambda_{\min}$, then $\|A^{-1}\| = \lambda_{\min}^{-1}$. Clearly, since H_i^k and H' are positive semidefinite, $(H_i^k + H' + \gamma I)^{-1}$ is positive definite with smallest eigenvalue larger than γ, and we have that

$$\begin{aligned} \|(H_i^k + H' + \gamma I)^{-1} g'\|_2 &\leq \|(H_i^k + H' + \gamma I)^{-1}\|_2 \|g'\|_2 \\ &\leq \frac{1}{\gamma}\|g'\|_2 \leq \frac{1}{\gamma}\cdot\frac{C_1}{|D_i|}. \end{aligned}$$

By Lemma 16 and the assumptions (D.1) we have that

$$\begin{aligned} &\|\left((H_i^k + H' + \gamma I)^{-1} - (H_i^k + \gamma I)^{-1}\right)(g_i^k + \theta)\|_2 \\ &\leq \|(H_i^k + H' + \gamma I)^{-1} - (H_i^k + \gamma I)^{-1}\|_2 \|g_i^k + \theta\|_2 \\ &\leq \frac{\|H'\|_F \|(H_i^k + H' + \gamma I)^{-2}\|_2}{1 - \|(H_i^k + H' + \gamma I)^{-1}\|_2 \|H'\|_F}\cdot C_2 \\ &\leq \frac{\|H'\|_F\cdot\gamma^{-2}}{1-\gamma^{-1}\|H'\|_F}\cdot C_2 \\ &\leq \frac{\frac{\Delta_H}{|D_i|}\gamma^{-2}}{1-\gamma^{-1}\frac{\Delta_H}{|D_i|}}\cdot C_2 \\ &= \frac{\Delta_H}{\gamma^2\cdot|D_i| - \gamma\Delta_H}\cdot C_2. \end{aligned}$$

□

E Proof of Lemma 4

Lemma 18. *Let $a, b \in \mathbb{R}^n$ and $C > 0$. If we set*

$$\xi = \frac{-2\langle a, \frac{b}{\|b\|_2}\rangle + \sqrt{4\langle a, \frac{b}{\|b\|_2}\rangle^2 + 4(C^2 - \|a\|_2^2)}}{2\|b\|_2},$$

we have that

$$\|a + \xi \cdot b\|_2 = C.$$

Proof. Denote by $\hat{b}$ a unit vector in the direction of b. Setting the right-hand side of

$$\|a + b\|_2^2 = \|a\|_2^2 + 2\|b\|_2\langle a, \hat{b}\rangle + \|b\|_2^2$$

equal to C^2 and solving the quadratic equation for $\|b\|_2$, we arrive at the claim.

□

F Convergence Analysis

F.1 Non-Private Analysis by Elgabli et al. [16]

To make following the DP convergence analysis easier to follow, we review here the main results of [16] and depict the main story of their analysis.

Consider the non-private FedNew algorithm, i.e., the variables y and λ are those given by the algorithm described in Section 3.

The convergence analysis of [16] is starts with the following auxiliary lemma.

Lemma 19. *Consider one iteration of FedNew. Assume the per-example approximations of the Hessian at user i, H_i^k, is positive semi-definite. Denote the dual residual $s^k := \rho(y^k - y^{k-1})$. We have:*

$$\mathbb{E}\langle \lambda_i^k + s^k - \lambda^{k,*}, y_i^k - y^{k,*}\rangle \leq -\alpha\mathbb{E}\|y^{k,*} - y_i^k\|^2 + \sigma^2 \cdot \mathrm{Trace}(H_i^k).$$

Next, the inequality given in Lemma 19 is reformulated to obtain the inequality of Lemma 20 below. This reformulation requires, however, two additional assumptions. First, it is assumed that for the function $Q_i(\theta, y) = \frac{1}{2}y(\nabla^2 f_i(\theta) + \alpha I)y - y^T\nabla f_i(\theta)$ we have that

$$\|\nabla_y Q_i(\theta_1, y_1) - \nabla_y Q_i(\theta_2, y_2)\|_2 \leq L_q\|y_1 - y_2\|_2 \tag{F.1}$$

for some constant $L_q > 0$. We remark that the condition (F.1) this is a fairly strong requirement. However, one can easily show that this holds for the linear regression, for example, since then $\nabla^2 f_i(\theta)$ independent of θ for all $i \in [n]$. Another requirement for Lemma 20 is that the iterates of FedNew satisfy the inequality

$$\|y^k - y^{k-1}\|_2 \leq \|y^k - y^{k,*}\|_2. \tag{F.2}$$

As we show in detail below in Section F.2, this inequality is satisfied for large enough values of the regularization parameter ρ.

With these assumption the following technical result is shown next. This will lead us to formulating a Lyapunov function for the iteration.

Lemma 20. *Assume the conditions* (F.1) *and* (F.2) *hold true for all $k \in [T]$. For any $\beta \leq \alpha - 2.5\rho - \frac{8L_q^2 n}{\rho}$, the iterates of FedNew satisfy the inequality*

$$\begin{aligned}
&\frac{1}{\rho}\sum_{i=1}^n \|\lambda_i^k - \lambda_i^{k,*}\|_2^2 + 2\beta\sum_{i=1}^n \|y_i^k - y^{k,*}\|_2^2 + \rho n\|y^k - y^{k,*}\|_2^2 + 2\rho n\|y^k - y^{k-1}\|_2^2 \\
&\leq \frac{1}{\rho}\sum_{i=1}^n \|\lambda_i^{k-1} - \lambda_i^{k-1,*}\|_2^2 \\
&+ \frac{2L_q^2}{\rho}\sum_{i=1}^n \|y_i^{k-1} - y^{k-1,*}\|_2^2 + \frac{4L_q^2 n}{\rho}\|y^{k-1} - y^{k-1,*}\|_2^2 \\
&+ 2\rho n\|y^{k-1} - y^{k-2}\|_2^2.
\end{aligned}$$

From Lemma (20) if follows that the Lyapunov function V_k defined as

$$\begin{aligned}
V_k := \frac{1}{\rho}\sum_{i=1}^n \|\lambda_i^k - \lambda^{k,*}\|_2^2 + 2\beta_1\sum_{i=1}^n \|y_i^k - y^{k,*}\|_2^2 \\
+ \rho n\|y^k - y^{k,*}\|_2^2 + 2\rho n\|y^k - y^{k-1}\|_2^2
\end{aligned} \tag{F.3}$$

satisfies the inequality

$$V_k \leq V_{k-1} - \beta_2\sum_{i=1}^n \|y_i^k - y^{k,*}\|^2 \tag{F.4}$$

for some constant $\beta_2 > 0$.

The main result of [16] follows from the inequality (F.4). We give an alternative proof for it, to also motivate the proof of our DP result.

Theorem 21. *As the number of iterations $k \to \infty$, the local ADMM iterates approach the Newton updates, i.e.,*

$$\|y_i^k - y^{k,*}\|_2 \to 0.$$

Proof. From Lemma 20 it follows that the Luapynov function V_k defined in (F.3) satisfies the inequality (F.4). Since V_k is non-negative and monotonously decreasing, by the monotone convergence theorem it has a limit as $k \to \infty$. Thus, $\beta_2\sum_{i=1}^n \|y_i^k - y^{k,*}\|^2 \to 0$ as $k \to \infty$ from which the claim follows. □

F.2 Assumption Used in Thm. 27

In the proof of Theorem 27 we need to assume that the iterates of the non-private FedNew algorithm satisfy

$$\|y^k - y^{k-1}\|_2 \leq \|y^k - y^{k,*}\|_2. \tag{F.5}$$

for all $k \in [T]$. As the following result shows, this in a reasonable assumption for a large enough ρ.

Lemma 22. *There exists $\rho_0 > 0$ such that for all $\rho \geq \rho_0$, the assumption* (F.5) *holds true.*

Proof. Recall that

$$\begin{aligned}
y^k &= \frac{1}{n}\sum_{i=1}^n y_i^k \\
&= \frac{1}{n}\sum_{i=1}^n (H_i^k + \alpha I + \rho I)^{-1}(g_i^k - \lambda_i^{k-1} + \rho \cdot y^{k-1}).
\end{aligned}$$

Therefore

$$y^k - y^{k-1} = \frac{1}{n}\sum_{i=1}^{n}\Big[(H_i^k + \alpha I + \rho I)^{-1}\big(g_i^k - \lambda_i^{k-1} + \rho\cdot y^{k-1} - (H_i^k + \alpha I + \rho I)y^{k-1}\big)\Big]$$
$$= \frac{1}{n}\sum_{i=1}^{n}(H_i^k + \alpha I + \rho I)^{-1}\big(g_i^k - \lambda_i^{k-1} - (H_i^k + \alpha I)y^{k-1}\big)$$

which shows that $\|y^k - y^{k-1}\|_2 \to 0$ as $\rho \to \infty$. On the other hand,

$$y^k - y^{k,*} = \frac{1}{n}\sum_{i=1}^{n}\Big[(H_i^k + \alpha I + \rho I)^{-1}\big(g_i^k - \lambda_i^{k-1} + \rho\cdot y^{k-1}\big) - (H_i^k + \alpha I)\big(g_i^k - \lambda_i^*(\theta^k)\big)\Big]$$
$$= \frac{1}{n}\sum_{i=1}^{n}\Big[y^{k-1} + (H_i^k + \alpha I + \rho I)^{-1}\cdot \big(g_i^k - \lambda_i^{k-1} - (H_i^k + \alpha I)y^{k-1}\big) - (H_i^k + \alpha I)\big(g_i^k - \lambda_i^*(\theta^k)\big)\Big]. \quad \text{(F.6)}$$

since

$$y_i^{k,*} = (H_i^k + \alpha I)^{-1}\big(g_i^k - \lambda_i^*(\theta^k)\big).$$

We see from (F.6) that

$$y^k - y^{k,*} \to \frac{1}{n}\sum_{i=1}^{n}\Big[y^{k-1} - (H_i^k + \alpha I)\big(g_i^k - \lambda_i^*(\theta^k)\big)\Big]$$

as $\rho \to \infty$. Thus, □

F.3 First Step of the DP Convergence Analysis: Proof of Lemma 23

The following result is a stochastic version of [Lemma 1, 16] and applies for the noisy update rule (5.2).

Lemma 23. *Consider one iteration of DP-FedNew. Assume the per-example approximations of the Hessian at user i at iteration k, H_i^k, is positive semidefinite. Denote the dual residual $s^k := \rho(y^k - y^{k-1})$. Assume in the added noise $C = 1$. We have:*

$$\mathbb{E}\langle \lambda_i^k + s^k - \lambda^{k,*}, \widetilde{y}_i^k - y^{k,*}\rangle \le -\alpha\mathbb{E}\|y^{k,*} - \widetilde{y}_i^k\|^2 + C^2\cdot\sigma^2\cdot \mathrm{Trace}(H_i^k) + \sigma^2\cdot(\alpha+\rho)\cdot d,$$

where the expectation is taken over the randomness of E_i^k, the noise added by the user i at iteration k.

Proof. It follows from the noisy update rule (5.2) that

$$(H_i^k + \alpha I)(\widetilde{y}_i^k - E_i^k) - g_i^k + \lambda_i^{k-1} + \rho(\widetilde{y}_i^k - y^{k-1}) - \rho E_i^k = 0. \quad \text{(F.7)}$$

Substituting the update of the dual variable

$$\lambda_i^{k-1} = \lambda_i^k - \rho(\widetilde{y}_i^k - y^k)$$

to Eq. (F.7) gives

$$(H_i^k + \alpha I)(\widetilde{y}_i^k - E_i^k) - g_i^k + \lambda_i^k + \rho(y^k - y^{k-1}) - \rho E_i^k = 0.$$

Recall the dual residual $s^k = \rho(y^k - y^{k-1})$. We get

$$\lambda_i^k + s^k = g_i^k - (H_i^k + \alpha I)(\widetilde{y}_i^k - E_i^k) + \rho E_i^k. \quad \text{(F.8)}$$

Recall that for the optimal values of the primal and dual variables of the non-DP dual iteration at the global iteration k for user i, y_i^{*k} and λ_i^{*k}, respectively, we have that

$$\lambda_i^{*k} = g_i^k - (H_i^k + \alpha I)y_i^{*k} = g_i^k - (H_i^k + \alpha I)y^{*k},$$

since $y_i^{*k} = y^{*k}$. Subtracting λ_i^{*k} from both sides of Eq. (F.8), we get

$$\lambda_i^k + s^k - \lambda^{*k} = (H_i^k + \alpha I)(E_i^k - \widetilde{y}_i^k) + (H_i^k + \alpha I)y^{*k} + \rho E_i^k = (H_i^k + \alpha I)(y^{*k} - \widetilde{y}_i^k + E_i^k) + \rho E_i^k. \quad \text{(F.9)}$$

where y^{*k} is the converged result of the non-DP dual iteration at global iteration k with Hessian evaluated at θ^k. Taking inner product of between both sides of Eq. (F.9) and the vector $\widetilde{y}_i^k - y^{*k}$, we get

$$\langle \lambda_i^k + s^k - \lambda^{*k}, \widetilde{y}_i^k - y^{*k}\rangle$$
$$= \langle (H_i^k + \alpha I)(y^{*k} - \widetilde{y}_i^k + E_i^k), \widetilde{y}_i^k - y^{*k}\rangle + \rho\langle E_i^k, \widetilde{y}_i^k - y^{*k}\rangle$$
$$= -\langle (H_i^k + \alpha I)(\widetilde{y}_i^k - y^{*k}), \widetilde{y}_i^k - y^{*k}\rangle + \langle (H_i^k + \alpha I)E_i^k, \widetilde{y}_i^k - y^{*k}\rangle + \rho\langle E_i^k, \widetilde{y}_i^k - y^{*k}\rangle$$
$$\le -\alpha\|y^{*k} - \widetilde{y}_i^k\|^2 + \langle (H_i^k + \alpha I)E_i^k, \widetilde{y}_i^k - y^{*k}\rangle + \rho\langle E_i^k, \widetilde{y}_i^k - y^{*k}\rangle \quad \text{(F.10)}$$

since H_i^k is positive semidefinite. Recall:

$$\widetilde{y}_i^k = \widehat{y}_i^k + E_i^k,$$

where $\widehat{y}_i^k$ denotes the non-perturbed update, i.e.

$$\widehat{y}_i^k = (H_i^k + \alpha I + \rho I)^{-1}(g_i^k - \lambda_i^{k-1} + \rho y^{k-1}).$$

Thus, we can write (F.10) as

$$\langle \lambda_i^k + s^k - \lambda^{*k}, \widetilde{y}_i^k - y^{*k}\rangle \le -\alpha\|y^{*k} - \widetilde{y}_i^k\|^2 + \langle (H_i^k + \alpha I)E_i^k, \widehat{y}_i^k + E_i^k - y^{*k}\rangle + \rho\langle E_i^k, y^{*k} - \widetilde{y}_i^k\rangle. \quad \text{(F.11)}$$

Since $\mathbb{E}_{x\sim\mathcal{N}(0,I_d)}x^TAx = \mathrm{Trace}(A)$ for any square matrix A and since $E_i^k \sim \mathcal{N}(0, \sigma^2 I_d)$, we have

$$\mathbb{E}\langle \lambda_i^k + s^k - \lambda^{*k}, \widetilde{y}_i^k - y^{*k}\rangle$$
$$\le -\alpha\mathbb{E}\|y^{*k} - \widetilde{y}_i^k\|^2 + \mathbb{E}\langle (H_i^k + \alpha I)E_i^k, E_i^k\rangle + \mathbb{E}\langle (H_i^k + \alpha I)E_i^k, \widehat{y}_i^k - y^{*k}\rangle + \rho\mathbb{E}\langle E_i^k, \widetilde{y}_i^k - y^{*k}\rangle$$
$$= -\alpha\mathbb{E}\|y^{*k} - \widetilde{y}_i^k\|^2 + \mathbb{E}\langle (H_i^k + \alpha I)E_i^k, E_i^k\rangle + \rho\mathbb{E}\langle E_i^k, \widehat{y}_i^k + E_i^k - y^{*k}\rangle$$
$$= -\alpha\mathbb{E}\|y^{*k} - \widetilde{y}_i^k\|^2 + \sigma^2\cdot\mathrm{Trace}(H_i^k + \alpha I) + \rho\cdot d$$
$$= -\alpha\|y^{*k} - \widetilde{y}_i^k\|^2 + \sigma^2\cdot\mathrm{Trace}(H_i^k) + \sigma^2\cdot(\alpha+\rho)\cdot d,$$

where the expectation is taken over the randomness of E_i^k. □

In case f is β-smooth, we have the following corollary.

Corollary 24. *If the loss function f is β-smooth, then*

$$\mathbb{E}\langle \lambda_i^k + s^k - \lambda^{*k}, \widetilde{y}_i^k - y^{*k}\rangle \le -\mathbb{E}\alpha\|y^{*k} - \widetilde{y}_i^k\|^2 + \sigma^2\cdot(\alpha+\rho+\beta)\cdot d,$$

where the expectation is taken over the randomness of the local additive noise E_i^k.

Proof. If f is β-smooth, since it is twice differentiable, we have that $\|H_i^k\|_2 \leq \beta$. Since the trace of a square matrix equals the sum of its singular values, $\text{Trace}(H_i^k) \leq d \cdot \beta$ and the claim follows from Lemma 23. □

F.4 Additional Auxiliary Results

In addition to the conditions (F.1), we need the assumption that the iterates of DP-FedNew satisfy the inequality

$$\mathbb{E}\|y^k - y^{k-1}\|_2^2 \leq \mathbb{E}\|y^k - y^{k,*}\|_2^2 \tag{F.12}$$

for all $k \in [T]$, where the expectation is taken over the additive normally distributed noises at iteration k. This condition is true in case the (F.1) for the non-private FedNew is true, since $\mathbb{E}_X\|Y + X\|_2^2 = \|Y\|_2^2 + \mathbb{E}\|X\|_2^2$ for all random variables X with $\mathbb{E}X = 0$.

Lemma 25. *Assume the conditions* (F.1) *and* (F.12) *are satisfied for all* $k \in [T]$. *For any* $\beta \leq \alpha - 2.5\rho - \frac{8L_q^2 n}{\rho}$, *the iterates of FedNew satisfy the inequality*

$$\begin{aligned}
&\frac{1}{\rho}\mathbb{E}\sum_{i=1}^n \|\lambda_i^k - \lambda_i^{k,*}\|_2^2 + 2\beta\mathbb{E}\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|_2^2 \\
&+ \rho n\mathbb{E}\|y^k - y^{k,*}\|_2^2 \\
&+ 2\rho n\mathbb{E}\|y^k - y^{k-1}\|_2^2 \\
&\leq \frac{1}{\rho}\sum_{i=1}^n \|\lambda_i^{k-1} - \lambda_i^{k-1,*}\|_2^2 + \frac{2L_q^2}{\rho}\sum_{i=1}^n \|y_i^{k-1} - y^{k-1,*}\|_2^2 \\
&+ \frac{4L_q^2 n}{\rho}\|y^{k-1} - y^{k-1,*}\|_2^2 + 2\rho n\|y^{k-1} - y^{k-2}\|_2^2 + \sigma^2 \cdot (\alpha + \rho) \cdot d,
\end{aligned}$$

where the expectation is taken over the additive normally distributed noises at iteration k.

The proof of Lemma 20 is obtained by carrying out a lengthy refactorization to the inequality of Lemma 19 and can be found in [16]. The proof of Lemma 25 is given by exactly the same refactorization applied to the inequality given by 23.

As a result of Lemma 25 we define a Lyapunov function for the stochastic DP-FedNew iteration and show the following inequality for it.

Lemma 26. *Let the Lyapunov function* V_k *be defined as*

$$\begin{aligned}
V_k := &\frac{1}{\rho}\sum_{i=1}^n \|\lambda_i^k - \lambda^{k,*}\|_2^2 + 2\beta_1\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|_2^2 \\
&+ \rho n\|y^k - y^{k,*}\|_2^2 + 2\rho n\|y^k - y^{k-1}\|_2^2,
\end{aligned} \tag{F.13}$$

where $\widetilde{y}_i^k$ *denotes the noisy update* (5.2) *Denote* $\widetilde{V}_k = \mathbb{E}V_k$, *where the expectation is taken over all additive noises up to iteration* k. *Then,* $\widetilde{V}_k$ *satisfies*

$$\widetilde{V}_k \leq \widetilde{V}_{k-1} - \beta_2\mathbb{E}\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|^2 + \sigma^2 \cdot (\alpha + \rho) \cdot d \tag{F.14}$$

for some constant $\beta_2 > 0$, *where the expectation is taken over the noise added at iteration* k.

F.5 Our Main Theorem

Theorem 27. *For all* $k \in \mathbb{Z}$, *there exists* $\ell > k$ *such that*

$$\|y^\ell - y^{\ell,*}\|^2 \leq \frac{\sigma^2 \cdot (\alpha + \rho) \cdot d}{n\beta_2}. \tag{F.15}$$

Proof. At a given iteration k, either

$$\beta_2\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|^2 \leq \sigma^2 \cdot (\alpha + \rho) \cdot d \tag{F.16}$$

which implies the inequality (F.15) for $\ell = k$ (combining with the inequality $\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|^2 \geq n\|y^k - y^{k,*}\|^2$), or then

$$\beta_2\sum_{i=1}^n \|\widetilde{y}_i^k - y^{k,*}\|^2 > \sigma^2 \cdot (\alpha + \rho) \cdot d,$$

which by the Lemma 26 implies that either $\widetilde{V}_k$ converges from which case the claim follows, or then there is an $\ell > k$ such that (F.16) holds from which case the claim follows. □

G Additional Tabular Results

Table 8: Effect of varying the local dataset size. The numbers in the ε columns report the mean test accuracy after the final 70th epoch for the best model trained on IID Adult dataset obtained after tuning the hyperparameters. The layer size is 123×2. Note that DP-FedNew-FC attains accuracies comparable to DP-FedFC without aggregating the covariance matrix of size 123×123. DP-FedNew-FC can perform $\frac{123}{2} \approx 62$ additional epochs with the number of bits needed to transmit this covariance matrix.

			$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 8$	$\epsilon = 10$
n	$\lvert D_i \rvert$	method									
10	2818	DP-FFC	0.824	0.824	0.824	0.825	0.831	0.834	0.834	0.835	0.841
		DP-FGD	0.822	0.822	0.822	0.822	0.823	0.822	0.822	0.822	0.822
		DP-FN	0.824	0.822	0.824	0.825	0.833	0.825	0.830	0.830	0.832
		DP-FN-FC	0.823	0.824	0.828	0.830	0.833	0.831	0.833	0.833	0.834
		DP-Scaffold	0.814	0.818	0.818	0.818	0.818	0.818	0.818	0.817	0.818
100	306	DP-FFC	0.823	0.825	0.825	0.824	0.833	0.833	0.833	0.827	0.839
		DP-FGD	0.821	0.822	0.823	0.822	0.822	0.822	0.822	0.822	0.822
		DP-FN	0.823	0.826	0.812	0.826	0.833	0.825	0.826	0.832	0.832
		DP-FN-FC	0.822	0.823	0.825	0.827	0.829	0.834	0.834	0.834	0.834
		DP-Scaffold	0.793	0.804	0.815	0.818	0.818	0.819	0.818	0.819	0.818
250	123	DP-FFC	0.823	0.825	0.825	0.826	0.833	0.833	0.834	0.842	0.841
		DP-FGD	0.821	0.822	0.822	0.822	0.823	0.822	0.822	0.822	0.822
		DP-FN	0.819	0.824	0.819	0.829	0.834	0.824	0.825	0.832	0.833
		DP-FN-FC	0.818	0.824	0.826	0.828	0.828	0.831	0.830	0.831	0.830
		DP-Scaffold	0.778	0.776	0.776	0.768	0.812	0.818	0.819	0.820	0.822
500	61	DP-FFC	0.820	0.827	0.825	0.825	0.834	0.833	0.833	0.839	0.842
		DP-FGD	0.821	0.823	0.822	0.822	0.822	0.822	0.822	0.822	0.822
		DP-FN	0.822	0.825	0.827	0.828	0.834	0.825	0.826	0.831	0.833
		DP-FN-FC	0.822	0.826	0.828	0.827	0.826	0.830	0.830	0.830	0.830
		DP-Scaffold	0.766	0.788	0.798	0.810	0.811	0.821	0.821	0.822	0.822

Table 9: Effect of varying the local dataset size. The numbers in the ε columns report the mean test accuracy after the final 70th epoch for the best model trained on IID CIFAR10 dataset obtained after tuning the hyperparameters. The layer size is 2048×10. Note that DP-FedNew-FC attains accuracies comparable to DP-FedFC without aggregating the covariance matrix of size 2048×2048. DP-FedNew-FC can perform $\frac{2048}{10} \approx 204$ additional epochs with the number of bits needed to transmit this covariance matrix.

n	$\|D_i\|$	method	$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 8$	$\epsilon = 10$
20	2500	DP-FedNew-FC	0.627	0.694	0.715	0.720	0.727	0.736	0.737	0.762	0.771
		DP-Scaffold	0.422	0.580	0.634	0.648	0.658	0.669	0.670	0.669	0.669
		DP-FedFC	0.627	0.693	0.714	0.727	0.731	0.743	0.748	0.771	0.775
		DP-FedGD	0.634	0.691	0.724	0.724	0.730	0.734	0.729	0.735	0.734
50	1000	DP-FedNew-FC	0.624	0.690	0.694	0.714	0.728	0.736	0.738	0.771	0.768
		DP-Scaffold	0.337	0.530	0.584	0.611	0.627	0.638	0.637	0.641	0.642
		DP-FedFC	0.626	0.692	0.716	0.724	0.732	0.738	0.746	0.771	0.775
		DP-FedGD	0.633	0.690	0.716	0.721	0.725	0.728	0.730	0.736	0.736
100	500	DP-FedNew-FC	0.635	0.668	0.698	0.718	0.722	0.735	0.737	0.761	0.767
		DP-Scaffold	0.278	0.466	0.535	0.567	0.589	0.606	0.612	0.615	0.615
		DP-FedFC	0.629	0.689	0.715	0.725	0.732	0.741	0.750	0.762	0.776
		DP-FedGD	0.631	0.692	0.715	0.724	0.730	0.733	0.734	0.737	0.735
250	200	DP-FedNew-FC	0.626	0.665	0.697	0.709	0.719	0.744	0.746	0.764	0.767
		DP-Scaffold	0.219	0.372	0.436	0.478	0.501	0.537	0.538	0.544	0.547
		DP-FedFC	0.628	0.693	0.715	0.725	0.731	0.740	0.745	0.771	0.773
		DP-FedGD	0.635	0.690	0.718	0.721	0.727	0.732	0.733	0.734	0.731
500	100	DP-FedNew-FC	0.637	0.665	0.698	0.708	0.716	0.734	0.747	0.772	0.778
		DP-Scaffold	0.176	0.296	0.347	0.379	0.399	0.427	0.425	0.439	0.433
		DP-FedFC	0.629	0.692	0.712	0.725	0.733	0.741	0.747	0.772	0.774
		DP-FedGD	0.634	0.692	0.712	0.724	0.727	0.729	0.733	0.731	0.731

Table 10: Effect of varying the local dataset size. The numbers in the ε columns report the mean test accuracy after the final 70th epoch for the best model trained on IID FashionMNIST dataset obtained after tuning the hyperparameters. The layer size is 2048×10. Note that DP-FedNew-FC attains accuracies comparable to DP-FedFC without aggregating the covariance matrix of size 2048×2048. DP-FedNew-FC can perform $\frac{2048}{10} \approx 204$ additional epochs with the number of bits needed to transmit this covariance matrix.

n	$\lvert D_i \rvert$	method	$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 8$	$\epsilon = 10$
20	3000	DP-FedNew-FC	0.697	0.760	0.786	0.795	0.793	0.803	0.810	0.837	0.836
		DP-Scaffold	0.506	0.675	0.706	0.716	0.724	0.728	0.729	0.731	0.731
		DP-FedFC	0.689	0.773	0.788	0.793	0.799	0.821	0.824	0.841	0.843
		DP-FedGD	0.696	0.774	0.788	0.794	0.795	0.798	0.799	0.799	0.798
50	1200	DP-FedNew-FC	0.693	0.754	0.776	0.795	0.799	0.799	0.810	0.835	0.841
		DP-Scaffold	0.420	0.599	0.631	0.648	0.651	0.655	0.657	0.660	0.659
		DP-FedFC	0.689	0.774	0.790	0.794	0.798	0.817	0.822	0.841	0.843
		DP-FedGD	0.693	0.775	0.789	0.792	0.797	0.798	0.800	0.799	0.799
100	600	DP-FedNew-FC	0.688	0.755	0.775	0.785	0.804	0.799	0.814	0.834	0.839
		DP-Scaffold	0.362	0.529	0.583	0.603	0.604	0.613	0.615	0.612	0.614
		DP-FedFC	0.688	0.776	0.790	0.795	0.797	0.818	0.822	0.841	0.843
		DP-FedGD	0.690	0.773	0.791	0.794	0.798	0.798	0.800	0.799	0.799
250	240	DP-FedNew-FC	0.691	0.754	0.772	0.780	0.798	0.810	0.810	0.834	0.838
		DP-Scaffold	0.289	0.447	0.522	0.542	0.551	0.560	0.555	0.554	0.559
		DP-FedFC	0.688	0.772	0.790	0.796	0.797	0.818	0.822	0.841	0.843
		DP-FedGD	0.683	0.771	0.789	0.795	0.798	0.798	0.799	0.798	0.799
500	120	DP-FedNew-FC	0.691	0.753	0.773	0.780	0.791	0.805	0.819	0.836	0.836
		DP-Scaffold	0.197	0.380	0.433	0.463	0.479	0.502	0.503	0.494	0.507
		DP-FedFC	0.690	0.773	0.790	0.795	0.798	0.818	0.823	0.841	0.844
		DP-FedGD	0.687	0.772	0.791	0.793	0.796	0.799	0.798	0.800	0.798

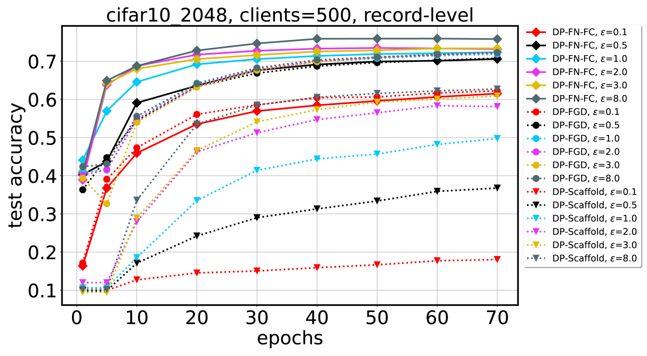
Table 11: Effect of varying the local dataset size. The numbers in the ε columns report the mean test accuracy after the final 70th epoch for the best model trained on IID EMNIST dataset obtained after tuning the hyperparameters. The layer size is 2048×10. Note that DP-FedNew-FC attains accuracies comparable to DP-FedFC without aggregating the covariance matrix of size 2048×2048. DP-FedNew-FC can perform $\frac{2048}{10} \approx 204$ additional epochs with the number of bits needed to transmit this covariance matrix.

			$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 8$	$\epsilon = 10$
n	$\lvert D_i \rvert$	method									
50	4800	DP-FedNew-FC	0.868	0.893	0.897	0.913	0.925	0.936	0.938	0.940	0.945
		DP-Scaffold	0.679	0.807	0.822	0.827	0.830	0.832	0.832	0.832	0.832
		DP-FedFC	$\underline{0.879}$	$\underline{0.912}$	$\underline{0.931}$	$\underline{0.936}$	$\underline{0.940}$	$\underline{0.945}$	$\underline{0.951}$	$\underline{0.959}$	$\underline{0.959}$
		DP-FedGD	0.878	0.893	0.896	0.894	0.894	0.896	0.894	0.896	0.895
100	2400	DP-FedNew-FC	0.867	0.900	0.897	0.912	0.925	0.936	0.938	0.940	0.940
		DP-Scaffold	0.626	0.755	0.772	0.780	0.782	0.785	0.785	0.786	0.785
		DP-FedFC	$\underline{0.880}$	$\underline{0.911}$	$\underline{0.929}$	$\underline{0.936}$	$\underline{0.940}$	$\underline{0.945}$	$\underline{0.951}$	$\underline{0.959}$	$\underline{0.959}$
		DP-FedGD	0.878	0.893	0.894	0.893	0.894	0.894	0.895	0.894	0.896
250	960	DP-FedNew-FC	0.856	0.894	0.912	0.912	0.925	0.936	0.938	0.940	0.940
		DP-Scaffold	0.509	0.663	0.694	0.701	0.703	0.705	0.708	0.705	0.707
		DP-FedFC	$\underline{0.879}$	$\underline{0.911}$	$\underline{0.930}$	$\underline{0.936}$	$\underline{0.940}$	$\underline{0.945}$	$\underline{0.951}$	$\underline{0.959}$	$\underline{0.959}$
		DP-FedGD	0.878	0.893	0.893	0.895	0.894	0.895	0.894	0.895	0.895
500	480	DP-FedNew-FC	0.854	0.891	0.906	0.917	0.925	0.936	0.938	0.940	0.940
		DP-Scaffold	0.431	0.605	0.639	0.651	0.656	0.660	0.664	0.663	0.668
		DP-FedFC	$\underline{0.878}$	$\underline{0.910}$	$\underline{0.930}$	$\underline{0.935}$	$\underline{0.940}$	$\underline{0.944}$	$\underline{0.951}$	$\underline{0.959}$	$\underline{0.959}$
		DP-FedGD	0.877	0.893	0.893	0.894	0.895	0.896	0.894	0.896	0.894

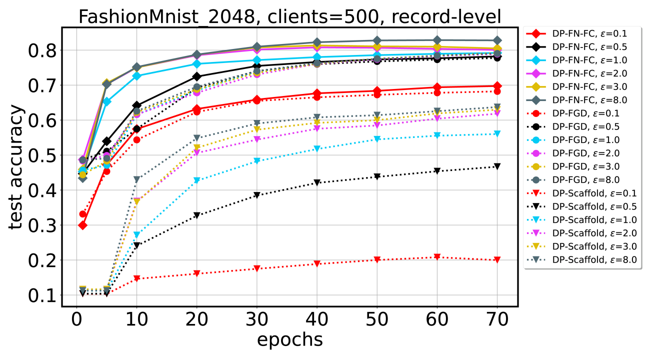
Table 12: Effect of ρ and α while varying the local dataset sizes. The numbers in the columns $\{1, 10, 30, 50, 70\}$ report the mean test accuracies in the correspond epochs for the top-3 most accurate models trained on EMNIST dataset for $\varepsilon = 1$. The layer size is 64×10. The value in the columns $\bar{\lambda}^j_{max}$ is computed by averging the maximum eigen value of local Hessians from all clients at epoch j. We tune the hyperparameters $\alpha = \{0.01, 0.1, 1\}$, and $\rho = \{0.01, 0.1, 1, 10, 100\}$. $\eta = C_1 = C_2 = \Delta_H = 1$.

					1	10	30	50	70	$\bar{\lambda}^1_{max}$	$\bar{\lambda}^{10}_{max}$	$\bar{\lambda}^{30}_{max}$	$\bar{\lambda}^{50}_{max}$	$\bar{\lambda}^{70}_{max}$
n	$\lvert D_i \rvert$	η	ρ	α										
50	4800	1	1	0.001	0.116	0.341	0.484	0.533	0.574	0.369	0.326	0.243	0.219	0.208
				0.010	0.148	0.346	0.485	0.533	0.573	0.436	0.370	0.252	0.223	0.216
			0.1	0.001	0.213	0.115	0.494	0.549	0.536	0.708	1.018	0.183	0.134	0.128
100	2400	1	0.1	0.010	0.228	0.192	0.405	0.542	0.579	0.723	0.088	0.544	0.200	0.204
			1	0.001	0.133	0.360	0.490	0.536	0.577	0.402	0.367	0.249	0.222	0.210
				0.010	0.144	0.359	0.490	0.533	0.572	0.389	0.375	0.253	0.226	0.218
250	960	1	1	0.001	0.216	0.327	0.493	0.538	0.575	0.360	0.346	0.252	0.225	0.211
				0.010	0.130	0.332	0.488	0.535	0.575	0.367	0.351	0.255	0.228	0.220
			0.1	0.100	0.257	0.359	0.436	0.494	0.533	0.846	0.420	0.339	0.314	0.303
500	480	1	1	0.001	0.148	0.372	0.498	0.537	0.576	0.338	0.314	0.255	0.229	0.217
				0.010	0.105	0.342	0.484	0.532	0.571	0.373	0.338	0.257	0.234	0.225
				0.100	0.187	0.327	0.465	0.501	0.524	0.366	0.339	0.289	0.273	0.264

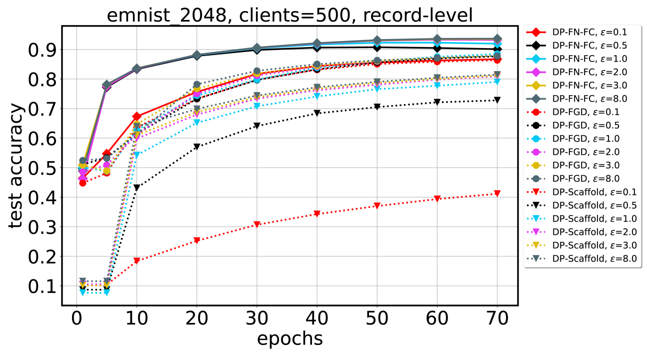
H Record-level DP results for layer size 2048×10.



(a) CIFAR10



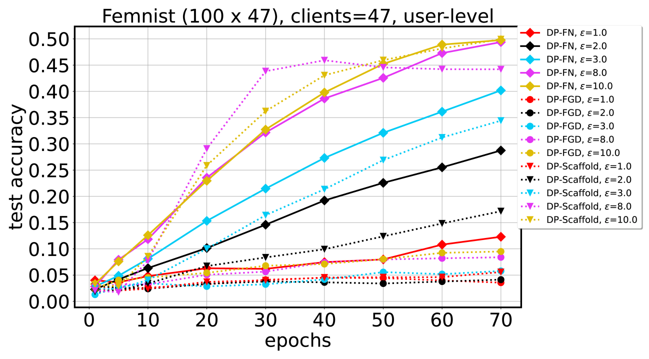
(b) FashionMNIST



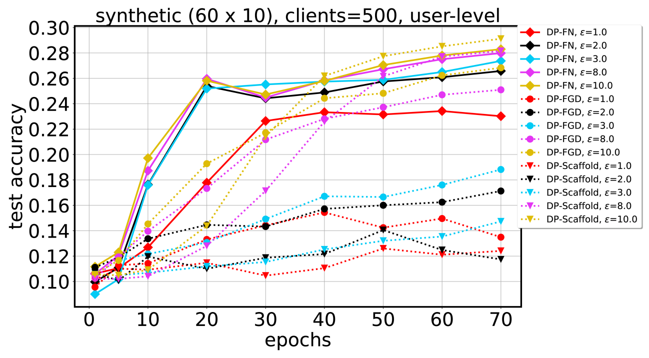
(c) Federated EMNIST

Figure 5: IID data: Record-level results for DP-FedNew-FC (DP-FN-FC in plots), DP-FedGD (DP-FGD in plots), and DP-Scaffold. For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size is 2048×10. Check Figure 10 for comparison with DP-FedFC.

I User-level DP experiments: Non-IID data



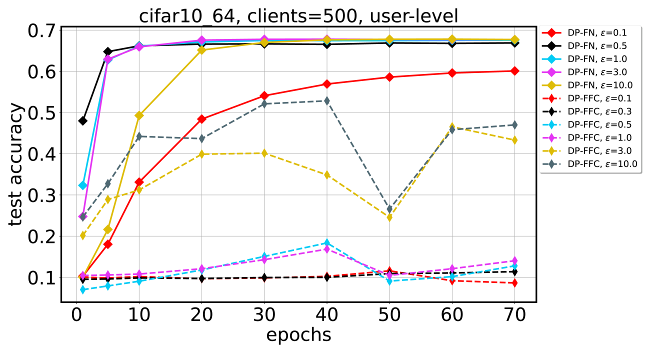
(a) Federated EMNIST



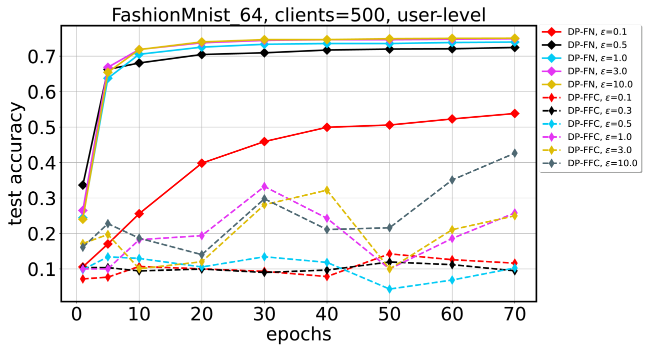
(b) Synthetic data

Figure 6: Non-IID data: User-level results for DP-FedNew (DP-FN in plots) and DP-FedGD (DP-FGD in plots). For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model sizes are 100×47 and 60×10.

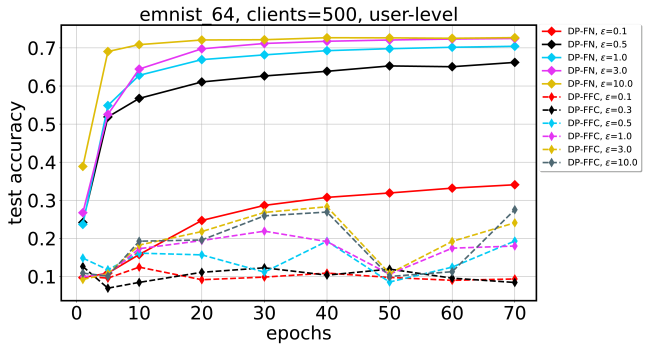
J Comparison with DP-FedFC.



(a) CIFAR10

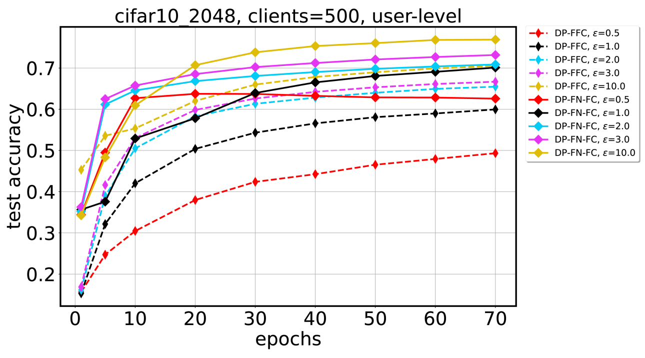


(b) FashionMNIST

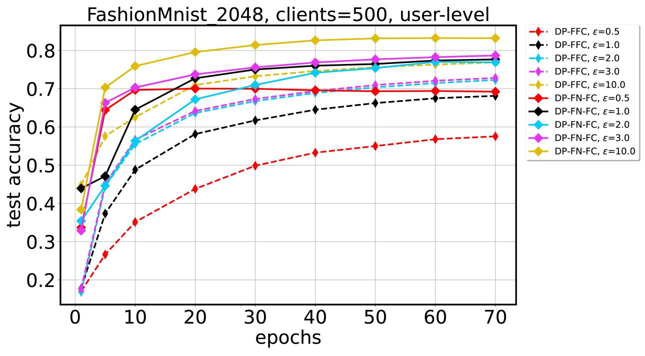


(c) EMNIST

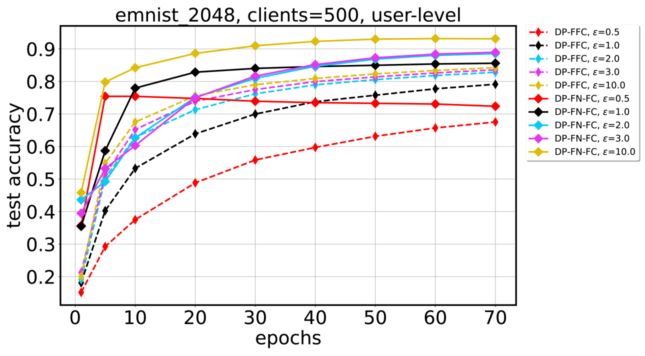
Figure 7: IID data: User-level results for DP-FedNew (DP-FN), DP-FedFC (DP-FFC in plots). For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size is 64×10.



(a) CIFAR10

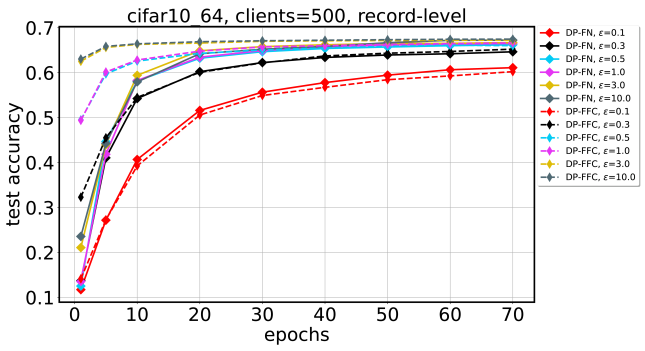


(b) FashionMNIST

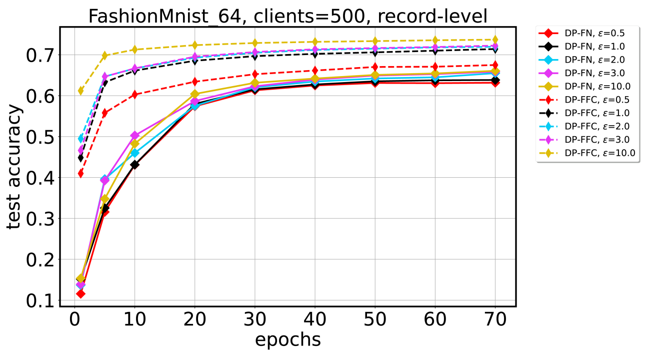


(c) EMNIST

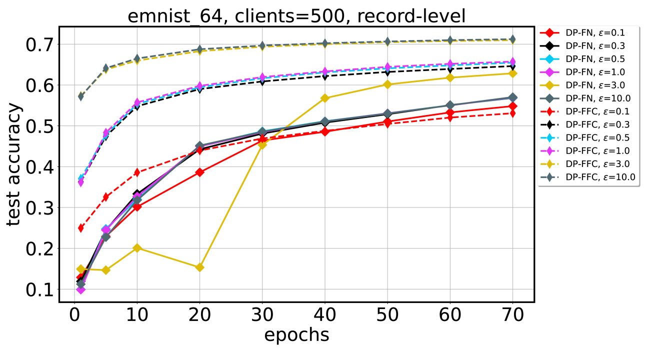
Figure 8: IID data: User-level results for DP-FedNew-FC (DP-FN-FC in plots), DP-FedFC (DP-FFC in plots). For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size is 2048×10.



(a) CIFAR10

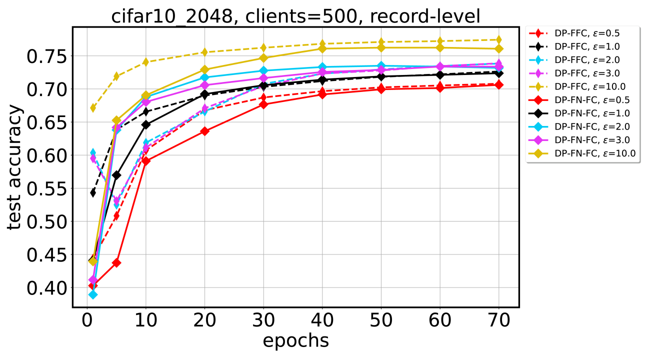


(b) FashionMNIST

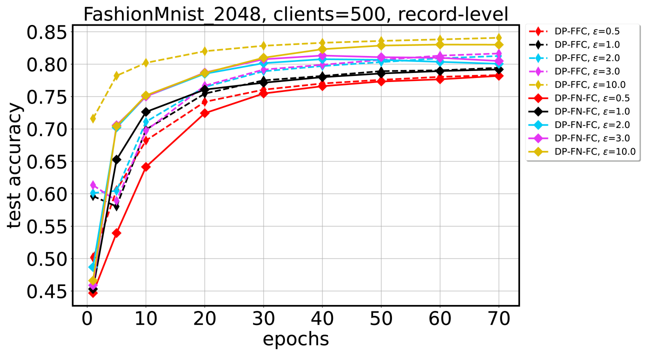


(c) EMNIST

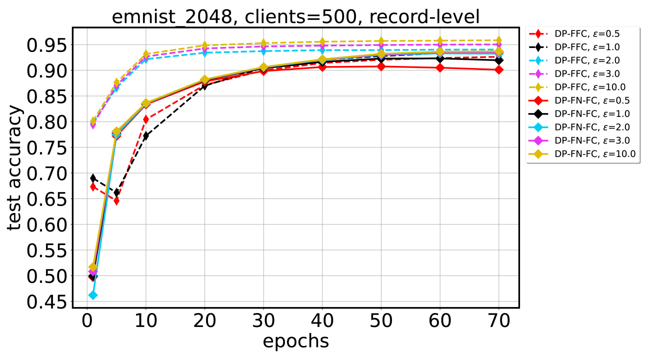
Figure 9: IID data: Record-level results for DP-FedNew (DP-FN in plots), DP-FedFC (DP-FFC in plots). For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size is 64×10.



(a) CIFAR10



(b) FashionMNIST



(c) EMNIST

Figure 10: IID data: Record-level results for DP-FedNew-FC (DP-FN-FC in plots), DP-FedFC (DP-FFC in plots). For each ϵ, we plot the test accuracies of the best model obtained after hyperparameter tuning. The model size is 2048×10.