Machine Learning (2024) 113:2955–3022
https://doi.org/10.1007/s10994-023-06332-x

Federated learning with superquantile aggregation for heterogeneous data

Krishna Pillutla[1] · **Yassine Laguel**[2] · **Jérôme Malick**[3] · **Zaid Harchaoui**[4]

Received: 19 November 2021 / Revised: 25 January 2023 / Accepted: 24 March 2023 /
Published online: 16 May 2023
© The Author(s), under exclusive licence to Springer Science+Business Media LLC, part of Springer Nature 2023

Abstract
We present a federated learning framework that is designed to robustly deliver good predictive performance across individual clients with heterogeneous data. The proposed approach hinges upon a superquantile-based learning objective that captures the tail statistics of the error distribution over heterogeneous clients. We present a stochastic training algorithm that interleaves differentially private client filtering with federated averaging steps. We prove finite time convergence guarantees for the algorithm: $O(1/\sqrt{T})$ in the nonconvex case in T communication rounds and $O(\exp(-T/\kappa^{3/2}) + \kappa/T)$ in the strongly convex case with local condition number κ. Experimental results on benchmark datasets for federated learning demonstrate that our approach is competitive with classical ones in terms of average error and outperforms them in terms of tail statistics of the error.

Keywords Federated learning · Data heterogeneity · Distribution shift · Risk measure · Distributed optimization · Stochastic optimization

Editors: Dana Drachsler Cohen, Javier Garcia, Mohammad Ghavamzadeh, Marek Petrik, Philip S. Thomas.

Krishna Pillutla, Yassine Laguel have contributed equally to this work.

✉ Krishna Pillutla
pillutla@cs.washington.edu

✉ Yassine Laguel
yassine.laguel@rutgers.edu

Jérôme Malick
jerome.malick@univ-grenoble-alpes.fr

Zaid Harchaoui
zaid@uw.edu

[1] Google Research, Mountain View, USA

[2] Rutgers University, New Jersey, USA

[3] CNRS, Grenoble, France

[4] University of Washington, Seattle, USA

1 Introduction

Federated learning is a distributed machine learning framework where many clients (e.g. mobile devices) collaboratively train a model under the orchestration of a central server (e.g. service provider) while keeping the training data private and local to the client throughout the training process (McMahan et al., 2017; Kairouz et al., 2021). It has found widespread adoption across industry (Bonawitz et al., 2019; Paulik et al., 2021) for artificial intelligence applications ranging from smart device apps (Yang et al., 2018; Hard et al., 2018) to healthcare (Brisimi et al., 2018; Huang et al., 2019).

A key feature of federated learning is the statistical heterogeneity, i.e., client data distributions are *not* identically distributed (Kairouz et al., 2021; Li et al., 2020). In typical cross-device federated learning scenarios, each client corresponds to a user. The diversity in the data they generate reflects the diversity in their unique personal, cultural, regional, and geographical characteristics.

This data heterogeneity in federated learning manifests itself as a train-test distributional shift. Indeed, the usual approach minimizes the prediction error of the model on average over the population of clients available for training (McMahan et al., 2017) while at test time, the same model is deployed on individual clients. This approach can fail on clients whose data distribution is far from most of the population or who may have less data than most of the population. It is highly desirable, therefore, to have a federated learning method that can robustly deliver good predictive performance across a wide variety of natural distribution shifts posed by individual clients.

We present in this paper a robust approach to federated learning that guarantees a minimum level of predictive performance to all clients, even in situations where the population is heterogeneous. The method we develop addresses these issues by minimizing a learning objective based on the notion of a superquantile (Rockafellar & Uryasev, 2002; Rockafellar et al., 2008), a risk measure that captures the tail behavior of a random variable.

Training models with a learning objective involving the superquantile raises challenges. The superquantile is a non-smooth functional with sophisticated properties. Furthermore, the superquantile function can be seen as a kind of nonlinear expectation that we would like to blend well with averaging mechanisms. We show how to address the former by leveraging the dual formulation and the latter by leveraging the tail-domain viewpoint. As a result, we can obtain an algorithm that can be implemented in a similar way to FedAvg (McMahan et al., 2017) yet offers important benefits to heterogeneous populations.

The approach we propose, Δ-FL, allows one to control higher percentiles of the distribution of errors over the heterogeneous population of clients; see Fig. 1 for an illustration. We show in the experiments that our approach is more efficient than a direct approach, simply seeking to minimize the worst error over the population of clients. Compared to FedAvg, Δ-FL delivers improved prediction to tail clients or data-poor clients. Our algorithm relies on differentially private quantile computation to filter out clients on which to run federated averaging steps. We present finite-time theoretical convergence guarantees for our algorithm when used to train additive models or deep networks and prove bounds on the privacy and utility of the algorithm.

1.1 Contributions

We make the following concrete contributions in this work.

The Δ-FL Framework The usual objective of federated learning, which we call the *vanilla FL* objective is

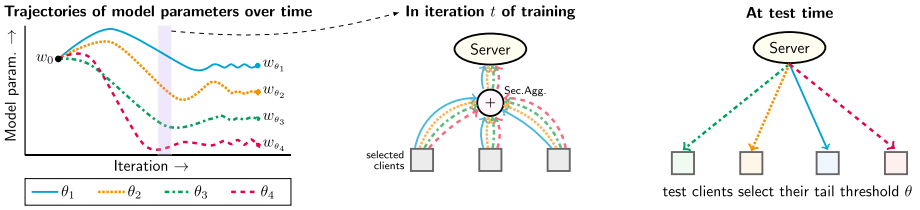

Fig. 1 Schematic summary of the Δ-FL framework. **Left**: The server maintains multiple models w_{θ_j}, one for each tail threshold θ_j. **Middle**: During training, selected clients participate in training *each* model w_{θ_j}. Individual updates are securely aggregated to update the server model. **Right**: Each test user is allowed to select their tail threshold θ, and are served the corresponding model w_θ

$$\min_{w\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} F_i(w) + \frac{\lambda}{2}\|w\|^2, \tag{1}$$

where $F_i(w) = \mathbb{E}_{\xi\sim q_i}[f(w;\xi)]$ is the expected loss on client i under its data distribution q_i for $i = 1,\dots,n$, and λ is a regularization parameter (McMahan et al., 2017). Minimizing the average loss can lead to poor performance on clients whose distribution p is far from the population training distribution $p_{\text{train}} = (1/n)\sum_{i=1}^{n} q_i$. Our goal is to improve the performance on such *tail clients*.

To this end, we directly minimize the average loss across tail clients whose loss is above a certain tail threshold. We formalize this through the notion of a risk measure known as the **superquantile**, a tail summary statistic of random variables (Rockafellar & Uryasev, 2002). The $(1-\theta)$-superquantile is defined for a continuous random variable Z and $\theta \in (0,1)$ as $\mathbb{S}_\theta(Z) = \mathbb{E}[Z \mid Z > Q_\theta(Z)]$, where $Q_\theta(Z)$ is the $(1-\theta)$-quantile of Z. A similar interpretation holds for discrete distributions; the formal definition of the superquantile for this case is given in Sect. 3.3.

Instead of minimizing the average loss as in (1), the proposed framework Δ-FL minimizes the tail loss across clients, as measured by the superquantile. Concretely, at a tail threshold $\theta \in (0,1)$, we minimize

$$F_\theta(w) := \mathbb{S}_\theta\big(F_1(w),\dots,F_n(w)\big) + \frac{\lambda}{2}\|w\|^2, \tag{2}$$

where $\mathbb{S}_\theta(a_1,\dots,a_n)$ is the $(1-\theta)$-superquantile of the empirical distribution $(1/n)\sum_{i=1}^{n}\delta_{a_i}$. Thus, the objective (2) measures the tail statistics of the per-client loss distribution.

By a duality argument, we show that the superquantile objective (2) promotes distributional robustness. If we have a test client who is unseen during training and whose distribution $p_\pi = \sum_{i=1}^{n}\pi_i q_i$ can be written as a mixture of the training distributions $q_1,\dots,q_n$, then the Δ-FL objective can be written as

$$F_\theta(w) = \max_{\pi_i \le 1/(\theta n)} \mathbb{E}_{\xi\sim p_\pi}\big[f(w;\xi)\big] + \frac{\lambda}{2}\|w\|^2.$$

In other words, we minimize the *worst-case loss over all mixture distributions with a constraint* $\pi_i \le 1/(\theta n)$ on the mixture weights; see Sect. 4.1 for details.

Optimization Algorithms. To design a federated optimization algorithm to optimize the Δ-FL objective, the nonsmoothness of the superquantile $a \mapsto \mathbb{S}_\theta(a_1,\dots,a_n)$ might lead to potential difficulties in optimization. Fortunately, we can derive an expression for the subgradient of the Δ-FL objective (2): when θn is an integer, we have

$$\sum_{i=1}^{n} \pi_i^\star F_i(w) + \lambda w \in \partial F_\theta(w)\,, \quad \text{where} \quad \pi_i^\star = \frac{\mathbb{I}(F_i(w) \geq Q_\theta)}{\sum_{j=1}^{n} \mathbb{I}(F_j(w) \geq Q_\theta)}\,,$$

and $Q_\theta = Q_\theta(F_1(w), \ldots, F_n(w))$ is the $(1-\theta)$-quantile of the losses evaluated at w. In other words, averaging the gradients of the losses that are larger than the quantile Q_θ gives a valid subgradient of the objective (2).

Using this expression, we design a federated optimization algorithm that interleaves federated averaging with differentially private quantile estimation. Specifically, the local updates w_i^+ from the subsample of m selected clients $i \in S$ are aggregated to update the global model with the following two steps:

- Estimate $\hat{Q}_\theta \approx Q_\theta(F_i(w) \,:\, i \in S)$ using the distributed discrete Gaussian mechanism (Kairouz et al., 2021) and hierarchical histograms (Cormode et al., 2019), and
- Aggregate the updates from the tail clients where $F_i(w) \geq \hat{Q}_\theta$ to find the new global model w^+ as

$$w^+ = \frac{1}{|S_\theta|} \sum_{i \in S_\theta} w_i^+\,, \quad \text{where} \quad S_\theta = \{i \,:\, F_i(w) \geq \hat{Q}_\theta\}\,.$$

Similar to FedAvg, this aggregation rule enjoys a simplification in the case of a single local update per-client with a learning rate γ. Specifically, under the assumption of full client participation (i.e., $m = n$), if the local update $w - w_i^+ = \gamma \nabla \left(F_i(w) + (\lambda/2)\|w\|^2\right)$ is a single gradient step and $\hat{Q}_\theta = Q_\theta(F_1(w), \ldots, F_n(w))$ is the exact quantile of the per-client losses, the aggregated update is simply a subgradient step $w - w^+ = \gamma \nabla F_\theta(w)$ where we denote the subgradient as $\nabla F_\theta(w) \in \partial F_\theta(w)$.

Convergence analysis Apart from the nonsmoothness of the superquantile, the convergence analysis also has to overcome the difficulty that we cannot obtain unbiased minibatch subgradient estimators for the superquantile objective.

Given m i.i.d. copies $Z_1, \ldots, Z_m$ of a random variable Z, the empirical mean $\bar{Z}_m = (1/m)\sum_{i=1}^m Z_i$ is an unbiased estimate of the population mean, i.e., $\mathbb{E}[\bar{Z}_m] = \mathbb{E}[Z]$. This is no longer true for the superquantile, i.e., $\mathbb{E}[\mathbb{S}_\theta(Z_1, \ldots, Z_m)] \neq \mathbb{S}_\theta(Z)$. As a result, we cannot access unbiased stochastic gradients in the learning setting, where m is the minibatch size. Moreover, it is not reasonable to assume in federated learning that we have access to all the clients due to a diurnal availability pattern of clients (Kairouz et al., 2021). We overcome this issue by actually minimizing the *expected minibatch superquantile* instead. It is defined as

$$\overline{F}_\theta(w) := \mathbb{E}_{(i_1,\ldots,i_m)\sim U_m}\left[\mathbb{S}_\theta\left(F_{i_1}(w), \ldots, F_{i_m}(w)\right)\right],$$

where U_m is the uniform distribution over all subsets of $\{1, \ldots, n\}$ of batch size m. We can build an unbiased subgradient estimator for this objective by sampling a minibatch $(i_1, \ldots, i_m) \sim U_m$. This is a uniform close surrogate of the original objective, as shown by Levy et al. (2020, Prop. 1)

$$|F_\theta(w) - \overline{F}_\theta(w)| \leq O\left(\frac{\max_{i=1,\ldots,n} |F_i(w)|}{\sqrt{\theta m}}\right).$$

Assuming that each F_i is G-Lipschitz and L-smooth, we establish a rate of $\sqrt{LG^2/T}$ in the nonconvex (and nonsmooth) case where $\lambda = 0$. If, additionally, each F_i is convex and

$\lambda > 0$, the problem is strongly convex and we establish a rate of $\exp(-T/\kappa^{3/2}) + G^2/(\lambda T)$ in this case where $\kappa = 1 + L/\lambda$ is the per-client condition number.

Privacy and utility analysis The standard algorithms to compute quantiles with differential privacy are based on the exponential mechanism and require a trusted central aggregator (Smith, 2011). Since this is not usually the case in federated learning, we estimate the cumulative distribution using the hierarchical histogram method and combine it with the distributed discrete Gaussian mechanism (Kairouz et al., 2021) in order to simulate a central aggregation using a cryptographic primitive known as secure aggregation (Bonawitz et al., 2017). The hierarchical histogram method, also known as tree aggregation, is a classical approach to answer range queries under differential privacy (Hay et al., 2010; Dwork et al., 2010; Chan et al., 2011; Smith et al., 2017; Cormode et al., 2019).

Privacy guarantees are obtained by adding noise to the per-client computations, resulting in a degradation of utility (i.e., the performance relative to the non-private case). This leads to a tradeoff between privacy and utility. For a hierarchical histogram of b bins, we prove a $(1/2)\epsilon^2$ -concentrated differential privacy (Bun & Steinke, 2016) guarantee given a per-client noise of scale $\log b/(\varepsilon\sqrt{n})$ and a quantile error of $\log^2 b/(\varepsilon n)$ up to constants and log factors.

Experiments We perform numerical experiments using neural networks and linear models on tasks including image classification and sentiment analysis based on public datasets. The experiments demonstrate the superior performance of Δ-FL over state-of-the-art baselines on the upper quantiles of the error on test clients, with particular improvements on data-poor clients, while being competitive on the mean error. A deeper analysis reveals that Δ-FL helps improve performance on data-poor clients.

We numerically study the privacy-utility tradeoff of the differentially private quantile estimation algorithm described above and the Δ-FL algorithm with end-to-end differential privacy guarantees. We find that Δ-FL outperforms FedAvg on the tail error across a wide range of privacy budgets while exhibiting a comparable privacy-utility tradeoff to FedAvg on the mean error.

1.2 Outline

We start with Sect. 2 to describe the related work. Section 3 describes the general setup, recalls the FedAvg algorithm, and formally defines the superquantile as a tail summary of a random variable. Section 4 presents a federated optimization algorithm for Δ-FL. We analyze its convergence in the convex and non-convex cases, as well as its differential privacy properties in Sect. 5. We discuss an extension to other risk measures and relations to fair allocation in Sect. 6. Section 7 presents experimental results, comparing the proposed approach to existing ones. Detailed proofs and additional details can be found in the appendices, while the code and the scripts to reproduce the experiments can be found at https://github.com/krishnap25/simplicial-fl.

An early version of this work was presented at IEEE CISS (Laguel et al., 2021). This paper extends and improves upon it in several respects. First, we give an improved and tighter convergence analysis in both the convex and general nonconvex cases. Second, we augment our algorithm with differential privacy and analyze its privacy and utility. Finally, we conduct an expanded numerical study, including (a) comparing with baselines such as Tilted-ERM (Li et al., 2021) that were published after our paper (Laguel et al., 2021), (b) an empirical comparison to model personalization, and, (c) a study of the privacy-utility tradeoff of Δ-FL under differential privacy.

Notation. The norm $\|\cdot\|$ denote the Euclidean norm $\|\cdot\|_2$ in $\mathbb{R}^d$. We use $\Delta^{n-1} = \left\{\pi \in \mathbb{R}^n_+ \ : \ \sum_{i=1}^n \pi_i = 1\right\}$ to denote the probability simplex in $\mathbb{R}^n$.

2 Related work

Federated learning was introduced by (McMahan et al., 2017) to handle distributed on-client learning (Kairouz et al., 2021; Li et al., 2020; Gafni et al., 2022). A plethora of recent extensions have also been proposed (Yurochkin et al., 2019; Sattler et al., 2020; Mills et al., 2020; Wei et al., 2020; Mohammadi Amiri & Gündüz, 2020; Shlezinger et al., 2021; Jhunjhunwala et al., 2021; Sery et al., 2021; Collins et al., 2021). Our approach to addressing the statistical heterogeneity by proposing a new objective is broadly applicable in these settings.

Distributionally robust optimization (Ben-Tal et al., 2013), which aims to train models that perform uniformly well across all subgroups instead of just on average, has witnessed a flurry of recent research (Lee & Raginsky, 2018; Duchi & Namkoong, 2019; Kuhn et al., 2019). This approach is closely related to the risk measures studied in economics and finance (Artzner et al., 1999; Rockafellar & Uryasev, 2000; Ben-Tal & Teboulle, 2007; Föllmer & Schied, 2016). The recent works (Laguel et al., 2020; Levy et al., 2020; Curi et al., 2020) study optimization algorithms for risk measures. More broadly, risk measures have been successfully utilized in problems ranging from bandits (Sani et al., 2012; Cassel et al., 2018), reinforcement learning (Chow et al., 2015; Tamar et al., 2015; Chow et al., 2017), and fairness in machine learning (Williamson & Menon, 2019; Rezaei et al., 2021). The federated learning method here is based on the superquantile (Rockafellar & Uryasev, 2002), a popular risk measure. We propose a stochastic optimization algorithm adapted to the federated setting and prove its convergence.

Addressing statistical heterogeneity in federated learning has led to two lines of work. The first includes algorithmic advances to alleviate the effect of heterogeneity on convergence rates while still minimizing the classical expectation-based objective function of empirical risk minimization. These techniques include the use of proximal terms (Li et al., 2020), control variates (Karimireddy et al., 2020) or augmenting the server updates (Wang et al., 2020; Reddi et al., 2021); we refer to the recent survey (Wang et al., 2021) for details. More generally, the framework of local SGD has been used to study federated optimization algorithms (Stich, 2019; Zhou & Cong, 2018; Haddadpour et al., 2019; Dieuleveut & Patel, 2019; Li et al., 2020; Khaled et al., 2020; Koloskova et al., 2020). Compared to these works, which study federated optimization algorithms in the smooth case, we tackle in our analysis the added challenge of nonsmoothness of the superquantile-based objective in both the general nonconvex and strongly convex cases.

The second line of work addressing heterogeneity involves designing new objective functions by modeling statistical heterogeneity and designing optimization algorithms. The AFL framework to minimize the worst-case error across all training clients and associated generalization bounds were given in Mohri et al. (2019). The concurrent work of Li et al. (2020) proposes the q-FFL framework whose objective is inspired by fair resource allocation to minimize the L^p norm of the per-client losses. Several related works were also published following the initial presentation of this work (Laguel et al., 2020). A federated optimization algorithm for AFL was proposed and its convergence was analyzed in Deng et al. (2020). Distributional robustness to affine shifts in the data was considered in

Reisizadeh et al. (2020) along with convergence guarantees. Finally, a classical risk measure, namely the entropic risk measure, was considered in Li et al. (2021). We note that no convergence guarantees are currently known for the stochastic optimization algorithms of Li et al. (2020). Furthermore, it is unclear if any of these algorithms can be implemented with differential privacy.

Differential privacy was introduced in Dwork et al. (2006a, 2006b) to formalize the loss of privacy of an individual user in releasing population-level aggregates. DP-FedAvg (McMahan et al., 2018), a differentially private variant of FedAvg, is also implemented in industrial systems (Ramaswamy et al., 2020). Recent contributions in this direction include differential privacy mechanism compatible with secure aggregation (Kairouz et al., 2021; Agarwal et al., 2021) and improving privacy-utility tradeoffs of federated learning with personalization (Jain et al., 2021; Bietti et al., 2022).

3 Problem setup

We begin this section by recalling the standard setup of federated learning in Sect. 3.1. We then describe the standard approach to federated learning and its associated optimization, FedAvg (McMahan et al., 2017) in Sect. 3.2. We then define the superquantile in Sect. 3.3.

3.1 Federated learning setup

Federated learning consists of heterogeneous clients who collaboratively train a machine learning model under the orchestration of a central server. The model is then deployed to all clients, including those not seen during training.

Let the vector $w \in \mathbb{R}^d$ denote the d model parameters. We assume that each client has a distribution q over some data space such that the data on the client is sampled i.i.d. from q. The loss incurred by the model $w \in \mathbb{R}^d$ on this client is $F(w;q) := \mathbb{E}_{\xi\sim q}[f(w;\xi)]$, where $f(w;\xi)$ is the chosen loss function, such as the logistic loss, on input-output pair ξ under the model w. The expectation above is assumed to be well-defined and finite. For a given distribution q, smaller values of $F(\cdot;q)$ denote a better fit of the model to the data.

There are n clients available for training. We number these clients as $1, \ldots, n$ and denote the distribution on training client i by q_i. We denote the loss on client i by $F_i(w) := F(w;q_i)$.

The goal of federated learning is to train a model w so that it achieves good performance when deployed on *each* test client, including those unseen during training. Owing to the statistical heterogeneity of federated learning, the distribution p of a specific test client could be different from the average distribution $(1/n)\sum_{i=1}^n q_i$ that the model is trained on.

Each federated learning method is characterized by an objective function and the federated optimization algorithm used to minimize it. It is not possible to achieve good performance on each client *simultaneously* with a single model w, as it would be a difficult multi-objective optimization problem. The usual approach is to combine the per-client losses into a scalar and minimize this objective. The choice of the objective function and optimization algorithm is primarily determined by the three key aspects of federated learning (Kairouz et al., 2021; Li et al., 2020):

(1) *Communication bottleneck*: The repeated exchange of massive models between the server and clients over resource-limited wireless networks makes communication a critical bottleneck. Therefore, training algorithms should be able to trade off more local computation for a lower communication cost.
(2) *Statistical heterogeneity*: The training distribution q_i and a specific test distribution p are likely to be different from each other. Therefore, a model which works well *on average* over all test clients might not work well on *each individual* test client.
(3) *Privacy*: The data on each client is highly privacy-sensitive. Federated learning is designed to protect data privacy since no user data is transferred to a data center. This privacy is enhanced by *secure aggregation* of model parameters, which refers to aggregating client updates such that no client update is directly revealed to any other client or the server. This is achieved by cryptographic protocols based on secure multiparty communication (Bonawitz et al., 2017).

3.2 Federated learning and the FedAvg algorithm

Analogous to the classical expectation-based objective function in the empirical risk minimization approach, the standard objective in federated learning is to minimize the average loss on the training clients

$$\min_{w\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} F_i(w) + \frac{\lambda}{2}\|w\|^2\,, \tag{3}$$

where $\lambda \geq 0$ is a regularization parameter. We will call this objective the *vanilla FL* objective.

The de facto standard training algorithm is FedAvg (McMahan et al., 2017). Each round of the algorithm consists of the following steps:

(a) The server samples a set S of m clients from $[n]$ and broadcasts the current model $w^{(t)}$ to these clients.
(b) Staring from $w_{i,0}^{(t)} = w^{(t)}$, each client $i \in S$ makes τ local gradient descent steps with a learning rate γ:

$$w_{i,k+1}^{(t)} = w_{i,k}^{(t)} - \gamma \nabla F_i(w_{i,k}^{(t)})\,.$$

In practice, one could also use local stochastic gradient steps, but we restrict ourselves to local full gradient steps for simplicity.
(c) The models from the selected clients are sent to the server and aggregated to update the server model

$$w^{(t+1)} = \frac{1}{m}\sum_{i\in S} w_{k,\tau}^{(t)}\,.$$

FedAvg addresses the communication bottleneck by using $\tau > 1$ local computation steps as opposed to $\tau = 1$ local steps in minibatch SGD. It also securely performs the averaging step (c) to enhance data privacy. However, the vanilla FL objective places a limit on how well statistical heterogeneity can be addressed. By minimizing the average training loss, the resulting model w can sacrifice performance on "difficult" clients to perform well on average. In other words, it is not guaranteed to perform well on *individual* test clients, whose

distribution p might be quite different from the average training distribution $(1/n)\sum_{i=1}^{n} q_i$. Our goal in this work is to design an objective function, different from the vanilla FL objective (3) to better handle statistical heterogeneity and the associated train-test mismatch. We also design a federated optimization algorithm similar to FedAvg to optimize it.

3.3 Summarizing the tail behavior with the superquantile

In this work, we consider clients with heterogeneous local data distributions $q_1, \ldots, q_n$. This data heterogeneity manifests itself as a spread over the losses $F_1(w), \ldots, F_n(w)$ for any w. In particular, some clients might suffer large losses due to their distributions being far from the average population distribution. Our goal is to improve the loss (and hence, predictive performance) on such tail clients whose loss is worse than average. In other words, we are concerned with the right tail statistics of the empirical distribution over the losses $F_1(w), \ldots, F_n(w)$.

A natural summary of the right tail of a random variable Z is its high quantiles. Recall that the $(1-\theta)$-quantile $Q_\theta(Z)$ of a real-valued random variable Z is defined as

$$Q_\theta(Z) := \inf\{\eta \in \mathbb{R} \ : \ \mathbb{P}(Z > \eta) \leq \theta\}.$$

Unfortunately, the quantile function of discrete random variables such as the empirical loss distribution is piecewise constant and is not amenable to gradient-based optimization. A better-behaved tail summary in this regard is the superquantile, also known as the conditional value at risk (CVaR) (Rockafellar & Uryasev, 2000, 2002).

The superquantile $\mathbb{S}_\theta(Z)$ of a random variable Z is defined as the average of all quantiles greater than the $(1-\theta)$-quantile:

$$\mathbb{S}_\theta(Z) = \frac{1}{\theta}\int_0^\theta Q_\alpha(Z)\,\mathrm{d}\alpha\,. \tag{4}$$

For continuous random variables, we have the equivalence $\mathbb{S}_\theta(Z) = \mathbb{E}[Z \mid Z > Q_\theta(Z)]$ of the superquantile as the *tail mean*, as illustrated in Fig. 2. Owing to this interpretation, we refer to the parameter θ as the *tail threshold*.

Central to our development is the dual expression of the superquantile (Föllmer & Schied, 2002):

$$\begin{aligned} \mathbb{S}_\theta(a_1, \ldots, a_n) &= \max_{\pi \in \mathcal{P}_\theta} \pi^\top a\,, \\ \text{where} \quad \mathcal{P}_\theta &= \{\pi \in \Delta^{n-1} \ : \ \pi_i \leq (\theta n)^{-1} \ \text{ for all } i\}\,. \end{aligned} \tag{5}$$

Here, $\mathbb{S}_\theta(a_1, \ldots, a_n)$ denotes the $(1-\theta)$-superquantile of the empirical measure $(1/n)\sum_{i=1}^{n}\delta_{a_i}$ and Δ^{n-1} is the probability simplex in $\mathbb{R}^n$. The discrete superquantile is thus the support function of the polytope $\mathcal{P}_\theta$, which is illustrated in Fig. 2. Not only is the discrete superquantile a continuous function of its inputs (unlike the quantile function), but it is also convex as it is the maximum of a family of linear functions in the expression (5).

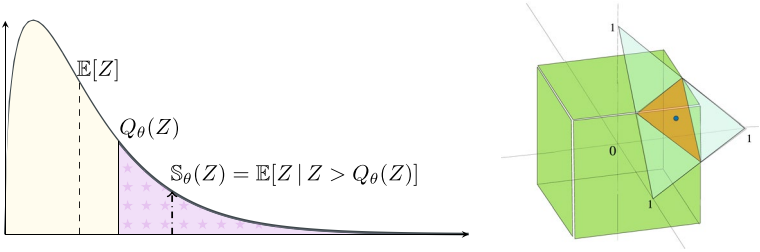

Fig. 2 **Left**: $(1-\theta)$-quantile $Q_\theta(Z)$ and superquantile $\mathbb{S}_\theta(Z)$ of a continuous r.v. Z. **Right**: The set of feasible mixture weights $\pi = (\pi_1, \pi_2, \pi_3) \in \mathcal{P}_\theta$ in the dual formulation (5) is given by the intersection of the box constraints $0 \le \pi_i \le (3\theta)^{-1}$ for $i = 1, 2, 3$, with the simplex constraint $\pi_1 + \pi_2 + \pi_3 = 1$

4 Handling heterogeneity with Δ-FL

In this section, we introduce the Δ-FL framework in Sect. 4.1 and propose an algorithm to optimize in the federated setting in Sect. 4.2.

4.1 The Δ-FL framework

The Δ-FL framework aims to improve the performance of the tail clients by minimizing the superquantile of the loss distribution. Given a discretization $\{\theta_1, \dots, \theta_r\}$ of $(0, 1]$, Δ-FL maintains r models $w_1, \dots, w_r$, one for each tail threshold θ_j. We allow each test client to select the best model $w \in \{w_1, \dots, w_r\}$, according to its local data. Recall the schematic in Fig. 1 for an illustration.

For a given tail threshold θ, we propose to minimize the $(1-\theta)$-superquantile of the distributions of losses:

$$\min_{w \in \mathbb{R}^d} \left[F_\theta(w) := \mathbb{S}_\theta\big(F_1(w), \dots, F_n(w)\big) + \frac{\lambda}{2}\|w\|^2 \right]. \tag{6}$$

The objective (6) focuses on poor-performing clients — specifically those with performance worse than the $(1-\theta)$-quantile of the distribution of losses $(F_1(w), \dots, F_n(w))$. In contrast, the vanilla FL objective optimizes $(1/n)\sum_{i=1}^n F_i(w) + \lambda/2\|w\|^2$, which is $\lim_{\theta \to 1} F_\theta(w)$; this equally weights all clients involved in training. At the other extreme $\theta \to 0$, we recover the worst-case loss over all clients.

Distributionally Robust Interpretation We have the following dual characterization of Δ-FL as a distributionally robust learning objective, as a consequence of the dual representation (5) of the superquantile.

Property 1 *The Δ-FL objective (6) can also be written as*

$$\begin{aligned} F_\theta(w) &= \max_{\pi \in \mathcal{P}_\theta} \sum_{i=1}^n \pi_i F_i(w), \\ \text{where,} \quad \mathcal{P}_\theta &:= \left\{ \pi \in \Delta^{n-1} \;:\; \pi_i \le (n\theta)^{-1} \text{ for all } i \in [n] \right\}. \end{aligned} \tag{7}$$

This reformulation shows that Δ-FL can be interpreted as a distributionally robust variant of the vanilla FL objective: since $\sum_{i=1}^n \pi_i F_i(w) = F(w; p_\pi)$ is loss of w on the mixture

$p_\pi = \sum_{i=1}^n \pi_i q_i$ of the training distributions $q_1, \ldots, q_n$, we get that Δ-FL aims to minimize the worst-case loss over all mixtures p_π subject to the constraint that $\pi_i \leq (n\theta)^{-1}$.

This formulation also reveals two important properties of the Δ-FL objective. First, we note that the objective F_θ, as a max function. is convex whenever the losses F_i are convex. Second, it is a non-smooth function, with the non-smoothness stemming from the maximum over the polytope $\mathcal{P}_\theta$ (cf. Fig. 2). These two properties will play important role in the convergence analysis of our federated algorithm in Sect. 5.1.

Algorithm 1 The Δ-FL Algorithm

Input: Initial iterate $w^{(0)}$, number of communication rounds T, number of clients per round m, number of local updates τ, local step size γ

1: **for** $t = 0, 1, \ldots, T-1$ **do**
2: Sample m clients from $[n]$ without replacement in S
3: Estimate the $(1-\theta)$-quantile of $F_i(w^{(t)})$ for $i \in S$ with distributed differential privacy (Algorithm 2); call this $Q^{(t)}$
4: **for** each selected client $i \in S$ in parallel **do**
5: Set $\tilde{\pi}_i^{(t)} = \mathbb{I}\left(F_i(w^{(t)}) \geq Q^{(t)}\right)$
6: Initialize $w_{k,0}^{(t)} = w^{(t)}$
7: **for** $k = 0, \ldots, \tau - 1$ **do**
8: $w_{i,k+1}^{(t)} = (1 - \gamma\lambda) w_{i,k}^{(t)} - \gamma \nabla F_i(w_{i,k}^{(t)})$
9: **end for**
10: **end for**
11: $w^{(t+1)} = \sum_{i \in S} \tilde{\pi}_i^{(t)} w_{i,\tau}^{(t)} / \sum_{i \in S} \tilde{\pi}_i^{(t)}$
12: **end for**
13: **return** w_T

4.2 Federated optimization for Δ-FL

We now propose a federated optimization algorithm for the Δ-FL objective (6). While there could be many approaches to optimizing (6), we consider algorithms similar to FedAvg for their ability to avoid communication bottlenecks and preserve the privacy of user data. Owing to the tail mean interpretation of the superquantile (Fig. 2), a natural algorithm to minimize it first evaluates the loss on all the clients and only performs gradient updates on those clients in the tail above the $(1-\theta)$-quantile. However, since a practical algorithm cannot assume that all the clients are available at a given time, we perform the same operation on a subsample of clients.

The optimization algorithm for the Δ-FL objective (6) is given in Algorithm 1. It has the following four steps:

(a) *Model Broadcast* (line 2): The server samples a set S of m clients from $[n]$ and sends the current model $w^{(t)}$.
(b) *Quantile Computation and Reweighting* (lines 3 and 5): Selected clients $i \in S$ and the server collaborate to estimate the $(1-\theta)$-quantile of the losses $F_i(w^{(t)})$ with differential privacy. The clients then update their weights to be zero if their loss is smaller than the estimated quantile and leave them unchanged otherwise. This ensures that model updates are only aggregated from the tail clients; cf. Fig. 2.

(c) *Local Updates* (loop of line 7): Staring from $w_{k,0}^{(t)} = w^{(t)}$, each client $i \in S$ makes τ local gradient or stochastic gradient descent steps with a learning rate γ.
(d) *Update Aggregation* (line 11): The models from the selected clients are sent to the server and aggregated to update the server model, with weights from line 5).

Compared to FedAvg, Δ-FL has the additional step of computing the quantile and new weights $\tilde{\pi}_i^{(t)}$ for each selected client $i \in S$ in lines 3 and 5. Let us consider Δ-FL in relation to the three key aspects of federated learning we introduced in Sect. 3.1.

(1) *Communication Bottleneck*: Identical to FedAvg, Δ-FL algorithm performs multiple computation rounds per communication round.
(2) *Statistical Heterogeneity*: The Δ-FL objective is designed to optimize the tail mean of the per-client loss distribution as formalized by the superquantile. The vanilla FL objective, in contrast, is oblivious to performance disparities across clients.
(3) *Privacy*: Identical to FedAvg, Δ-FL does not require any data transfer, and the aggregation of line 11 can be securely performed using secure multiparty communication. The extra step of quantile computation is also performed with distributed differential privacy, as we describe next.

Quantile Estimation with Distributed Differential Privacy The naïve way to compute the quantile of the per-client losses in line 3 of Algorithm 1 is to have the clients send their losses to the server. To avoid the privacy risk of leakage of information about the clients to the server, we compute the quantile with distributed differential privacy (Kairouz et al., 2021) using the discrete Gaussian mechanism (Canonne et al., 2020). The key idea behind differential privacy (Dwork et al., 2006b, 2016) is to ensure that the addition or removal of the data from one client does not lead to a substantial change in the output of an algorithm. A significant difference in the output would give a privacy adversary enough signal to learn about the client who was added or removed.

Algorithm 2 Quantile Computation with Distributed Differential Privacy

Input: Ring size M, set S of $m = |S|$ clients where each client i has a scalar $\ell_i \in [0, B]$, target quantile $1-\theta \in (0,1)$, discretization $l_0, l_1, \ldots, l_b$ of $[0, B]$, variance proxy σ^2, scaling factor $c \in \mathbb{Z}_+$

1: Each client i computes a hierarchical histogram $x_i(r,j) = \mathbb{I}\left(l_{2^r(j-1)+1} \le F_i(w) < l_{2^r j}\right)$ for $j = 1, \ldots, b/2^r$ and $r = 0, \ldots, \log_2 b - 1$
2: Each client i samples $\xi_i(r,j) \sim \mathcal{N}_{\mathbb{Z}}(0, \sigma^2)$ i.i.d. and sets $\tilde{x}_i(r,j) = \left(c x_i(r,j) + \xi_i(r,j)\right) \mod M$ for each r, j
3: Compute $s = \left(\sum_{i \in S} \tilde{x}_i\right) \mod M$ securely
4: Set hierarchical histogram $\hat{h} = s/c$ and define for $j \in [b]$ its cumulative sum $\hat{H}(j) = \sum_{(r,o) \in P_j} \hat{h}(r,o)$ using a maximal dyadic partition P_j of $[1, j]$
5: **return** Quantile estimate $l_{j_\theta^*(\hat{h})}$ corresponding to index $j_\theta^*(\hat{h})$; cf. Eq. (8)

Distributed differential privacy simulates a trusted central aggregator by using a secure summation oracle (Bonawitz et al., 2017), which enables the computation of summations $\sum_{i \in S} v_i$ where $v_i \in \mathbb{R}^d$ is a privacy-sensitive vector residing with client i. Practical

implementations of such algorithms are based on cryptographic techniques such as secure multiparty computation (Evans et al., 2018), which requires each component of the vectors v_i to be discretized to the ring $\mathbb{Z}_M$ of integers modulo M. We abstract out the details of the secure summation oracle and only require that it returns the sum $\left(\sum_{i\in S} x_v\right) \mod M$ without revealing any further information to a privacy adversary.

We assume that the losses are bounded as $F_i(w) \in [0, B]$ for each $i \in S$, and that we are given b bin edges $0 \le l_0 < l_1 < \cdots < l_b = B$. We aim to construct a hierarchical histogram h that maintains the number of clients not only in every single bin but also in groups of bins organized as a binary tree. Concretely, $h(r, j)$ maintains the number of clients whose losses lie between the bin edges $l_{2^r(j-1)+1}$ and $l_{2^r j}$ for index $j = 1, \ldots, b/2^r$ and level $r = 0, \ldots, \log_2 b - 1$.[1] The lower levels $r = 0$ and $r = 1$ correspond respectively to individual bins and pairs of bins, while the topmost level $r = \log_2 b - 1$ refers to two groups: the first $b/2$ bins and the last $b/2$ bins. We skip the topmost level in the tree because the count at this node is the publicly known number $m = |S|$ of clients. The hierarchical histogram method, also known as tree aggregation, is a classical technique to answer range queries and in cumulative distribution estimation (Hay et al., 2010; Dwork et al., 2010; Chan et al., 2011; Smith et al., 2017).

Our algorithm is given in Algorithm 2. Each client i first computes its local hierarchical histogram x_i as

$$x_i(r,j) = \mathbb{I}\left(l_{2^r(j-1)+1} \le F_i(w) < l_{2^r j}\right),$$

such that the overall hierarchical histogram can be obtained as $h = \sum_{i\in S} x_i$. To enforce differential privacy, each client then adds random discrete Gaussian[2] noise $\xi_i \sim \mathcal{N}_{\mathbb{Z}}(0, \sigma^2 I)$ with scale parameter σ^2 and of appropriate dimension. These noisy $\tilde{x}_i$'s are summed up using a secure summation oracle so that the server receives an approximate hierarchical histogram $\hat{h}$ which approximates the true histogram $h = \sum_{i\in S} x_i$. With slight abuse of notation, we still refer to $\hat{h}$ as a hierarchical histogram, although it could have negative entries and could be inconsistent, i.e., the count $\hat{h}(r, j)$ at a node might not equal the sum $\hat{h}(r-1, 2j-1) + \hat{h}(r-1, 2j)$ of counts at its children nodes.

The final step is to define and return an appropriate notion of a $(1-\theta)$-quantile of the approximate histogram $\hat{h}$. A non-negative hierarchical histogram h can be viewed as a random variable Z with (scaled) cumulative distribution function $H(j) = m\,\mathbb{P}\left(Z \le l_j\right) = h(0,1) + \ldots + h(0,j)$, from which we can estimate the quantile. We can obtain a greater utility under differential privacy by expressing the cumulative distribution function $H(j)$ of this random variable Z by using nodes higher up in the tree. Concretely, using a maximal dyadic partition P_j of the range $[1, j]$, we have $H(j) = \sum_{(r,o)\in P_j} h(r, o)$ from summing up $|P_j| \le \log_2 b$ terms. For instance, the dyadic partition for $j = 15$ is $P_{15} = [1, 8] \cup [9, 12] \cup [13, 14] \cup [15]$, where the counts of each range on the right side can be obtained from an intermediate node in the hierarchical histogram h.

With this definition of the cumulative mass $H(j)$, we define $(1-\theta)$-quantile of the hierarchical histogram h as the quantile function of this induced random variable Z:

$$Q_\theta(H) := Q_\theta(Z) = \min_{j\in[b]} \left\{ l_j \;:\; H(j) > (1-\theta)m \right\}.$$

[1] We assume for simplicity that b is a power of 2 so that $\log_2 b$ is an integer.

[2] See Appendix B for a formal definition.

Similarly, for approximate hierarchical histograms $\hat{h}$ that are inconsistent and allow for negative values, we define the cumulative function $\hat{H}(j) = \sum_{(r,o)\in P_j} \hat{h}(r,o)$ from a maximal dyadic partition P_j of $[1, j]$. As an estimate of the quantile, we return the bin edge l_j such that the estimated cumulative mass $\hat{H}(j)$ is as close to $1-\theta$ as possible:

$$Q_\theta(\hat{h}) := l_{j^*_\theta(\hat{h})} \quad \text{where} \quad j^*_\theta(\hat{h}) = \arg\min_{j\in[b]} \left|\hat{H}(j) - (1-\theta)m\right|. \tag{8}$$

5 Theoretical analysis

In this section, we analyze the convergence analysis of Δ-FL (Sect. 5.1) and study the differential privacy properties of the quantile computation (Sect. 5.2).

5.1 Convergence analysis

We study the convergence of Algorithm 1 with respect to the objective (6) in two cases: (i) the general non-convex case, and (ii) when each $F_i(w)$ is convex.

Assumption We make some assumptions on the per-client losses F_i, which are assumed to hold throughout this section. For each client $i \in [n]$, the objective F_i is

(a) B-bounded, i.e., $0 \le F_i(w) \le B$ for all $w \in \mathbb{R}^d$,
(b) G-Lipschitz, i.e., $|F_i(w) - F_i(w')| \le G\|w - w'\|$ for all $w, w' \in \mathbb{R}^d$, and,
(c) L-smooth, i.e., F_i is continuously differentiable and its gradient ∇F_i is L-Lipschitz.

Algorithm 3 The Δ-FL Algorithm with Exact Reweighting

Input: Same as Algorithm 1
1: **for** $t = 0, 1, \ldots, T-1$ **do**
2: Sample m clients from $[n]$ without replacement in S
3: Compute $\pi^{(t)} = \arg\max_{\pi\in\mathcal{P}_{\theta,S}} \sum_{i\in S} \pi_i F_i(w^{(t)})$
4: **for** each selected client $i \in S$ in parallel **do**
5: Initialize $w^{(t)}_{i,0} = w^{(t)}$
6: **for** $k = 0, \ldots, \tau-1$ **do**
7: $w^{(t)}_{i,k+1} = (1-\gamma\lambda)w^{(t)}_{i,k} - \gamma\nabla F_i(w^{(t)}_{i,k})$
8: **end for**
9: **end for**
10: $w^{(t+1)} = \sum_{i\in S} \pi^{(t)}_i w^{(t)}_{i,\tau}$
11: **end for**
12: **return** w_T

Equivalent Algorithm. Algorithm 1 is not amenable to theoretical analysis as it is stated because the quantile function of discrete random variables computed in line 3 is piecewise constant and discontinuous. To overcome this obstacle, we introduce a near-equivalent algorithm in Algorithm 3, which replaces the reweighting step of

Algorithm 1 (lines 3 and 5) with the ideal reweighting suggested by the dual representation of (7).

Let us start with the case of $S = [n]$. Our first observation shows that the weights $\pi^{(t)}$ that attain the maximum over π in the objective (7) can be used to construct a subgradient of F_θ in the general nonconvex case — this will eventually allow us to derive convergence guarantees.

Property 2 *Fix a* $w \in \mathbb{R}^d$ *and let* $\pi^\star \in \arg\max_{\pi \in \mathcal{P}_\theta} \sum_{i=1}^n \pi_i F_i(w)$. *Then, we have,*

$$\sum_{i=1}^n \pi_i^\star F_i(w) + \lambda w \in \partial F_\theta(w)\,,$$

where $\partial F_\theta(w)$ *denotes the regular subdifferential of* F_θ.

Proof Let $h_\theta(a) := \max_{\pi \in \mathcal{P}_\theta} \pi^\top a$ denote the support function of the polytope $\mathcal{P}_\theta$, and let $g_n(w) = (F_1(w), \dots, F_n(w))$ denote the concatenation of the losses into a vector. Then, $F_\theta(w) = h_\theta \circ g_n(w) + (\lambda/2)\|w\|^2$. Since h_θ is convex, we get that its (convex) subdifferential (e.g., Hiriart-Urruty & Lemaréchal 1996, Cor. 4.4.4) is

$$\partial h_\theta(a) = \arg\max_{\pi \in \mathcal{P}_\theta} \pi^\top a\,.$$

Since g_n is smooth and h_θ is convex with full domain, we obtain the regular subdifferential of $h_\theta \circ g_n$ by the chain rule (cf. Rockafellar & Wets (2009, Thm. 10.6)) as

$$\partial(h_\theta \circ g_n) = \nabla g_n(w) \partial h_\theta\big(g_n(w)\big)\,,$$

where $\nabla g_n(w) \in \mathbb{R}^{d \times n}$ is the transpose of the Jacobian matrix of g_n. We can handle the regularization by absorbing it into the superquantile by defining $\tilde{F}_i(w) = F_i(w) + (\lambda/2)\|w\|^2$. □

Algorithm 3 extends this intuition to the setting where only a subsample $S \subset [n]$ of clients are available in each round. We define the counterpart of the constraint set $\mathcal{P}_\theta$ from (7) defined on a subset $S \subset [n]$ of m clients as:

$$\mathcal{P}_{\theta,S} = \left\{ \pi \in \Delta^{|S|-1} \;:\; \pi_i \le \frac{1}{\theta m},\ \text{for } i \in S \right\}, \tag{9}$$

where we denote $(\pi_i)_{i \in S} \in \mathbb{R}^{|S|}$ by π with slight abuse of notation. With this notation, Algorithm 3 computes the new weights of the clients as

$$\pi^{(t)} = \arg\max_{\pi \in \mathcal{P}_{\theta,S}} \sum_{i \in S} \pi_i F_i(w^{(t)})\,.$$

We now analyze how close Algorithm 3 is to Algorithm 1. Let $Z(w)$ be a discrete random variable which takes the value $F_i(w)$ with probability $1/n$ for $i = 1, \dots, n$, and let $Q_\theta(Z(w))$ denote its $(1-\theta)$-quantile. The weights $\hat{\pi} \in \Delta^{n-1}$ considered in Algorithm 1 (assuming that $Q^{(t)}$ is the exact quantile of $\{F_i(w^{(t)}) \;:\; i \in S\}$) are given by a hard-thresholding based on whether $F_i(w)$ is larger than its $(1-\theta)$-quantile:

$$\tilde{\pi}_i = \mathbb{I}\big(F_i(w) \geq Q_\theta(Z(w))\big)\,, \quad \text{and,} \quad \hat{\pi}_i = \frac{\tilde{\pi}_i}{\sum_{i'=1}^n \tilde{\pi}_{i'}}. \tag{10}$$

The objective defined by these weights is $\hat{F}_\theta(w) = \sum_{i=1}^n \hat{\pi}_i F_i(w) + (\lambda/2)\|w\|^2$. The next proposition shows that $\hat{F}_\theta(w) = F_\theta(w)$ under certain conditions, or is a close approximation, in general.

Proposition 3 *Assume $F_1(w) < \cdots < F_n(w)$ and let $i^\star = \lceil \theta n \rceil$. Then, we have,*

(a) $\pi^\star = \arg\max_{\pi \in \mathcal{P}_\theta} \sum_{i=1}^n \pi_i F_i(w)$ *is unique,*
(b) $Q_\theta(Z(w)) = F_{i^\star}(w)$,
(c) *if θn is an integer, then $\hat{\pi} = \pi^\star$ so that $\hat{F}_\theta(w) = F_\theta(w)$, and,*
(d) *if θn is not an integer, then*

$$0 \leq F_\theta(w) - \hat{F}_\theta(w) \leq \frac{B}{\theta n}\,.$$

Proof We assume w.l.o.g. that $\lambda = 0$. We apply the property that the superquantile is a tail mean (cf. Fig. 2) for discrete random variables shown by Rockafellar & Uryasev (2002, Proposition 8) to get

$$F_\theta(w) = \frac{1}{\theta n} \sum_{i=i^\star+1}^n F_i(w) + \left(1 - \frac{\lfloor \theta n \rfloor}{\theta n}\right) F_{i^\star}(w)\,.$$

Comparing with dual representation (7), this gives a closed-form expression for $\pi^\star$, which is unique because $F_{i^\star-1}(w) < F_{i^\star}(w) < F_{i^\star+1}(w)$. For (b), note that $Q_\theta(Z(w)) = \inf\{\eta \in \mathbb{R} \,:\, \mathbb{P}(Z(w) > \eta) \leq \theta\}$ equals $F_{i^\star}(w)$ by definition of $i^\star$. Therefore, if θn is an integer, $\pi^\star$ coincides exactly with $\hat{\pi}$.

When θn is not an integer, we have

$$\hat{F}_\theta(w) = \frac{1}{n - i^\star + 1} \sum_{i=i^\star}^n F_i(w)\,.$$

The bound on $\hat{F}_\theta(w) - F_\theta(w)$ follows from elementary manipulations together with $0 \leq F_i(w) \leq B$. □

In our context where we sample m clients per round, Proposition 3 holds for each round. In particular, part (c) of Proposition 3 states that when θm is an integer, the weights $\pi^\star$ computed as an exact argmax in Algorithm 3 are identical to the weights $\hat{\pi}$ in Algorithm 1 where line 5 exactly computes the quantile of the per-client losses. We record another consequence of Proposition 3, namely, that the reweighting $\pi^{(t)}$ is sparse.

Remark 1 Proposition 3 shows that Δ-FL 's reweighting $\pi^{(t)}$ (line 3 of Algorithm 3) is sparse. That is, $\pi_i^{(t)}$ is non-zero only for exactly $\lceil \theta m \rceil$ clients with the largest losses.

Bias due to Partial Participation. Note that the dual representation (7) is the maximum over all distributions in $\mathcal{P}_\theta$, but Algorithm 3 and Algorithm 1 only maximize the

weights over a set S of m clients in each round (line 3). Therefore, the updates performed by Algorithm 3 are not unbiased. To formalize this, define the objective

$$\overline{F}_\theta(w) := \mathbb{E}_{S\sim U_m}\big[F_{\theta,S}(w)\big], \quad \text{where } F_{\theta,S}(w) = \max_{\pi\in\mathcal{P}_{\theta,S}} \sum_{i\in S} \pi_i F_i(w) + \frac{\lambda}{2}\|w\|^2$$

is the analogue of (7) defined on a sample $S \subset [n]$ of clients, and U_m is the uniform distribution over subsets of $[n]$ of size m. In each step, Algorithm 3 approximates the subgradients of $F_{\theta,S}$. Indeed, Property 2 gives

$$\sum_{i\in S} \pi_i^{(t)} F_i(w^{(t)}) + \lambda w^{(t)} \in \partial F_{\theta,S}(w^{(t)})\,. \tag{11}$$

In expectation, Algorithm 3 therefore takes subgradient steps for $\overline{F}_\theta$ — this introduces a bias when compared to the original F_θ that we would like to optimize. Fortunately, this bias can be bounded as shown by Levy et al. (2020, Prop. 1):

$$\sup_{w\in\mathbb{R}^d} \left|\overline{F}_\theta(w) - F_\theta(w)\right| \le \frac{B}{\sqrt{\theta m}}\,. \tag{12}$$

Our analysis strategy will be to study the convergence (near-stationarity or near-optimality) in terms of the objective $\overline{F}_\theta$ which Algorithm 3 actually minimizes, and then translate that to a convergence result on the original objective F_θ using the bound (12).

Convergence: Nonconvex Case We start with the convergence analysis in the nonconvex case with no regularization (i.e., $\lambda = 0$). Since $\overline{F}_\theta$ is nonsmooth and nonconvex, we state the convergence guarantee in terms of the Moreau envelope of $\overline{F}_\theta$ (Hiriart-Urruty & Lemaréchal, 1996) following the idea of Drusvyatskiy and Paquette (2019), Davis and Drusvyatskiy (2019). Given a parameter $\mu > 0$, we define the Moreau envelope of $\overline{F}_\theta$ as

$$\overline{\Phi}_\theta^\mu(w) = \inf_{z\in\mathbb{R}^d} \left\{\overline{F}_\theta(z) + \frac{\mu}{2}\|w - z\|^2\right\}. \tag{13}$$

The Moreau envelope satisfies several remarkable properties for $\mu > L$, see for example, Drusvyatskiy & Paquette (2019, Lemma 4.3). First, it is well-defined, and the infimum on the right-hand side admits a unique minimizer, called the proximal point of w, and denoted $\mathrm{prox}_{\overline{F}_\theta/\mu}(w)$. Second, the Moreau envelope is continuously differentiable with $\nabla\overline{\Phi}_\theta^\mu(w) = \mu(w - \mathrm{prox}_{\overline{F}_\theta/\mu}(w))$. Finally, the stationary points of $\overline{\Phi}_\theta^\mu$ and $\overline{F}_\theta$ coincide. Interestingly, the bound $\left\|\nabla\overline{\Phi}_\theta^\mu(w)\right\| \le \varepsilon$ directly implies a near-stationarity on $\overline{F}_\theta$, and hence the original F_θ, in the following variational sense: the proximal point $z = \mathrm{prox}_{\overline{F}_\theta/\mu}(w)$ satisfies the following, cf. Drusvyatskiy & Paquette (2019, Sect. 4.1):

(a) z is close to w; that is, $\|z - w\| \le \varepsilon/\mu$,
(b) z is nearly stationary on $\overline{F}_\theta$; that is $\mathrm{dist}\big(0, \partial\overline{F}_\theta(z)\big) \le \varepsilon$, where $\partial\overline{F}_\theta$ refers to the regular subdifferential, and,
(c) $\overline{F}_\theta$ is uniformly close to F_θ as per (12).

Thus, we state the convergence guarantee of our algorithm in the nonsmooth nonconvex case in terms of $\overline{\Phi}_\theta^\mu$ (although it never appears in the algorithm).

Theorem 4 *Let the number of rounds T be fixed and set $\mu = 2L$. Denote $\Delta F_0 = F_\theta(w^{(0)}) - \inf \overline{F}_\theta$. Let $\hat{w}$ denote a uniformly random sample from the sequence $\left(w^{(0)}, \dots, w^{(T-1)}\right)$ produced by Algorithm 3. Then, there exists a learning rate γ depending on the number of rounds T and problem parameters $\tau, L, G, \Delta F_0$ such that*

$$\mathbb{E}\left\|\nabla \overline{\Phi}_\theta^\mu(\hat{w})\right\|^2 \le \sqrt{\frac{\Delta_0 L G^2}{T}} + (1-\tau^{-1})^{1/3}\left(\frac{\Delta_0 L G}{T}\right)^{2/3} + \frac{\Delta_0 L}{T}\,.$$

Proof Sketch Let $z^{(t)} = \operatorname{prox}_{\overline{F}_\theta/\mu}(w^{(t)})$ be the proximal point of $w^{(t)}$. We expand out the recursion $w^{(t+1)} = w^{(t)} - \gamma \sum_{i\in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \nabla F_i(w_{i,k}^{(t)})$ to get

$$\begin{aligned}
\overline{\Phi}_\theta^\mu(w^{(t+1)}) &\le \overline{F}_\theta(z^{(t)}) + \frac{\mu}{2}\left\|z^{(t)} - w^{(t+1)}\right\|^2 \\
&= \overline{F}_\theta(z^{(t)}) + \frac{\mu}{2}\left\|z^{(t)} - w^{(t)}\right\|^2 \\
&\quad + \mu\gamma\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\rangle \\
&\quad + \frac{\mu\gamma^2}{2}\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\|^2 \\
&= \overline{\Phi}_\theta^\mu(w^{(t)}) + \mathcal{T}_1 + \mathcal{T}_2\,.
\end{aligned}$$

The term $\mathcal{T}_1$ which carries a $O(\gamma)$-coefficient controls the convergence rate while $\mathcal{T}_2$ carries a $O(\gamma^2)$-coefficient and is a noise term. The latter can be controlled by making the learning rate small. We can handle the first term $\mathcal{T}_1$ by leveraging a property of $F_{\theta,S}$ known as *weak convexity*, meaning that adding a quadratic makes it convex. In particular, $F_{\theta,S} + (L/2)\|\cdot\|^2$ is convex, so that

$$\begin{aligned}
\mathcal{T}_1' &:= \mu\tau\gamma\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\nabla F_i(w^{(t)})\right\rangle \\
&\le \mu\tau\gamma\left(F_{\theta,S}(z^{(t)}) - F_{\theta,S}(w^{(t)}) + \frac{L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2\right),
\end{aligned}$$

where we used (11) to construct a subgradient of $F_{\theta,S}$. This term $\mathcal{T}_1'$ is the result of a single step with learning rate $\tau\gamma$ rather than τ local steps with learning rate γ. The difference $\mathcal{T}_1' - \mathcal{T}_1$ is the effect of the drift induced by multiple local steps, which we will handle later. We take an expectation with respect to the sampling S of clients (i.e., conditioned on $\mathcal{F}^{(t)} = \sigma(w^{(t)})$, the σ-algebra generated by $w^{(t)}$). Since $z^{(t)}$ is independent of S (i.e., $z^{(t)}$ is $\mathcal{F}^{(t)}$-measurable), we get $\overline{F}_\theta$ on the right-hand side. Next, we use that $z^{(t)}$ minimizes the strongly convex right hand side of (13) to get

$$\mathbb{E}_t[\mathcal{T}_1'] \le -\mu\tau\gamma(\mu - L)\left\|z^{(t)} - w^{(t)}\right\|^2 = -\frac{\tau\gamma(\mu-L)}{\mu}\left\|\nabla\overline{\Phi}_\theta^\mu(w^{(t)})\right\|^2.$$

Next, we bound the effect of the drift using the Cauchy-Schwarz inequality and the smoothness of F_i's as

$$\begin{aligned}\mathbb{E}_t\left|\mathcal{T}_1 - \mathcal{T}_1'\right| &= \mu\gamma\,\mathbb{E}_t\left|\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\left(\nabla F_i(w_{i,k}^{(t)}) - \nabla F_i(w^{(t)})\right)\right\rangle\right| \\ &\le \frac{\mu\tau\gamma(\mu-L)}{2}\|z^{(t)} - w^{(t)}\|^2 + \frac{\mu\gamma L^2}{2(\mu-L)}\,\mathbb{E}_t\left[\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\|w_{i,k}^{(t)} - w^{(t)}\|^2\right] \\ &\le \frac{\tau\gamma(\mu-L)}{2\mu}\left\|\nabla\overline{\Phi}_\theta^{\mu}(w^{(t)})\right\|^2 + O(\gamma^3)\,,\end{aligned}$$

where we bound the client drift $d^{(t)} = \mathbb{E}_t\left[\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\|w_{i,k}^{(t)} - w^{(t)}\|^2\right] = O(\gamma^2)$ using standard techniques. We plug in $\mu = 2L$ to get a bound on $\mathcal{T}_1'$ in terms of $\left\|\nabla\overline{\Phi}_\theta^{\mu}(w^{(t)})\right\|^2$. A standard argument to handle the noise term $\mathcal{T}_2 \le O(\gamma^2)$ and telescoping the resulting inequality over $t = 0, \ldots, T-1$ completes the proof. The full details are given in Sect. A.2. □

Convergence: Convex Case We consider the convergence of function values in the case where each F_i is convex. Owing to the non-smoothness of F_θ and $\overline{F}_\theta$, we consider the following smoothed version of the objective in (7) and the corresponding modification to Algorithm 3. First, define the Kullback-Leibler (KL) divergence between $\pi \in \Delta^{|S|-1}$ and the uniform distribution $(1/|S|, \ldots, 1/|S|)$ over $S \subset [n]$ as

$$D_S(\pi) = \sum_{i\in S}\pi_i \log(\pi_i\,|S|)\,.$$

We simply write $D(\pi)$ when $S = [n]$. Inspired by Nesterov (2005), Beck and Teboulle (2012), Devolder et al. (2014), we define the smooth counterpart to (7) as

$$F_\theta^{\nu}(w) = \max_{\pi\in\mathcal{P}_\theta}\left\{\sum_{i=1}^{n}\pi_i F_i(w) - \nu D(\pi)\right\} + \frac{\lambda}{2}\|w\|^2\,, \tag{14}$$

where $\nu > 0$ is a fixed smoothing parameter. We have that $|F_\theta^{\nu}(w) - F_\theta(w)| \le 2\nu\log n$. Finally, we modify line 3 of Algorithm 3 to handle F_θ^{ν} rather than F_θ as

$$\pi^{(t)} = \arg\max_{\pi\in\mathcal{P}_{\theta,S}}\left\{\sum_{i\in S}\pi_i F_i(w^{(t)}) - \nu D_S(\pi)\right\}. \tag{15}$$

Theorem 5 *Suppose each function F_i is convex and $0 < \lambda < L$. Define the condition number $\kappa = (L+\lambda)/\lambda$ and fix a time horizon $T \ge 16\kappa^{3/2}$. Consider the sequence $(w^{(t)})_{t=0}^{T}$ of iterates produced by the Algorithm 3 with line 3 replaced by (15). Define the averaged iterate*

$$\overline{w}^{(t)} = \frac{\sum_{j=0}^{t}\beta_j w^{(i)}}{\sum_{i=0}^{t}\beta_j}, \quad \text{where} \quad \beta_j = \left(1 - \frac{\gamma\lambda\tau}{2}\right)^{-(1+j)},$$

and $w^\star = \arg\min_{w\in\mathbb{R}^d} F_\theta(w)$. Then, there exist learning rate γ and smoothing parameter ν depending on the number of communication rounds T as well as problem parameters $\tau, G, \lambda, L, \|w^{(0)} - w^\star\|^2, \theta, m$, such that the iterate $\overline{w}^{(T)}$ satisfies the bound

$$\mathbb{E}F_\theta(\overline{w}^{(T)}) - F_\theta(w^\star) \le \lambda\|w^{(0)} - w^\star\|^2 \exp\left(-\frac{T}{16\kappa^{3/2}}\right) + \frac{G^2}{\lambda T} + \frac{G^2\kappa^2}{\lambda T^2} + \frac{B}{\sqrt{\theta m}},$$

where we hide absolute constants and factors polylogarithmic in T and problem parameters.

Remark 2 (*About the Rate*) As soon as $T \gtrsim \kappa^{3/2}$ (ignoring constants and polylog factors), we achieve the optimal rate of $1/(\lambda T)$ rate of strongly convex stochastic optimization up to the bias $B/\sqrt{\theta m}$.

Further, the bias $B/\sqrt{\theta m}$ due to partial participation is larger at small θ and can be controlled by choosing the cohort size m large enough. In the experiments of Sect. 7, we obtain meaningful numerical results when m is around 50 or 100 and θ around 1/2, indicating that the worst-case bound (12) can be pessimistic.

Proof Sketch of Theorem 5 We start with some additional notation. We absorb the regularization into the client losses to define $\tilde{F}_i(w) = F_i(w) + (\lambda/2)\|w\|^2$. Now, consider the smoothed counterpart of (7) on a subset $S \subset [n]$ with a smoothing parameter $\nu > 0$ as

$$F^\nu_{\theta,S}(w) = \max_{\pi \in \mathcal{P}_{\theta,S}} \left\{ \sum_{i\in S} \pi_i \tilde{F}_i(w) - \nu D_S(\pi) \right\}.$$

It follows from the properties of smoothing (Nesterov, 2005; Beck & Teboulle, 2012) and composition rules that $F^\nu_{\theta,S}$ is L'-Lipschitz, where $L' = L + \lambda + G^2/\nu$. Finally, let $\mathcal{F}_t$ denote the sigma-algebra generated by $w^{(t)}$ and let $\mathbb{E}_t[\cdot] := \mathbb{E}[\cdot|\mathcal{F}_t]$.

We start the proof with the decomposition

$$\|w^{(t+1)} - w\|^2 = \|w^{(t)} - w\|^2 \underbrace{- 2\gamma \sum_{i\in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \langle \nabla \tilde{F}_i(w_{i,k}^{(t)}), w^{(t)} - w\rangle}_{=:\mathcal{T}_1}$$
$$\underbrace{+ \gamma^2 \left\| \sum_{i\in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \nabla \tilde{F}_i(w_{i,k}^{(t)}) \right\|^2}_{=:\mathcal{T}_2},$$

where w is arbitrary. For the first order term $\mathcal{T}_1$, we bound using λ-strong convexity and L-smoothness of $\tilde{F}_i$ as

$$\mathcal{T}_1' := 2\tau\gamma \sum_{i\in S} \pi_i^{(t)} \left\langle \nabla \tilde{F}_i(w^{(t)}, w^{(t)} - \overline{w}^\star \right\rangle = 2\tau\gamma \left\langle \nabla F^\nu_{\theta,S}(w^{(t)}), w^{(t)} - \overline{w}^\star \right\rangle$$
$$\ge 2\tau\gamma \left(F^\nu_{\theta,S}(w^{(t)}) - F^\nu_{\theta,S}(\overline{w}^\star) + \frac{\lambda}{2}\|w^{(t)} - \overline{w}^\star\|^2 \right).$$

where we used $\sum_{i\in S} \pi_i^{(t)} \nabla F_i(w^{(t)}) = \nabla F^\nu_{\theta,S}(w^{(t)})$ holds with smoothing, analogous to (11), and strong convexity.

The gap $\mathcal{T}_1 - \mathcal{T}_1'$ is due to the effect of the drift from multiple local steps. We bound this term similar to the non-convex case of Theorem 4. For the second order term $\mathcal{T}_2$, we rely on the variance bound of Levy et al. (2020, Prop. 2). Concretely, we have,

$$\mathbb{E}_{S\sim U_m}\left\|\sum_{i\in S}\pi_i^{(t)}\nabla\tilde{F}_i(w^{(t)})-\nabla\overline{F}_\theta^\nu(w^{(t)})\right\|^2\le\frac{8G^2}{\theta m},$$

where U_m is the uniform distribution over subsets $S\subset[n]$ of size m, and $\overline{F}_\theta^\nu(w):=\mathbb{E}_{S\sim U_m}F_{\theta,S}^\nu(w)$ as the expectation of $F_{\theta,S}^\nu$ over random subsets $S\sim U_m$. Putting these together and taking $w=\overline{w}^\star:=\arg\min\overline{F}_\theta^\nu$ gives the inequality,

$$\begin{aligned}\overline{F}_\theta^\nu(w^{(t)})-\overline{F}_\theta^\nu(\overline{w}^\star)\le\ &\gamma A+\gamma^2B\\&+\frac{1}{\gamma\tau}\left(1-\frac{\lambda\gamma\tau}{2}\right)\|w^{(t)}-\overline{w}^\star\|^2\\&-\frac{1}{\gamma\tau}\mathbb{E}_t\|w^{(t+1)}-\overline{w}^\star\|^2,\end{aligned}\tag{16}$$

where A, B are problem-dependent constants. We sum this up with the averaging weights β_t given in the statement of the theorem to get

$$\overline{F}_\theta^\nu(\overline{w}^{(T)})-\overline{F}_\theta^\nu(\overline{w}^\star)\le\frac{\lambda\|w^{(0)}-w^\star\|^2}{\exp(\lambda\tau\gamma T)-1}+A\gamma+B\gamma^2.$$

The final missing piece is a bound which allows us to translate statements about the convergence of $\overline{F}_\theta^\nu$ in terms of the convergence of F_θ. We achieve this using the bias bound of (12) together with the approximation error of smoothing. Finally, we optimize the choice of the learning rate and smoothing coefficient to give the final statement of the theorem. The details are provided in Sect. A.3. □

5.2 Privacy and Utility analysis

We now analyze the privacy and utility of Algorithm 2. In this section, we assume without loss of generality that $S=[n]$ so that $m=|S|=n$.

First, we recall the definition of concentrated differential privacy (Bun & Steinke, 2016). A randomized algorithm $\mathcal{A}$ satisfies $(1/2)\varepsilon^2$-concentrated differential privacy if the Rényi α-divergence $D_\alpha(\mathcal{A}(X)\|\mathcal{A}(X'))\le\alpha\varepsilon^2/2$ for all $\alpha\in(0,\infty)$ and all sequences X,X' of inputs that differ by the addition or removal of one client's data. Intuitively, the addition or removal of the data contributed by one client should not change the output distribution of the randomized algorithm by much, as measured by the Rényi divergence. A smaller value of ε implies a stronger privacy guarantee. This notion of differential privacy can be translated back and forth with the usual one, cf. Canonne et al. (2020).

Error Criterion. We approximate the $(1-\theta)$-quantile of the n per-client losses $\ell_i=F_i(w)$ for $i=1,\dots,n$ by the quantile of a hierarchical histogram h with entries $h(r,j)=\sum_{i=1}^n\mathbb{I}\left(l_{2^r(j-1)+1}\le F_i(w)<l_{2^rj}\right)$ where $0=l_0<l_1<\cdots<l_b=B$ are the bin edges. The edge l_j corresponding to index $j\in[b]$ approximates the $(1-\theta)$-quantile well if the cumulative mass $H(j)\approx(1-\theta)n$. We measure this error of approximation by the difference between the two sides. Formally, we define the error $R_\theta(H,j)$ of approximating the $(1-\theta)$-quantile of the cumulative function H of a hierarchical histogram with index $j\in[b]$ by

$$R_\theta(H,j) = \left| \frac{H(j)}{n} - (1-\theta) \right|. \tag{17}$$

We define the best achievable error $R^*_\theta(H)$ for estimating the $(1-\theta)$-quantile of the cumulative function H and the best approximating index $j^*(H)$ as

$$R^*_\theta(H) = \min_{j\in[b]} R_\theta(H,j), \quad \text{and} \quad j^*_\theta(H) = \arg\min_{j\in[b]} R_\theta(H,j), \tag{18}$$

where we assume ties are broken in an arbitrary but deterministic manner — note that $j^*_\theta(H)$ is defined here identically to (8). Lastly, we define the *quantile error* $\Delta_\theta(H,\hat{H})$ of estimating the quantile of the cumulative function H from that of $\hat{H}$ as

$$\Delta_\theta(\hat{H},H) = R_\theta\big(H, j^*_\theta(\hat{H})\big). \tag{19}$$

Essentially, if the index $j^*_\theta(\hat{H})$ computed from the estimate $\hat{H}$ corresponds to the $(1-\theta')$-quantile of H, the quantile error satisfies $\Delta_\theta(\hat{H},H) = |\theta - \theta'|$.

Privacy and Utility Analysis We now analyze the differential privacy bound of Algorithm 2 and the error in the quantile computation.

Theorem 6 *Fix a $\delta > 0$. Suppose that $\sigma \geq 1/2$ and $c > 0$ are given, and the modular arithmetic is performed on the base $M \geq 2 + 2cn + 2n\sqrt{2\sigma^2 \log(16nb/\delta)}$. Then, we have*:

(a) *Algorithm* 2 *satisfies $(1/2)\varepsilon^2$-concentrated DP with*

$$\varepsilon = \min\left\{ \sqrt{\frac{c^2 \log_2^2 b}{n\sigma^2} + \psi b}, \frac{c \log_2 b}{\sqrt{n}\sigma} + \psi\sqrt{2b} \right\},$$

where $\psi = 10\sum_{i=1}^{n-1} \exp\big(-2\pi^2\sigma^2 i/(i+1)\big) \leq 10(n-1)\exp(-2\pi^2\sigma^2)$.

(b) *With probability at least $1-\delta$, the quantile error of cumulative function $\hat{H}$ returned by Algorithm* 2 *is at most*

$$\Delta_\theta(\hat{H},H) \leq R^*_\theta(\hat{H}) + \sqrt{\frac{4\sigma^2}{c^2 n} \log_2 b \, \log\frac{4b}{\delta}}$$

where $R^*_\theta(\hat{H})$ is the error in the estimation of $(1-\theta)$-quantile of the cumulative function $\hat{H}$.

Let us interpret the result. The effective noise scale is σ/c. Since the dominant term of the privacy error is $\varepsilon \approx c\log_2 b/(\sigma\sqrt{n})$, we choose $\sigma/c \approx \log_2 b/(\varepsilon\sqrt{n})$, so that the algorithm satisfies $(1/2)\varepsilon^2$-concentrated DP. The role of c is to avoid the degeneracy of the discrete Gaussian as $\sigma \to 0$. In particular, the theorem requires $\sigma \geq 1/2$. The error resulting quantile error $\Delta_\theta(\hat{H},H)$ is (ignoring constants and log factors)

$$\Delta_\theta(\hat{H},H) \lesssim R^*_\theta(\hat{H}) + \frac{\log^2 b}{\varepsilon n}.$$

The quantile error scales as $1/(\varepsilon n)$. The total communication cost is $O(bn \log M)$ bits since the dimension of each hierarchical histogram is $2(b-2)$. If we take $\sigma = O(1)$ and $c = O(\varepsilon\sqrt{n})$, we require $M \gtrsim n^{3/2}$, so that the total communication cost is $O(bn \log n)$.

Proof of Theorem 6 We can show that no modular wraparound occurs anywhere in the algorithm with high probability. We assume that it holds for the proof sketch. Thus, for all valid levels r and indices j, we have $\tilde{x}_i(r,j) = cx_i(r,j) + \xi_i(r,j)$ and

$$\hat{h}(r,j) = \sum_{i=1}^{n} \frac{\tilde{x}_i(r,j)}{c} = \sum_{i=1}^{n} \left(x_i(r,j) + \frac{\xi_i(r,j)}{c} \right).$$

The privacy analysis follows from the sensitivity of the sum query. Namely, let $X = (x_1, \ldots, x_n)$ be a sequence and define $A(X) = \sum_{i=1}^{n} cx_i$ as the (rescaled) sum query. In our case, each x_i is a hierarchical histogram with $\log_2 b$ ones being the only non-zeros, one for each level of the tree. Algorithm 2 adds discrete Gaussian noise to the sum query to make it differentially private. That is, we get the randomized algorithm $\mathcal{A}(X) = A(X) + \sum_{i=1}^{n} \xi_i$. It was shown in Kairouz et al. (2021, Corollary 12) that $\mathcal{A}(X)$ is approximately distributed as $\mathcal{N}_{\mathbb{Z}}(A(X), n\sigma^2)$, so the desired privacy guarantee follows from that of the discrete Gaussian mechanism (Canonne et al., 2020). In particular, for two sequences X and X' differing by the addition or removal of a single basis vector x', we have that

$$D_\alpha(\mathcal{A}(X) \| \mathcal{A}(X')) \approx D_\alpha(\mathcal{N}_{\mathbb{Z}}(A(X), n\sigma^2) \| \mathcal{N}_{\mathbb{Z}}(A(X'), n\sigma^2)) = \frac{\alpha c^2}{2n\sigma^2}.$$

A rigorous analysis of the error, following the recipe of Kairouz et al. (2021), leads to the first part of the theorem; the details can be found in Appendix B.

Utility Analysis. The triangle inequality gives

$$\begin{aligned} \Delta_\theta(\hat{H}, H) &\le \frac{1}{n} \left| H\left(j_\theta^*(\hat{H})\right) - \hat{H}\left(j_\theta^*(\hat{H})\right) \right| + \left| \frac{1}{n} \hat{H}\left(j_\theta^*(\hat{H})\right) - (1-\theta) \right| \\ &\le \max_{j \in [b]} \left\{ \frac{1}{n} \left| H(j) - \hat{H}(j) \right| \right\} + R_\theta^*(\hat{H}). \end{aligned}$$

Using standard concentration arguments, we show that the first term is, at most $\sqrt{2\sigma^2 n \log_2(b) \log(4b/\delta)}$, completing the proof. □

6 Discussion

We discuss connections of Δ-FL to risk measures, fair resource allocation, and model personalization.

Connection to Risk Measures. The framework of risk measures in economics and finance formalizes the notion of minimizing the worst-case cost over a set of distributions (Föllmer & Schied, 2002; Rockafellar & Uryasev, 2013; Föllmer & Schied, 2016). The superquantile $\mathbb{S}_\theta(\cdot)$ is a special case of a risk measure. The Δ-FL framework, which minimizes the superquantile of the per-client losses, can be extended to other risk measures $\mathbb{M}$ by minimizing the objective

$$F_M(w) := \mathbb{M}(Z(w)) + \frac{\lambda}{2}\|w\|^2,$$

where $Z(w)$ is a discrete random variable which takes value $F_i(w)$ with probability $1/n$ for $i \in [n]$. Another example of a risk measure is the *entropic risk measure*, which is defined as $\mathbb{M}^{\nu}_{\text{ent}}(Z) = \mathbb{E}[\exp(\nu Z)]/\nu$ where $\nu \in \mathbb{R}_+$ is a parameter. The entropic risk measure is well defined provided the moment generating function $\mathbb{E}[\exp(\nu Z)]$ exists, for instance, for sub-Gaussian Z. The analog of Δ-FL with the entropic risk minimizes

$$F^{\nu}_{\text{ent}}(w) = \frac{1}{\nu}\log\left(\frac{1}{n}\sum_{i=1}^{n}\exp\left(\nu F_i(w)\right)\right) + \frac{\lambda}{2}\|w\|^2.$$

This objective $F^{\nu}_{\text{ent}}(w)$ coincides with the one studied recently in Li et al. (2021) under the name Tilted-ERM subsequent to the first presentation of this work (Laguel et al., 2020). Finally, we note that F^{ν}_{ent} is also related to the smoothed objective F^{ν}_{θ} from (14) as the limit

$$F^{\nu}_{\text{ent}}(w) = \lim_{\theta \to 0} F^{\nu}_{\theta}(w),$$

Maximin Strategy for Resource Allocation We would like to point out an interesting analogy between distributional robustness and proportional fairness. The superquantile-based objective in Eq. (7) is a maximin-type objective that is reminiscent of maximin objectives used in load balancing and network scheduling (Kubiak, 2008; Stanczak et al., 2009; Pantelidou & Ephremides, 2011).

We can draw an analogy between the two worlds, federated learning and resource allocation resp., by identifying errors to rates and clients to users. The maximin fair strategy to resource allocation seeks to treat all users as fairly as possible by making their rates as large and as equal as possible so that no rate can be increased without sacrificing other rates that are smaller or equal (Pantelidou & Ephremides, 2011).

Our superquantile-based Δ-FL framework builds off the maximin decision-theoretic foundation to frame an objective that we *optimize* with respect to *parameters* of models, and this, iteratively, over multiple rounds of client-server communication, while preserving the privacy of each client.

This compositional nature of our problem, where we optimize a composition (in the mathematical sense) of a maximin-type objective, a loss function, and model predictions differ with resource allocation in communication networks. Further explorations of the analogy are left for future work.

Model Family and Tail Thresholds Using a single global value of the tail threshold θ for all clients could fail to balance supporting tail clients with fitting the population average. To circumvent this issue, we use a similar idea to the one of Li et al. (2020) where a family of models is trained simultaneously for various levels, and each test client can tune its tail threshold.

Δ-FL vs. Model Personalization Consider a family of distributions $q_i(x, y)$ for $i = 1, \ldots, n$ over input-output pairs. From the decomposition $q_i(x, y) = q_i(x)q_i(y|x)$, it follows that the heterogeneity of the joint distributions can be due to (a) heterogeneity of the marginal distributions $q_i(x)$ over the input x, or, (b) heterogeneity of the conditional distributions $q_i(y|x)$, or in other words, the input-output mapping.

If two clients do not agree on their input-output mapping, a single global model cannot serve both simultaneously. Thus, when training one single global model (as in vanilla FL) or a small number of them (as in Δ-FL), there is an implicit assumption that the heterogeneity

of $\{q_i(y|x) \ : \ i \in [n]\}$ is small. Δ-FL was designed to handle the heterogeneity of $q_i(x)$ better than vanilla FL by providing better worst-case performance on tail clients.

On the other hand, the cases where the heterogeneity of the conditional distributions $q_i(y|x)$ is large requires a separate model per client, or in other words, model personalization. Standard approaches to model personalization still aim to minimize the average error across all clients (Dinh et al., 2020; Pillutla et al., 2022), similar to the vanilla FL objective. Thus, it can still suffer from disparate performance across clients, including poor performance on some tail clients or data-poor clients. One solution to reduce this disparity is to combine personalization with the Δ-FL objective. We refer to Sect. 7.6 for numerical experiments.

Quantile-based Filtering and Client Availability We note that the quantile-based filtering of Algorithm 3 implies that only θm tail clients contribute their updates to the global model in the absence of noise (that is, the weight $\pi^{(t)}$ in line 3 of Algorithm 3 is sparse; see also Proposition 3). In order to include the updates of m' clients after filtering, Δ-FL would require initially sampling an initial cohort of $m = m'/\theta$ clients. On the other hand, clients in cross-device federated learning are typically available in a diurnal pattern (Eichner et al., 2019; Kairouz et al., 2021), where a large enough number of clients might not be available at certain times of the day. This issue might be exacerbated by Δ-FL 's requirement of m'/θ clients per round as compared to FedAvg's m'. Devising strategies to dynamically vary the tail threshold θ based on the number of available clients to overcome this issue is an interesting venue for future work.

7 Experiments

In this section, we demonstrate the effectiveness of Δ-FL in handling heterogeneity in federated learning. Our experiments were implemented in Python using automatic differentiation provided by PyTorch while the data was preprocessed using LEAF (Caldas et al., 2018). The code to reproduce our experiments can be found online.[3] We start by describing the datasets, tasks, and models in Sect. 7.1. We present numerical comparisons to several recent works – we list them in Sect. 7.2 and show the experimental results in Sect. 7.3. We demonstrate that Δ-FL provides the most favorable tradeoff between average error and the error on tail clients in Sect. 7.4. Next, we compare Δ-FL with model personalization in Sect. 7.5. Finally, we numerically study the privacy-utility tradeoff of the differentially private quantile computation (in Sect. 7.6), and of Δ-FL with end-to-end differential privacy (in Sect. 7.7).

Full details regarding the experiments, as well as additional results, are provided in the appendices.

7.1 Datasets, tasks and models

We consider two learning tasks. The dataset and task statistics are summarized in Table 1.

(a) *Character Recognition*: We use the EMNIST dataset (Cohen et al., 2017), where the input x is a 28×28 grayscale image of a handwritten character, and the output y is its

[3] https://github.com/krishnap25/simplicial-fl.

Table 1 Dataset description and statistics

Task	Dataset	#Classes	Devices	#Data per client	
				Median	Max
Image recognition	EMNIST	62	1730	179	447
Sentiment analysis	Sent140	2	877	69	549

label (0-9, a-z, A-Z). Each client is a writer of the character x. The weight α_i assigned to author i is the number of characters written by this author. We train both a linear model and a convolutional neural network architecture (ConvNet). The ConvNet consists of two 5×5 convolutional layers with max-pooling followed by one fully connected layer. Outputs are vectors of scores for each of the 62 classes. The multinomial logistic loss is used to train both models.

(b) *Sentiment Analysis*: We use the Sent140 dataset (Go et al., 2009) where the input x is a tweet, and the output $y = \pm 1$ is its sentiment. Each client is a distinct Twitter user. The weight α_i assigned to user i is the number of tweets published by this user. We train a logistic regression and a Long-Short Term Memory neural network architecture (LSTM). The LSTM is built on the GloVe embeddings of the words of the tweet (Hochreiter & Schmidhuber, 1997). The hidden dimension of the LSTM is the same as the embedding dimension, i.e., 50. We refer to the latter as "RNN". The loss used to train both models is the binary logistic loss.

7.2 Algorithms and hyperparameters

We list here the competing approaches we benchmark and discuss their hyperparameters.

Algorithms As discussed in Sect. 3, a federated learning method is characterized by the objective function, as well as the federated optimization algorithm. We compare Δ -FL with the following baselines:

(a) Vanilla FL objective: We consider two methods that attempt to minimize the vanilla FL objective: FedAvg (McMahan et al., 2017) and FedProx (Li et al., 2020). The latter augments FedAvg with a proximal term for more stable optimization.

(b) Heterogeneity-aware objectives: We consider Tilted-ERM (Li et al., 2021), which is the analogue of Δ-FL with the entropic risk measure (cf. Sect. 6) and AFL (Mohri et al., 2019), whose objective is obtained as the limit $\lim_{\theta\to 0} F_\theta(w)$ of the Δ-FL objective. We also consider q-FFL (Li et al., 2020), which raises the per-client loss F_i to the $(q+1)^{\text{th}}$ power, for some $q > 0$. We optimize q-FFL and Tilted-ERM with the federated optimization algorithms proposed in their respective papers. We use q-FFL with $q = 10$ in place of AFL, as it was found to have more stable convergence with similar performance.

We compare to one more baseline for the vanilla FL objective. Note that Δ-FL the weight $\pi^{(t)}$ (see line 3 of Algorithm 3) is sparse, i.e., it is non-zero for only some of the

m selected clients, cf. Proposition 3. This is equivalent to a fewer number of effective clients per round, which is θm on average. We use as baseline FedAvg with θm clients per round, where m is the number of clients per round in Δ-FL; we call it FedAvg-Sub.

Similar to McMahan et al. (2017), we consider a weighted version of the vanilla FL objective where each client's loss is weighted by $\alpha_i = N_i/N$, where N_i is the number of data points on client i and $N = \sum_i N_i$. Similarly, we also consider a weighted version of the Δ-FL objective as a superquantile of a random variable that takes value $F_i(w)$ with probability α_i. For a fair comparison, we run all algorithms, including Δ -FL, without differential privacy. We postpone a study of Δ-FL with differential privacy to Sect. 7.7.

Hyperparameters We fix the number of clients per round to be $m = 100$ for each dataset-model pair except for Sent140-RNN, for which we use $m = 50$. We fixed an iteration budget for each dataset during which FedAvg converged. We tuned a learning rate schedule using grid search to find the smallest terminal loss averaged over training clients for FedAvg. The same iteration budget and learning rate schedule were used for *all* other methods, including Δ-FL. Each method, except FedAvg-Sub, selected m clients per round for training, as specified earlier. The regularization parameter λ, and the proximal weight of FedProx were tuned to minimize the 90th percentile of the misclassification error on a held-out subset of training clients. We run q-FFL for $q \in \{10^{-3}, 10^{-2}, \ldots, 10\}$ and report q with the smallest 90th percentile of misclassification error on *test* clients. We run Tilted-ERM with a temperature parameter $\nu \in \{0.1, 0.5, 1, 5, 10, 50, 100, 200\}$ and also report ν with the smallest 90th percentile of misclassification error on *test* clients. We optimize Δ-FL with Algorithm 3 for threshold levels $\theta \in \{0.8, 0.5, 0.1\}$.

7.3 Experimental results

We measure in Table 2 the 90th percentile of the misclassification error across the test clients as a measure of the right tail of the per-client performance. We also measure in Table 3 the mean error, which measures the average test performance. Our main findings are summarized below.

Δ-FL consistently achieves the smallest 90th percentile error. Δ-FL achieves a 3.3% absolute (12% relative) improvement over any vanilla FL objective on EMNIST-ConvNet. Among the heterogeneity-aware objectives, Δ-FL achieves 1.8% improvement over the next best objective, which is Tilted-ERM. We note that q-FFL marginally outperforms Δ-FL on Sent140-Linear, but the difference 0.05% is much smaller than the standard deviation across runs.

Δ-FL is competitive at multiple values of θ. For EMNIST-ConvNet, Δ-FL with $\theta \in \{0.5, 0.8\}$ is better in 90th percentile error than *all* other methods we compare to, and Δ -FL with $\theta = 0.1$ is tied with Tilted-ERM, the next best method. We also empirically confirm that Δ-FL interpolates between FedAvg ($\theta \to 1$) and AFL ($\theta \to 0$).

Δ-FL works best for larger threshold levels. We observe that Δ-FL with $\theta = 0.1$ is unstable for Sent140-RNN. This is consistent with Theorem 5, which requires m to be much larger than $1/\theta$ (cf. Remark 2). Indeed, this can be explained by Δ-FL 's sparse reweighting, which only gives non-zero weights to $\theta m = 5$ clients on average in each round (cf. Remark 1).

Table 2 **90th percentile** of the distribution of misclassification error (in %) on the test devices

	EMNIST		Sent140	
	Linear	ConvNet	Linear	RNN
FedAvg	$49.66_{0.67}$	$28.46_{1.07}$	$46.83_{0.54}$	$49.67_{3.95}$
FedAvg-Sub	$50.28_{0.77}$	$27.57_{0.81}$	$46.60_{0.38}$	$46.94_{3.84}$
FedProx	$49.15_{0.74}$	$27.01_{1.86}$	$46.83_{0.54}$	$49.86_{4.07}$
q-FFL	$49.90_{0.58}$	$28.02_{0.80}$	$\mathbf{46.39}_{0.40}$	$48.66_{4.68}$
Tilted-ERM	$48.59_{0.62}$	$25.46_{1.49}$	$46.69_{0.49}$	$46.54_{3.27}$
AFL	$51.62_{0.28}$	$45.08_{1.00}$	$47.52_{0.32}$	$57.78_{1.19}$
Δ-FL, $\theta = 0.8$	$49.10_{0.24}$	$26.23_{1.15}$	$46.44_{0.38}$	$\mathbf{46.46}_{4.39}$
Δ-FL, $\theta = 0.5$	$\mathbf{48.44_{0.38}}$	$\mathbf{23.69_{0.94}}$	$46.64_{0.41}$	$50.48_{8.24}$
Δ-FL, $\theta = 0.1$	$50.34_{0.95}$	$25.46_{2.77}$	$51.39_{1.07}$	$86.45_{10.95}$

Each entry is the mean over five random seeds while the standard deviation is reported in the subscript. The boldfaced/highlighted entries denote the smallest value for each dataset-model pair

Table 3 Mean of the distribution of misclassification error (in %) on the test devices

	EMNIST		Sent140	
	Linear	ConvNet	Linear	RNN
FedAvg	$34.38_{0.38}$	$16.64_{0.50}$	$34.75_{0.31}$	$30.16_{0.44}$
FedAvg-Sub	$34.51_{0.47}$	$16.23_{0.23}$	$34.47_{0.03}$	$\mathbf{29.86}_{0.46}$
FedProx	$\mathbf{33.82_{0.30}}$	$16.02_{0.54}$	$34.74_{0.31}$	$30.20_{0.48}$
q-FFL	$34.34_{0.33}$	$16.59_{0.30}$	$34.48_{0.06}$	$29.96_{0.56}$
Tilted-ERM	$34.02_{0.30}$	$15.68_{0.38}$	$34.70_{0.31}$	$30.04_{0.25}$
AFL	$39.33_{0.27}$	$33.01_{0.37}$	$35.98_{0.08}$	$37.74_{0.65}$
Δ-FL, $\theta = 0.8$	$34.49_{0.26}$	$16.09_{0.40}$	$\mathbf{34.41}_{0.22}$	$30.31_{0.33}$
Δ-FL, $\theta = 0.5$	$35.02_{0.20}$	$\mathbf{15.49_{0.30}}$	$35.29_{0.25}$	$33.59_{2.44}$
Δ-FL, $\theta = 0.1$	$38.33_{0.48}$	$16.37_{1.03}$	$37.79_{0.89}$	$51.98_{11.81}$

Each entry is the mean over five random seeds while the standard deviation is reported in the subscript. The boldfaced/highlighted entries denote the smallest value for each dataset-model pair

Yet, Δ-FL is competitive in terms of average error. Perhaps surprisingly, Δ-FL gets the best test error performance on EMNIST-ConvNet and Sent140-Linear. This suggests that the average test distribution is shifted relative to the average training distribution p_α. In the other cases, we find that the reduction in mean error is small relative to the gains in the 90th percentile error compared to Vanilla FL methods.

Minimizing superquantile loss over all clients performs better than minimizing worst error over all clients. Specifically, AFL which aims to minimize the worst error among all clients, as well as other objectives which approximate it (Δ-FL with $\theta \to 0$, q-FFL with $q \to \infty$, Tilted-ERM with $\nu \to 0$) tend to achieve poor performance. We find that AFL achieves the highest error both in terms of 90th percentile and the mean. Δ-FL offers a more nuanced and more effective approach through an averaging of the tail performances rather than the straightforward pessimistic approach minimizing the worst error among all clients.

7.4 Exploring the trade-off between average and tail error

We visualize in Figs. 3 and 4 the distribution of test errors to explore the trade-off various methods provide between the average error and the error on tail clients.

Δ-FL yields improved prediction on tail clients. This can be observed from the histogram of Δ-FL in Fig. 3, which exhibits thinner tails than FedAvg or Tilted-ERM. We see that the vanilla FL objective of FedAvg sacrifices performance on the tail clients. Tilted-ERM does improve over FedAvg in this regard, but Δ-FL has a thinner right tail than Tilted-ERM, showing better handling of heterogeneity.

Δ-FL yields improved prediction on data-poor clients. We observe in Fig. 4 that Tilted-ERM and q-FFL mainly improve the performance on data-rich clients, that is clients with lots of data. On the other hand, Δ-FL gives a more significant reduction in misclassification error on data-poor clients, that is clients with little data (< 200 examples per client).

7.5 Δ-FL and model personalization

We now repeat the experiment of Sect. 7.3 with model personalization for the EMNIST ConvNet model.

Setup We personalize a model to a test client by finetuning a model trained either via FedAvg or Δ-FL on the particular test client's data at the end of federated training. This simple baseline is competitive with more sophisticated personalization algorithms (Pillutla et al., 2022). Towards this end, we split the data on each test client into a training set used for the finetuning and a test set used to report the evaluation metrics. We finetune the model for 10 epochs with the same local learning rate as at the end of federated training.

Results The numerical results are given in Table 4. We observe that after model personalization, both FedAvg and Δ-FL models perform similarly, often within one standard deviation of each other. The mean error is marginally smaller for FedAvg while the 90th percentile error is marginally smaller for Δ-FL with $\theta = 0.8$. The gap between these, 0.01 or 0.02 percentage points, is smaller than the standard deviation, 0.1 percentage points.

7.6 Differentially private quantile estimation

We study the privacy-utility tradeoff of Algorithm 2.

Setup We sample $n = 256$ numbers from a uniform distribution over $[0, B]$ or a $\chi^2(4)$ distribution clipped to $[0, B]$ with $B = 10$. We consider the performance of Algorithm 2 by varying the number b of bins and the ring size M. Since the communication cost of the protocol scales as the bit width $\log_2 M$, we display it instead in the plots. Recall that if our algorithm returns the $(1 - \theta')$-quantile when we aim to find the $(1 - \theta)$-quantile, then its quantile error is $|\theta - \theta'|$, cf. (19). We plot the quantile error averaged over $\theta = 0.1, 0.2, \ldots, 0.9$, and the standard deviations are obtained from 10 random runs.

Results The results are given in Fig. 5. For $n = 256$ and $b = 64$, we find that the quantile error is 0.14 for the uniform distribution at $\varepsilon \approx 1$; this means we might find the 36rd percentile or the 64th percentile instead of the median. This error quickly falls to 0.03 at $\varepsilon \approx 5$ at large enough bit widths. At a bit width of 10, we incur errors due to the modular wraparound at $\varepsilon \geq 5$. The results are also qualitatively similar for other settings, although the quantile error is unsurprisingly higher at $b > n$.

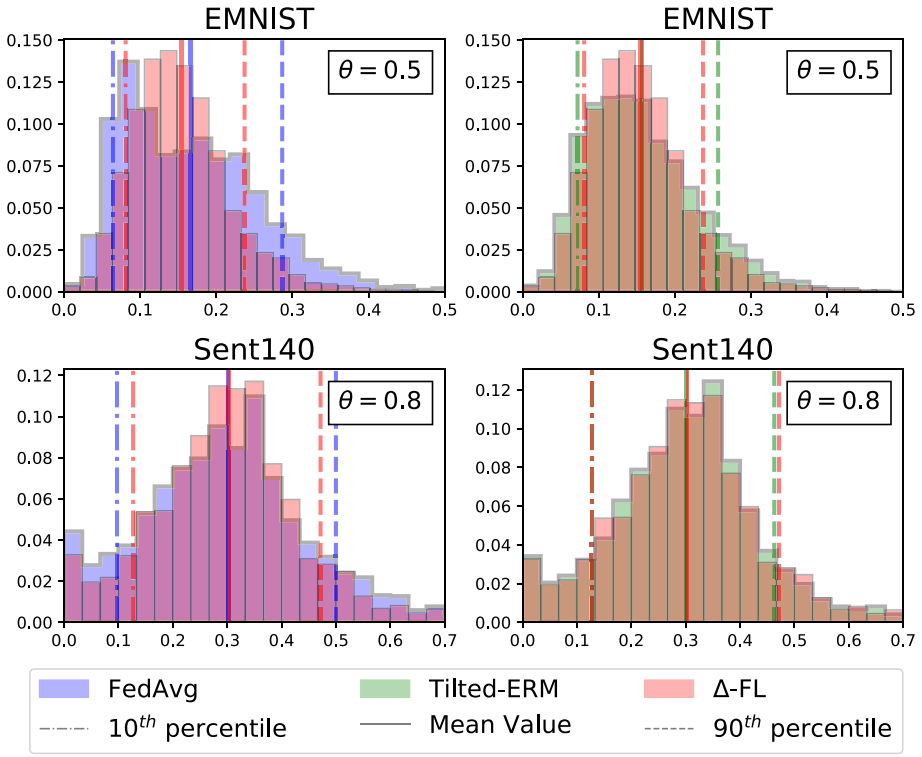

Fig. 3 Histogram of misclassification error on test clients for the EMNIST-ConvNet and Sent140-RNN

7.7 End-to-end differential privacy with Δ-FL

We now compare Δ-FL with FedAvg with end-to-end differential privacy on a synthetic classification dataset.

Dataset and models The synthetic dataset contains $k = 10$ classes in $d = 20$ dimensions and $n = 2500$ training clients. The class-conditional distribution $q(x|y = k) = \mathcal{N}(\mu_k, I_d)$ is a Gaussian and is the same across all the clients while there is a label shift, i.e., $q_i(y)$ varies across clients. For the training clients, we have $q_i(y) = \text{Dir}(0.5)$ is a Dirichlet distribution with parameter 0.5, while for validation and test clients, we have $q_i(y) = \text{Dir}(0.01)$. For each client, we sample 100 examples from its data distribution. We refer to Appendix D for details.

Algorithms and privacy budgeting For the FedAvg baseline, we clip the model updates to an ℓ_2 norm bound of C, which is a tunable hyperparameter. We add Gaussian noise $\mathcal{N}(0, \sigma_w^2 I)$ — thus, each update satisfies $\sigma_w^2/(2C^2)$-concentrated differential privacy. To get a privacy bound across all the rounds, we use the generic bounds of Zhu and Wang (2019) for privacy amplification by subsampling and composing the privacy loss across the number of rounds of the algorithm. Given a fixed norm bound C, we select the noise scale σ_w to get $(\varepsilon, 1/n)$-differential privacy over the entire algorithm, where ε is provided as an input, and n is the number of clients.

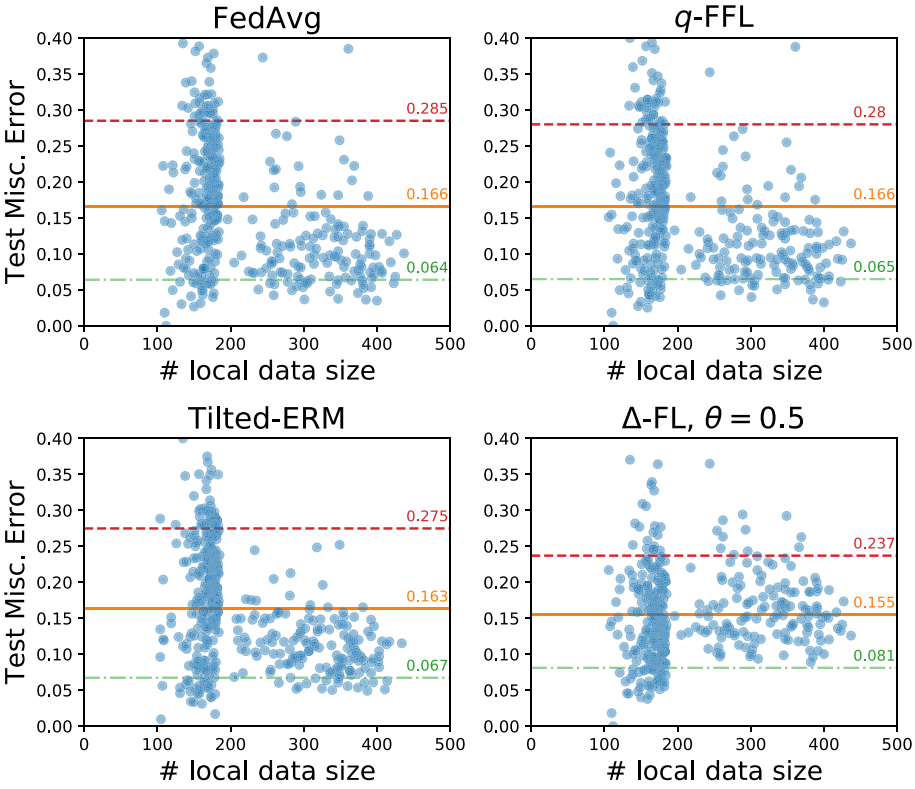

Fig. 4 Scatter plots of misclassification error on test clients against its data size for the EMNIST-ConvNet

Each round of Δ-FL involves quantile computation and weight aggregation: we use Algorithm 2 to compute the quantile of the losses clipped to a tuned bound B using a hierarchical histogram with b bins. We clip the weight updates to a norm bound C and add Gaussian noise, similar to FedAvg. The total privacy loss is calculated by composing the privacy loss across both the quantile and weight updates, and the number of rounds together with amplification by subsampling using the bounds of Zhu and Wang (2019).

We calculate the noise scales σ_q of the quantile and σ_w of the weight update so that (a) the privacy budget for the quantile computation to be r times the privacy budget of the weight update, where r is a hyperparameter, and (b) the overall algorithm satisfies $(\epsilon, 1/n)$ -differential privacy. We tune the loss bound B, norm bound C, the number of bins b, and the quantile privacy ratio r to attain the best 90th percentile misclassification error across validation clients. For all experiments, we train for 1000 rounds with 100 clients per round and a fixed learning rate of 0.1. For further details on the algorithms, privacy budgeting, and hyperparameters, we refer to Appendix D.

Results: Δ-FL gives better tail performance under the same privacy budget. The privacy-utility tradeoff of Δ-FL and FedAvg are shown in Fig. 6. We see that Δ-FL with threshold level $\theta = 0.5$ has a privacy-utility tradeoff within one standard deviation of FedAvg on the mean misclassification error while being 3.1 percentage points better on the tail misclassification error as measured by its 90th percentile: 55.7% for FedAvg versus 52.6% for Δ-FL at $\varepsilon = 5$. Smaller values of θ, such as $\theta = 0.25$ are 0.6 percentage points worse on

Table 4 Misclassification error % of FedAvg and Δ-FL with model personalization on the EMNIST ConvNet model

	Mean error		90th percentile error	
	Before pers.	After pers.	Before pers.	After pers.
FedAvg	$16.68_{0.50}$	$\mathbf{5.43}_{0.12}$	$28.44_{1.15}$	$8.71_{0.19}$
Δ-FL, $\theta = 0.8$	$16.00_{0.44}$	$5.44_{0.08}$	$26.26_{1.28}$	$\mathbf{8.69}_{0.12}$
Δ-FL, $\theta = 0.5$	$\mathbf{15.50}_{\mathbf{0.31}}$	$5.58_{0.07}$	$\mathbf{23.61}_{\mathbf{1.02}}$	$8.76_{0.15}$
Δ-FL, $\theta = 0.1$	$16.05_{0.78}$	$6.17_{0.11}$	$24.58_{1.96}$	$9.38_{0.06}$

Each table entry is the average over 5 random seeds, while the subscript denotes the standard deviation. The boldfaced entries indicate the smallest error in each column

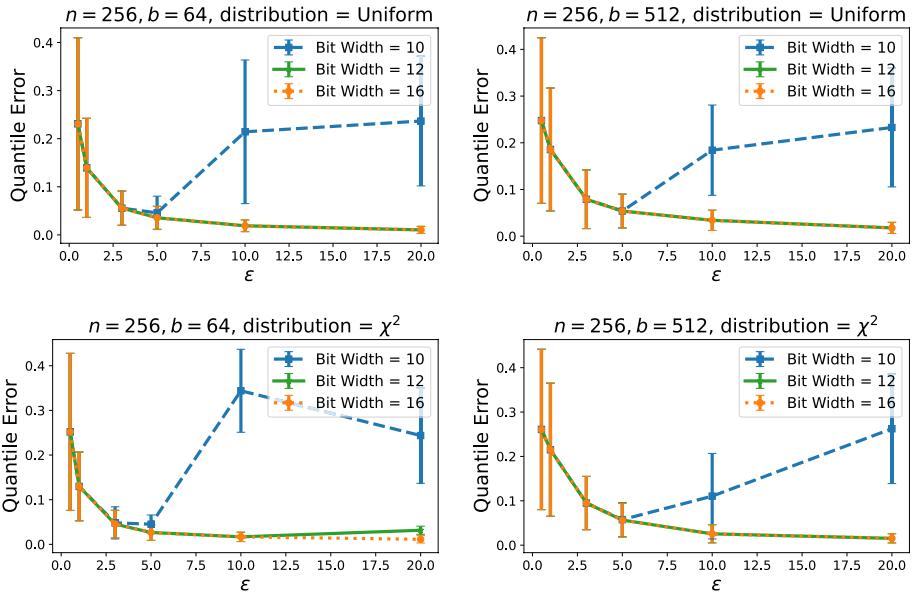

Fig. 5 The quantile error (defined in (19)) incurred by Algorithm 2 to estimate the quantile of $n = 256$ numbers drawn from a uniform or $\chi^2(4)$ distribution with $(\varepsilon, 10^{-5})$-differential privacy

the mean error while being 1.2 and 4.3 percentage points better than $\theta = 0.5$ and FedAvg respectively on the tail error. We note that the utility of Δ-FL degrades more at smaller ε when compared to FedAvg: 1.64 percentage points for $\theta = 0.5$ versus 0.2 percentage points for FedAvg from $\varepsilon = 10$ to $\varepsilon = 3$ for the tail error. Despite this effect, the tail error for Δ-FL is smaller than FedAvg even at $\varepsilon = 3$.

8 Conclusion

We present the Δ-FL framework that operates with heterogeneous clients while guaranteeing a minimal predictive performance to each client. Δ-FL relies on a superquantile-based objective, parameterized by a tail threshold level, to optimize the tail statistics of

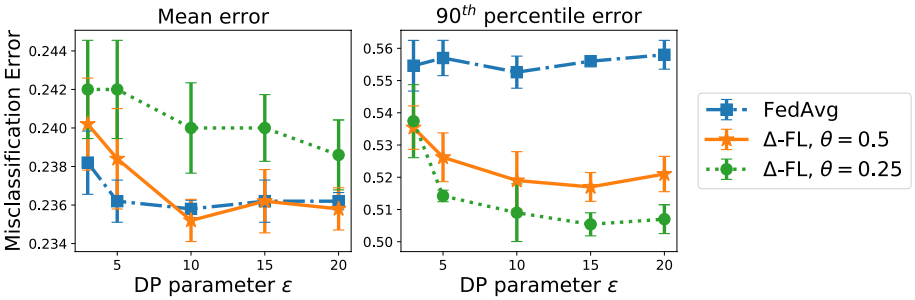

Fig. 6 Δ-FL versus FedAvg with $(\varepsilon, 1/n)$-differential privacy on a synthetic classification task in $\mathbb{R}^{20}$ with 10 classes and $n = 2500$ clients. The error bars denote the standard deviation across 5 random runs

the prediction errors on the client data distributions. We present a federated optimization algorithm that combines differentially private quantile estimation to filter out clients to run federated averaging steps. We derive finite time convergence guarantees of $O(1/\sqrt{T})$ in T communication rounds in the nonconvex case and $O(\exp(-T/\kappa^{3/2}) + \kappa/T)$ in the strongly convex case with local condition number κ. We establish a utility bound of $O(\log^2 b/(\varepsilon n))$ for (ε, δ)-differentially private quantile computation. Experimental results on federated learning benchmarks demonstrate the superior performance of Δ-FL over state-of-the-art baselines on the upper quantiles of the error on test clients, with particular improvements on data-poor clients, while being competitive on the mean error with and without differential privacy.

Appendix A: Convergence analysis

Below, we restate and prove Theorem 4 as Theorem 7 in Sect. A.2 and Theorem 5 as Theorem 8 in Sect. A.3.

A.1: Review of Notation

Here, we review the notation of the variants of the functions F_i and F_θ in Table 5.

A.2: Convergence analysis: non-convex case

We review some definitions of subdifferentials and weak convexity before we get to the main theorem.

Nonconvex Subdifferentials. We start by recalling the definition of subgradients for nonsmooth functions (in finite dimension), following the terminology of Rockafellar and Wets (2009). Consider a function $\psi : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ and a point $\bar{w}$ such that $\psi(\bar{w}) < +\infty$. The regular (or Fréchet) subdifferential of ψ at $\bar{w}$ is defined by

$$\partial\psi(\bar{w}) = \left\{ s \in \mathbb{R}^d \ : \ \psi(w) \geq \psi(\bar{w}) + \langle s, w - \bar{w} \rangle + o(\|w - \bar{w}\|) \right\}.$$

Table 5 Review of notation

Function	Description
F_i	Loss function of client i
$\tilde{F}_i$	Loss plus regularization on client i: $\tilde{F}_i(w) = F_i(w) + \frac{\lambda}{2}\|w\|^2$
F_θ	The main objective of Δ-FL, defined in (7)
$F_{\theta,S}$	The analogue of F_θ defined on only on a sample S of clients
$\overline{F}_\theta$	Averaged minibatch objective: $\overline{F}_\theta(w) = \mathbb{E}_S[F_{\theta,S}(w)]$ where the expectation is over uniform subsamples of clients of size $\|S\| = m$
$\overline{\Phi}_\theta^\mu$	The Moreau envelope of $\overline{F}_\theta$; see (13)
$\hat{F}_\theta$	The variant of the Δ-FL objective computed with a tail mean, and used to formalize the connection between Algorithms 1 and 3
F_θ^ν	Smoothing of F_θ using the KL divergence; see (14)

The regular subdifferential thus corresponds to the set of gradients of smooth functions that are below ψ and coincide with it at $\bar{w}$. These notions generalize (sub)gradients of both smooth functions and convex functions: it reduces to the singleton $\{\nabla\psi(\bar{w})\}$ when ψ is smooth and to the standard subdifferential from convex analysis when ψ is convex.

Weak Convexity. We recall the notion of weak convexity, which is one way of characterizing functions that are "close" to convex. A function $\psi : \mathbb{R}^d \to \mathbb{R}$ is said to be η -weakly convex if the function $w \mapsto \psi(w) + (\eta/2)\|w\|^2$ is convex (Nurminskii, 1973). The class of weakly convex functions includes all convex functions (with $\eta = 0$) and all L-smooth functions (with $\eta = L$).

Weak convexity also admits an equivalent first-order condition: for any $w, z \in \mathbb{R}^d$ and $s \in \partial\psi(w)$, we have,

$$\psi(z) \geq \psi(w) + \langle s, z - w\rangle - \frac{\eta}{2}\|z - w\|^2 . \tag{A1}$$

Weak convexity will feature in our developments in two ways:

- In our case, both F_θ as well as $F_{\theta,S}$ are L-weakly convex, since each can be written as the maximum of a family of L-smooth functions; cf. Drusvyatskiy & Paquette (2019, Lemma 4.2).
- The prox operator for weakly convex functions is well-defined. Let ψ be a η-weakly convex function. Its proximal or prox operator, with parameter $\mu > 0$, is defined as

$$\text{prox}_{\psi/\mu}(w) = \arg\min_z \left\{ \psi(z) + \frac{\mu}{2}\|w - z\|^2 \right\} .$$

  It is well-defined (i.e., the argmin exists and is unique) for $\mu > \eta$, since the function inside the argmin is $(\mu - \eta)$-strongly convex.

In nonsmooth and nonconvex optimization of weakly convex functions, we are interested in finding stationary points w.r.t. the regular subdifferential, i.e., points w satisfying $0 \in \partial\psi(w)$. A natural measure of near-stationarity is, therefore,

$$\text{dist}(0, \partial\psi(w)) = \inf_{s \in \partial\psi(w)} \|s\| .$$

Moreau Envelope. Given a parameter $\mu > 0$, we define the Moreau envelope of $\overline{F}_\theta$ as

$$\overline{\Phi}_\theta^\mu(w) = \inf_z \left\{ \overline{F}_\theta(z) + \frac{\mu}{2}\|w - z\|^2 \right\}.$$

The Moreau envelope is well-defined since $\overline{F}_\theta$ is bounded from below by our assumptions. We will use two standard properties of the Moreau envelope:

- Since $\overline{F}_{\theta,S}$ is L-weakly convex, we have that its Moreau envelope $\overline{\Phi}_\theta^\mu(w)$ is continuously differentiable for $\mu > L$ with

$$\nabla \overline{\Phi}_\theta^\mu(w) = \mu\left(w - \mathrm{prox}_{\overline{F}_\theta/\mu}(w)\right). \tag{A2}$$

- The stationary points of $\overline{\Phi}_\theta^\mu$ and $\overline{F}_\theta$ coincide and $\inf \overline{\Phi}_\theta^\mu = \inf \overline{F}_\theta$ for $\mu > L$.
- We have for all $\mu > 0$ that $\overline{\Phi}_\theta^\mu(w) \le \overline{F}_\theta(w)$.

Notation. Let $S = S^{(t)}$ denote the random set of clients selected in round t of Algorithm 3. We define

$$\tilde{\nabla} F_{\theta,S}(w^{(t)}) = \sum_{i \in S} \pi_i^{(t)} \nabla F_i(w^{(t)}), \tag{A3}$$

where $\pi_i^{(t)} \in \arg\max_{\pi \in \mathcal{P}_{\theta,S}} \sum_{i \in S} \pi_i F_i(w^{(t)})$ is selected as in line 3 of Algorithm 3. A key consequence of the chain rule (cf. Rockafellar & Wets (2009, Thm. 10.6)) is

$$\tilde{\nabla} F_{\theta,S}(w^{(t)}) \in \partial F_{\theta,S}(w^{(t)}). \tag{A4}$$

Convergence analysis We now state and prove the convergence result in the nonconvex case.

Theorem 7 *Fix the number of local steps τ and the number of rounds T, fix $\mu = 2L$ and set the learning rate*

$$\gamma = \min\left\{ \frac{1}{4\tau L}, \frac{1}{\tau\sqrt{T}}\sqrt{\frac{\Delta F_0}{LG^2}}, \frac{1}{\tau T^{1/3}}\left(\frac{\Delta F_0}{32L^2G^2(1-\tau^{-1})}\right)^{1/3} \right\},$$

where we denote $\Delta F_0 = \overline{\Phi}_\theta^\mu(w^{(0)}) - \inf \overline{\Phi}_\theta^\mu \le \overline{F}_\theta(w^{(0)}) - \inf \overline{F}_\theta$. Let $\hat{w}$ be sampled uniformly at random from $\{w^{(0)}, \ldots, w^{(T-1)}\}$. Ignoring absolute constants, we have the bound,

$$\mathbb{E}\left\|\nabla \overline{\Phi}_\theta^\mu(\hat{w})\right\|^2 \le \sqrt{\frac{\Delta F_0 LG^2}{T}} + \left(\frac{\Delta F_0 LG(1-\tau^{-1})^{1/2}}{T}\right)^{2/3} + \frac{\Delta F_0 L}{T}.$$

Proof We start with some notation. Throughout, we denote $z^{(t)}$ as the proximal point of $w^{(t)}$:

$$z^{(t)} = \mathrm{prox}_{\overline{F}_\theta/\mu}(w^{(t)}) = \arg\min_z \left\{ \overline{F}_\theta(z) + \frac{\mu}{2}\left\|z - w^{(t)}\right\|^2 \right\}.$$

Let $\mathcal{F}^{(t)}$ denote the sigma algebra generated by $w^{(t)}$ and define $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \mid \mathcal{F}^{(t)}]$. By definition, we have that $z^{(t)}$ is also $\mathcal{F}^{(t)}$-measurable.

We use the update $w^{(t+1)} = w^{(t)} - \gamma \sum_{i \in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \nabla F_i(w_{i,k}^{(t)})$ to get

$$
\begin{aligned}
\overline{\Phi}_\theta^\mu(w^{(t+1)}) &= \min_z \left\{ \overline{F}_\theta(z) + \frac{\mu}{2}\left\|z - w^{(t+1)}\right\|^2 \right\} \\
&\le \overline{F}_\theta(z^{(t)}) + \frac{\mu}{2}\left\|z^{(t)} - w^{(t+1)}\right\|^2 \\
&= \overline{F}_\theta(z^{(t)}) + \frac{\mu}{2}\left\|z^{(t)} - w^{(t)}\right\|^2 \\
&\quad + \mu\gamma\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\rangle \\
&\quad + \frac{\mu\gamma^2}{2}\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\|^2 \\
&= \overline{\Phi}_\theta^\mu(w^{(t)}) + \underbrace{\mu\gamma\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\rangle}_{=:\mathcal{T}_1} \\
&\quad + \underbrace{\frac{\mu\gamma^2}{2}\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\|^2}_{=:\mathcal{T}_2}.
\end{aligned}
\tag{A5}
$$

For $\mathcal{T}_1$, we consider the effect of a single update with a learning $\tau\gamma$:

$$
\mathcal{T}_1' := \mu\tau\gamma\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\nabla F_i(w^{(t)})\right\rangle,
$$

so that the difference $\mathcal{T}_1 - \mathcal{T}_1'$ is the effect of the drift introduced by taking multiple local steps. We bound the first order term $\mathcal{T}_1'$, the drift term $\mathcal{T}_1 - \mathcal{T}_1'$ and the second order term $\mathcal{T}_2$ separately.

Bounding the first order term $\mathcal{T}_1'$. By definition of the weights $\pi_i^{(t)}$, we have $\sum_{i\in S}\pi_i^{(t)}\nabla F_i(w^{(t)}) = \tilde{\nabla} F_{\theta,S}(w^{(t)}) \in \partial F_{\theta,S}(w^{(t)})$, see also (A3). This allows us to invoke the weak convexity of $F_{\theta,S}$, in particular (A1), to bound

$$
\frac{\mathcal{T}_1'}{\mu\tau\gamma} = \left\langle z^{(t)} - w^{(t)}, \tilde{\nabla} F_{\theta,S}(w^{(t)})\right\rangle \le F_{\theta,S}(z^{(t)}) - F_{\theta,S}(w^{(t)}) + \frac{L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2.
$$

Taking an expectation conditioned on $\mathcal{F}^{(t)}$ (i.e., over the randomness in S), we get $\mathbb{E}_t[F_{\theta,S}(w^{(t)})] = \overline{F}_\theta(w^{(t)})$. Further, since $z^{(t)}$ is $\mathcal{F}^{(t)}$-measurable, we also have $\mathbb{E}_t[F_{\theta,S}(z^{(t)})] = \overline{F}_\theta(z^{(t)})$. That gives,

$$
\frac{1}{\mu\tau\gamma}\mathbb{E}_t[\mathcal{T}_1'] \le \left(\overline{F}_\theta(z^{(t)}) + \frac{\mu}{2}\left\|z^{(t)} - w^{(t)}\right\|^2\right) - \overline{F}_\theta(w^{(t)}) - \frac{\mu - L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2.
$$

Note that the function

$$
h(z) := \overline{F}_\theta(z) + \frac{\mu}{2}\left\|z - w^{(t)}\right\|^2
$$

is $(\mu - L)$-strongly convex and $z^{(t)}$ is its minimizer. This gives,

$$h(w^{(t)}) - h(z^{(t)}) \geq \frac{\mu - L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2,$$

so that we have the bound

$$\frac{1}{\mu\tau\gamma}\mathbb{E}_t[\mathcal{T}_1'] \leq -(\mu - L)\left\|z^{(t)} - w^{(t)}\right\|^2 \overset{(A2)}{=} -\frac{\mu - L}{\mu^2}\left\|\nabla\overline{\Phi}_\theta^\mu(w^{(t)})\right\|^2. \tag{A6}$$

Bounding the effect of the drift $\mathcal{T}_1 - \mathcal{T}_1'$. The contribution of k^{th} local step to the drift $\mathcal{T}_1 - \mathcal{T}_1'$ can be bounded as

$$\begin{aligned}
&\left|\left\langle z^{(t)} - w^{(t)}, \sum_{i\in S}\pi_i^{(t)}\left(\nabla F_i(w_{i,k}^{(t)}) - \nabla F_i(w^{(t)})\right)\right\rangle\right| \\
&\overset{\text{(i)}}{\leq}\frac{\mu - L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2 + \frac{1}{2(\mu - L)}\left\|\sum_{i\in S}\pi_i^{(t)}\left(\nabla F_i(w_{i,k}^{(t)}) - \nabla F_i(w^{(t)})\right)\right\|^2 \\
&\overset{\text{(ii)}}{\leq}\frac{\mu - L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2 + \frac{1}{2(\mu - L)}\sum_{i\in S}\pi_i^{(t)}\left\|\nabla F_i(w_{i,k}^{(t)}) - \nabla F_i(w^{(t)})\right\|^2 \\
&\overset{\text{(iii)}}{\leq}\frac{\mu - L}{2}\left\|z^{(t)} - w^{(t)}\right\|^2 + \frac{L^2}{2(\mu - L)}\sum_{i\in S}\pi_i^{(t)}\left\|w_{i,k}^{(t)} - w^{(t)}\right\|^2.
\end{aligned}$$

Here, we first used (i) the Cauchy-Schwarz inequality, (ii) Jensen's inequality, and (iii) the smoothness of F_i. Summing this over k, we get the bound

$$\begin{aligned}
\mathbb{E}_t|\mathcal{T}_1 - \mathcal{T}_1'| &\leq \frac{\tau\gamma(\mu - L)}{2\mu}\left\|\nabla\overline{\Phi}_\theta^\mu(w^{(t)})\right\|^2 + \frac{\mu\gamma L^2}{2(\mu - L)}d^{(t)} \\
&\leq \frac{\tau\gamma(\mu - L)}{2\mu}\left\|\nabla\overline{\Phi}_\theta^\mu(w^{(t)})\right\|^2 + \frac{4\mu\tau^3\gamma^3G^2}{\mu - L}(1 - \tau^{-1}),
\end{aligned} \tag{A7}$$

where we bounded $d^{(t)} := \mathbb{E}_t\left[\sum_{i\in S}\sum_{k=0}^{\tau-1}\pi_i^{(t)}\left\|w_{i,k}^{(t)} - w^{(t)}\right\|^2\right]$ by Proposition 12.

Bounding the second order term $\mathcal{T}_2$. Next, we bound $\mathcal{T}_2$ as

$$\begin{aligned}
\mathcal{T}_2 = \frac{\mu\gamma^2}{2}\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla F_i(w_{i,k}^{(t)})\right\|^2 &\leq \frac{\mu\gamma^2\tau}{2}\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\left\|\nabla F_i(w_{i,k}^{(t)})\right\|^2 \\
&\leq \frac{\mu\gamma^2\tau^2G^2}{2},
\end{aligned} \tag{A8}$$

where we used Jensen's inequality and $\left\|\nabla F_i(w_{i,k}^{(t)})\right\|^2 \leq G^2$ since F_i is G-Lipschitz.

One step update and telescoping the bound Plugging (A6) to (A8) into (A5), we have,

$$\begin{aligned}
\mathbb{E}_t\left[\overline{\Phi}_\theta^\mu(w^{(t+1)})\right] \leq \overline{\Phi}_\theta^\mu(w^{(t)}) &- \frac{\tau\gamma(\mu - L)}{2\mu}\left\|\nabla\overline{\Phi}_\theta^\mu(w^{(t)})\right\|^2 \\
&+ \frac{\mu\gamma^2\tau^2G^2}{2}\left(1 + \frac{8L^2\gamma}{\mu - L}(\tau - 1)\right).
\end{aligned}$$

Finally, taking an unconditional expectation, summing this up over $t = 0$ to $T - 1$ and rearranging gives us the bound

$$\mathbb{E}\left\|\nabla\overline{\Phi}^{\mu}_{\theta}(\hat{w})\right\|^2 \leq \frac{4\Delta F_0}{\tau\gamma T} + 4\tau\gamma LG^2(1 + 8L\gamma(\tau - 1)),$$

where we plugged in $\mu = 2L$. Plugging in the choice of γ (cf. Lemma 14) completes the proof. □

A.3: Convergence analysis: strongly convex case

The fully specified version of Theorem 5 is the following.

Theorem 8 (Convergence rate, Strongly Convex Case) *Suppose that each F_i is convex and the regularization parameter satisfies $0 < \lambda < L$. Define notation $\kappa = (L + \lambda)/\lambda$, $w^\star = \arg\min_w F_\theta(w)$ and $\Delta_0 = \|w^{(0)} - w^\star\|^2$. Assume also that the number of rounds is $T \geq 16\kappa^{3/2}$. Fix a smoothing parameter $\nu > 0$ as*

$$\nu = \frac{8G^2\delta}{\lambda}\left(1 \vee 32\kappa^2\delta\right),$$

where $\delta > 0$ is given by

$$\delta = \min\left\{\frac{1}{16\kappa^{3/2}}, \frac{1}{T}\left(1 \vee \log\frac{CT}{\log m}\right), \frac{1}{T}\left(1 \vee \log\frac{CT^2}{\kappa^2\log m}\right)\right\},$$

and $C = \lambda^2\Delta_0/G^2$. Letting $L' = L + \lambda + G^2/\nu$, fix a learning rate

$$\gamma = \min\left\{\frac{1}{4\tau L'}, \frac{1}{8\tau\kappa\sqrt{2\lambda L'}}, \frac{1}{\lambda\tau T}(1 \vee \log C\theta mT), \frac{1}{\lambda\tau T}\left(1 \vee \log\frac{CT^2}{\kappa^2(1 - \tau^{-1})}\right)^2\right\}.$$

Consider the sequence $(w^{(t)})_{t=0}^T$ produced by Algorithm 3 run with smoothing parameter ν and learning rate γ chosen as above, and the corresponding averaged iterate

$$\overline{w}^{(T)} := \frac{\sum_{t=0}^T w^{(t)}\left(1 - \frac{\lambda\tau\gamma}{2}\right)^{-(1+t)}}{\sum_{r=0}^T\left(1 - \frac{\lambda\tau\gamma}{2}\right)^{-(1+r)}}.$$

Then, ignoring absolute constants, we have,

$$\begin{aligned}\mathbb{E}\left[F_\theta(\overline{w}^{(T)}) - F_\theta(w^\star)\right] \leq{}& \lambda\|w^{(0)} - w^\star\|^2\exp\left(-\frac{T}{16\kappa^{3/2}}\right) + \frac{B}{\sqrt{\theta m}} \\ &+ \frac{G^2}{\lambda T}\left(\frac{1}{\theta m} + \log m\right)\left(1 \vee \log\frac{\lambda^2\Delta_0\theta mT}{G^2}\right) \\ &+ \frac{G^2\kappa^2}{\lambda T^2}\left(1 - \tau^{-1} + \log m\right)\left(1 \vee \log\frac{\lambda^2\Delta_0 T^2}{G^2\kappa^2}\right)^2.\end{aligned}$$

We review some notation before giving the proof.

Notation. Analogous to the smoothing F_θ^ν of F_θ, we define the smoothing of the sample version $F_{\theta,S}$ as

$$F_{\theta,S}^\nu(w) = \max_{\pi \in \mathcal{P}_{\theta,S}} \left\{ \sum_{i \in S} \pi_i F_i(w) - \nu D_S(\pi) \right\} + \frac{\lambda}{2}\|w\|^2 , \tag{A9}$$

From Danskin's theorem, cf. Bertsekas (1999, Proposition B.25), we get the expression of its gradient as

$$\nabla F_{\theta,S}^\nu(w^{(t)}) = \sum_{i \in S} \pi_i^{(t)} \nabla \tilde{F}_i(w^{(t)}) , \tag{A10}$$

where $\pi^{(t)}$ attains the unique argmax in (A9) (see also (15) for the definition).

We define the averaged superquantile as

$$\overline{F}_\theta^\nu(w) = \mathbb{E}_{S \sim U_m}[F_{\theta,S}^\nu(w)] , \tag{A11}$$

where U_m is the uniform distribution over subsets of $[n]$ of size m. Finally, let $\overline{w}^\star = \arg\min_w \overline{F}_\theta^\nu(w)$.

We also define the notion of *client drift* as

$$d^{(t)} := \mathbb{E}_{S \sim U_m}\left[\sum_{i \in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \|w_{i,k}^{(t)} - w^{(t)}\|^2 \,\middle|\, \mathcal{F}_t \right] . \tag{A12}$$

Proof of Theorem 8 We denote $\mathbb{E}_t[\cdot] := \mathbb{E}[\cdot \mid \mathcal{F}_t]$. We expand the update $w^{(t+1)} = w^{(t)} - \gamma \sum_{i \in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \nabla \tilde{F}_i(w_{i,k}^{(t)})$ to get

$$\begin{aligned} \|w^{(t+1)} - \overline{w}^\star\|^2 = \|w^{(t)} - \overline{w}^\star\|^2 &\underbrace{- 2\gamma \sum_{i \in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \left\langle \nabla \tilde{F}_i(w_{i,k}^{(t)}), w^{(t)} - \overline{w}^\star \right\rangle}_{=: \mathcal{T}_1} \\ &\underbrace{+ \gamma^2 \left\| \sum_{i \in S} \pi_i^{(t)} \sum_{k=0}^{\tau-1} \nabla \tilde{F}_i(w_{i,k}^{(t)}) \right\|^2}_{=: \mathcal{T}_2} . \end{aligned} \tag{A13}$$

In order to bound the first order term $\mathcal{T}_1$, we analyze the effect of a single local step of learning rate $\tau\gamma$ rather than τ local steps of learning rate γ. The analogue of the first order term $\mathcal{T}_1$, in this case, would be

$$\mathcal{T}_1' := 2\tau\gamma \sum_{i \in S} \pi_i^{(t)} \left\langle \nabla \tilde{F}_i(w^{(t)}), w^{(t)} - \overline{w}^\star \right\rangle \overset{(A.10)}{=} 2\tau\gamma \left\langle \nabla F_{\theta,S}^\nu(w^{(t)}), w^{(t)} - \overline{w}^\star \right\rangle .$$

The difference $\mathcal{T}_1 - \mathcal{T}_1'$ is the effect of the drift from taking multiple local steps. From here, the proof consists of the following steps:

1. Bound the first order term $\mathcal{T}_1'$,
2. Bound the drift $\mathcal{T}_1 - \mathcal{T}_1'$,
3. Bound the second order term $\mathcal{T}_2$,
4. Combine these to get the effect of one communication round t,
5. Unroll the bound over all communication rounds $t = 1, \dots, T$,
6. Connect optimization on the surrogate $\overline{F}_\theta^\nu$ to the original F_θ,
7. Optimize the choices of the learning rate γ and smoothing parameter ν.

1. Bounding the first order term $\mathcal{T}_1'$. We use the λ-strong convexity (cf. (A25)) of $F_{\theta,S}^\nu$ to get

$$\mathcal{T}_1' \geq 2\tau\gamma \left(F_{\theta,S}^\nu(w^{(t)}) - F_{\theta,S}^\nu(\overline{w}^\star) + \frac{\lambda}{2}\|w^{(t)} - \overline{w}^\star\|^2 \right).$$

Taking an expectation w.r.t. the sampling S (i.e., conditioned on $\mathcal{F}_t$) gives

$$\mathbb{E}_t[\mathcal{T}_1'] \geq 2\tau\gamma \left(\overline{F}_\theta^\nu(w^{(t)}) - \overline{F}_\theta^\nu(\overline{w}^\star) + \frac{\lambda}{2}\|w^{(t)} - \overline{w}^\star\|^2 \right). \tag{A14}$$

2. Bounding the effect of the drift $\mathcal{T}_1 - \mathcal{T}_1'$. The contribution of k^{th} local step to the drift $\mathcal{T}_1 - \mathcal{T}_1'$ can be bounded as

$$\begin{aligned}
&\left| \left\langle \sum_{i\in S} \pi_i^{(t)} \left(\nabla \tilde{F}_i(w_{i,k}^{(t)}) - \nabla \tilde{F}_i(w^{(t)}) \right), w^{(t)} - \overline{w}^\star \right\rangle \right| \\
&\overset{\text{(i)}}{\leq} \frac{\lambda}{4} \left\| w^{(t)} - \overline{w}^\star \right\|^2 + \frac{1}{\lambda} \left\| \sum_{i\in S} \pi_i^{(t)} \left(\nabla \tilde{F}_i(w_{i,k}^{(t)}) - \nabla \tilde{F}_i(w^{(t)}) \right) \right\|^2 \\
&\overset{\text{(ii)}}{\leq} \frac{\lambda}{4} \left\| w^{(t)} - \overline{w}^\star \right\|^2 + \frac{1}{\lambda} \sum_{i\in S} \pi_i^{(t)} \left\| \nabla \tilde{F}_i(w_{i,k}^{(t)}) - \nabla \tilde{F}_i(w^{(t)}) \right\|^2 \\
&\overset{\text{(iii)}}{\leq} \frac{\lambda}{4} \left\| w^{(t)} - \overline{w}^\star \right\|^2 + \frac{(L+\lambda)^2}{\lambda} \sum_{i\in S} \pi_i^{(t)} \left\| w_{i,k}^{(t)} - w^{(t)} \right\|^2.
\end{aligned}$$

Here, we first used (i) the Cauchy–Schwarz inequality, (ii) Jensen's inequality, and (iii) the $(L+\lambda)$-smoothness of $\tilde{F}_i$. Summing this over k, we get the bound

$$\mathbb{E}_t|\mathcal{T}_1 - \mathcal{T}_1'| \leq \frac{\lambda\tau\gamma}{2}\|w^{(t)} - \overline{w}^\star\|^2 + \frac{2\gamma(L+\lambda)^2}{\lambda} d^{(t)}, \tag{A15}$$

where we use the definition of $d^{(t)}$ from (A12).

3. Bounding the second order term $\mathcal{T}_2$. By using the expression (A10) of $\nabla F_{\theta,S}^\nu$, we get

$$\begin{aligned}
&\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla\tilde{F}_i(w_{i,k}^{(t)})\right\|^2 \\
&\quad\le 2\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\left(\nabla\tilde{F}_i(w_{i,k}^{(t)})-\nabla\tilde{F}_i(w^{(t)})\right)\right\|^2+2\left\|\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\nabla\tilde{F}_i(w^{(t)})\right\|^2 \\
&\quad\le 2\tau\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\left\|\nabla\tilde{F}_i(w_{i,k}^{(t)})-\nabla\tilde{F}_i(w^{(t)})\right\|^2+2\tau^2\left\|\nabla F_{\theta,S}^{\nu}(w^{(t)})\right\|^2.
\end{aligned}$$

For the first term, we invoke $(L+\lambda)$-smoothness of $\tilde{F}_i$ and take an expectation to get $2\tau(L+\lambda)^2 d^{(t)}$. For the second term, we have from the definition (A11) of $\overline{F}_\theta^{\nu}$ that $\mathbb{E}_t\left[\nabla F_{\theta,S}^{\nu}(w^{(t)})\right]=\nabla\overline{F}_\theta^{\nu}(w^{(t)})$. Therefore, we can write

$$\begin{aligned}
\mathbb{E}_t\left\|\nabla F_{\theta,S}^{\nu}(w^{(t)})\right\|^2 &= \mathbb{E}_t\left\|\nabla F_{\theta,S}^{\nu}(w^{(t)})-\nabla\overline{F}_\theta^{\nu}(w^{(t)})\right\|^2+\left\|\nabla\overline{F}_\theta^{\nu}(w^{(t)})\right\|^2 \\
&\le \frac{8G^2}{\theta m}+2L'\left(\overline{F}_\theta^{\nu}(w^{(t)})-\overline{F}_\theta^{\nu}(\overline{w}^\star)\right),
\end{aligned}$$

where we invoked Property 9 to bound the variance of the partial superquantile and L' -smoothness of $\overline{F}_\theta^{\nu}$. Overall, this gives us

$$\mathbb{E}_t[\mathcal{T}_2]\le 2\gamma^2\tau(L+\lambda)^2\,d^{(t)}+\frac{16\tau^2\gamma^2G^2}{\theta m}+4\tau^2\gamma^2L'\left(\overline{F}_\theta^{\nu}(w^{(t)})-\overline{F}_\theta^{\nu}(\overline{w}^\star)\right). \tag{A16}$$

4. One-step update Plugging (A14) to (A16) into (A13), we get,

$$\begin{aligned}
\mathbb{E}_t\|w^{(t+1)}-\overline{w}^\star\|^2 \le &\left(1-\frac{\lambda\tau\gamma}{2}\right)\|w^{(t)}-\overline{w}^\star\|^2 \\
&-(2\tau\gamma-4\gamma^2\tau^2L')\left(\overline{F}_\theta^{\nu}(w^{(t)})-\overline{F}_\theta^{\nu}(\overline{w}^\star)\right) \\
&+\frac{16\tau^2\gamma^2G^2}{\theta m}+2\gamma(L+\lambda)^2(\tau\gamma+\lambda^{-1})d^{(t)}.
\end{aligned}$$

Next, we plug in the bound on $d^{(t)}$ from Proposition 12 and simplify some coefficients. First, since $\gamma\le(4\tau L')^{-1}$ we have $2\tau\gamma-4\gamma^2\tau^2L'\ge\tau\gamma$. Likewise, the same condition on γ also implies $\tau\gamma+1/\lambda\le 2/\lambda$. Finally, $\gamma\le\left(8\tau\kappa\sqrt{2\lambda L'}\right)^{-1}$ implies $64L'(L+\lambda)^2\tau^2\gamma^2/\lambda\le 1/2$. As a result, we arrive at the bound

$$\begin{aligned}
\overline{F}_\theta^{\nu}(w^{(t)})-\overline{F}_\theta^{\nu}(\overline{w}^\star)\le &\frac{2}{\tau\gamma}\left(1-\frac{\lambda\tau\gamma}{2}\right)\|w^{(t)}-\overline{w}^\star\|^2-\frac{2}{\tau\gamma}\mathbb{E}_t\|w^{(t+1)}-\overline{w}^\star\|^2 \\
&+\underbrace{\frac{32\tau\gamma G^2}{\theta m}+\frac{64G^2(L+\lambda)^2\tau^2(1-\tau^{-1})\gamma^2}{\lambda}\left(4+\frac{8}{\theta m}\right)}_{=:\mathcal{T}_3}.
\end{aligned}$$

5. Telescoping the bound By telescoping the one-step improvement and convexity, we get, Next, we use convexity to get

$$\begin{aligned}
&\mathbb{E}\left[\overline{F}_\theta^\nu(\overline{w}^{(T)}) - \overline{F}_\theta^\nu(\overline{w}^\star)\right] \\
&\quad\le \frac{1}{\sum_{t=0}^T \left(1-\frac{\lambda\tau\gamma}{2}\right)^{-(1+t)}} \sum_{t=0}^T \left(1-\frac{\lambda\tau\gamma}{2}\right)^{-(1+t)} \mathbb{E}\left[\overline{F}_\theta^\nu(w^{(t)}) - \overline{F}_\theta^\nu(\overline{w}^\star)\right] \\
&\quad\le \frac{2\left\|w^{(0)} - \overline{w}^\star\right\|^2}{\tau\gamma \sum_{t=0}^T \left(1-\frac{\lambda\tau\gamma}{2}\right)^{-(1+t)}} + \mathcal{T}_3\,.
\end{aligned}$$

Now, we can bound the denominator from below with

$$\sum_{t=0}^T \left(1-\frac{\lambda\tau\gamma}{2}\right)^{-(1+t)} \ge \frac{2}{\lambda\tau\gamma}\left(\left(1-\frac{\lambda\tau\gamma}{2}\right)^{-(T+1)} - 1\right) \ge \frac{2}{\tau\gamma\lambda}\left(e^{(T+1)\lambda\tau\gamma} - 1\right).$$

This gives us the final bound

$$\mathbb{E}\left[\overline{F}_\theta^\nu(\overline{w}^{(T)}) - \overline{F}_\theta^\nu(\overline{w}^\star)\right] \le \frac{\lambda}{e^{T\lambda\tau\gamma}-1}\|w^{(0)} - \overline{w}^\star\|^2 + \mathcal{T}_3\,. \tag{A17}$$

6. Translating the results from the surrogate $\overline{F}_\theta^\nu$ to the original F_θ. We optimize the surrogate $\overline{F}_\theta^\nu$ defined on a sample S of clients rather than the full superquantile. The effect of this shows up in both sides of (A17). We bound the left-hand side by noting that the bias introduced by the surrogate is bounded as in Property 9. For the right hand side, we use the λ-strong convexity of F_θ and Property 9 to get

$$\begin{aligned}
\|w^{(0)} - \overline{w}^\star\|^2 &\le 2\|w^{(0)} - w^\star\|^2 + 2\|\overline{w}^\star - w^\star\|^2 \\
&\le 2\|w^{(0)} - w^\star\|^2 + \frac{4}{\lambda}\left(F_\theta(\overline{w}^\star) - F_\theta(w^\star)\right) \\
&\le 2\|w^{(0)} - w^\star\|^2 \\
&\quad + \frac{4}{\lambda}\left(F_\theta(\overline{w}^\star) - \overline{F}_\theta^\nu(\overline{w}^\star) + \overline{F}_\theta^\nu(\overline{w}^\star) - \overline{F}_\theta^\nu(w^\star) + \overline{F}_\theta^\nu(w^\star) - F_\theta(w^\star)\right) \\
&\le 2\|w^{(0)} - w^\star\|^2 + \frac{4}{\lambda}\left(\frac{2B}{\sqrt{\theta m}} + 4\nu\log m\right)\quad,
\end{aligned}$$

since $\overline{F}_\theta^\nu(\overline{w}^\star) - \overline{F}_\theta^\nu(w^\star) \le 0$. Plugging this into (A17) gives us the bound

$$\begin{aligned}
\mathbb{E}\left[F_\theta(\overline{w}^{(T)}) - F_\theta(w^\star)\right] \le\ & \frac{2\lambda}{e^{T\lambda\tau\gamma}-1}\|w^{(0)} - w^\star\|^2 + \frac{32\tau\gamma G^2}{\theta m} \\
&+ \frac{64G^2(L+\lambda)^2\tau^2(1-\tau^{-1})\gamma^2}{\lambda}\left(4 + \frac{8}{\theta m}\right) \\
&+ \left(\frac{2B}{\sqrt{\theta m}} + 4\nu\log m\right)\left(1 + \frac{8}{e^{T\lambda\tau\gamma}-1}\right).
\end{aligned} \tag{A18}$$

7. Hyperparameter optimization To complete the proof from here, it remains to optimize the learning rate γ and the smoothing parameter ν by repeated invocations of Lemma 13.

We start with the learning rate γ. Ignoring absolute constants gives us the bound

$$\mathbb{E}\left[F_\theta(\overline{w}^{(T)})\right] - F_\theta(w^\star) \le \lambda\Delta_0 \exp(-\lambda\tau\Gamma T) + \frac{G^2}{\theta m \lambda T}\left(1 \vee \log \frac{\lambda^2\Delta_0\theta m}{G^2} T\right) + \frac{G^2\kappa^2}{\lambda T^2}(1-\tau^{-1})\left(1 \vee \log \frac{\lambda^2\Delta_0 T^2}{G^2\kappa^2}\right)^2 + \frac{B}{\sqrt{\theta m}} + \nu \log m\,, \tag{A19}$$

where we take

$$\Gamma = \min\left\{\frac{\sqrt{\lambda}}{8\tau(L+\lambda)\sqrt{2L'}}, \frac{1}{4\tau L'}\right\}.$$

This application of Lemma 13 requires $\lambda\tau\Gamma T \ge 1$, which we will ensure later, based on the choice of ν. Recall that Γ depends on L', which itself depends on ν as $L' = L + \lambda + G^2/\nu$.

Next, we set ν. We will require that $\nu \le G^2/(\lambda\kappa)$, so that the two terms from (A19) that depend on ν can be bounded as

$$\lambda\Delta_0 \exp(-\lambda\tau\Gamma T) + \nu \log m \le \lambda\Delta_0 \exp\left(-\frac{T}{16\kappa\sqrt{\frac{G^2}{\lambda\nu}} \vee \frac{8G^2}{\lambda\nu}}\right) + \nu \log m\,. \tag{A20}$$

To simplify the expression, we substitute

$$\frac{1}{\delta} = \max\left\{16\kappa\sqrt{\frac{G^2}{\lambda\nu}}, \frac{8G^2}{\lambda\nu}\right\} \quad \Longleftrightarrow \quad \nu = \max\left\{\frac{256\kappa^2G^2\delta^2}{\lambda}, \frac{8G^2\delta}{\lambda}\right\}.$$

The bound $\nu \le G^2/(\lambda\kappa)$ translates to the upper bound $\delta \le (16\kappa^{3/2})^{-1}$. Therefore, the right hand side of (A20) can be further upper bounded by using $\max\{a,b\} \le a+b$ as

$$\lambda\Delta_0 \exp(-\delta T) + \frac{8G^2\log m}{\lambda}\,\delta + \frac{256G^2\kappa^2G^2\log m}{\lambda}\,\delta^2\,.$$

We now invoke Lemma 13 under the condition $T \ge 16\kappa^{3/2}$. We set δ as specified by Lemma 13 — this gives us the choices of the smoothing parameter ν and learning rate γ. Plugging this into (A19) gives the bound of the theorem. Finally, to complete the proof, it can be verified that the condition $\lambda\tau\Gamma T \ge 1$ is guaranteed by $T \ge 16\kappa^{3/2}$. □

A.4: Intermediate results

We present some prerequisites and some intermediate results which are required in the convergence proofs.

Note that for any $S \subset [n]$ of size m, the partial superquantile is differentiable at w with :

$$\nabla F^{\nu}_{\theta,S}(w) = \sum_{i\in S} \pi_i^\star \nabla \tilde{F}_i(w) \tag{A21}$$

where $\pi^\star$ denotes the solution to the maximization

$$F^{\nu}_{\theta,S}(w) = \max_{\pi \in \mathcal{P}_{\theta,S}} \sum_{i \in S} \pi_i \tilde{F}_i(w) - \nu D_S(\pi)\,.$$

Bias and variance of the partial superquantile We use the partial superquantile defined on a subset $S \subset [n]$ to approximate the full superquantile. We start with the quality of this approximation.

Property 9 *Let U_m denote the uniform distribution over all subsets of $[n]$ of size m. For any $w \in \mathbb{R}^d$, we have*

$$\left|\overline{F}^{\nu}_{\theta}(w) - F_{\theta}(w)\right| \leq \frac{B}{\sqrt{\theta m}} + 2\nu \log m\,,$$
$$\mathbb{E}_{S\sim U_m}\left\|\nabla F^{\nu}_{\theta,S}(w) - \nabla \overline{F}^{\nu}_{\theta}(w)\right\|^2 \leq \frac{8G^2}{\theta m}\,.$$

Smoothing and smoothness constants The following result is standard, cf. Beck & Teboulle (2012, Theorem 4.1, Lemma 4.2).

Property 10 *For every $\nu > 0$, we have that $F^{\nu}_{\theta,S}$ and $\overline{F}^{\nu}_{\theta,S}$ are L'-smooth with $L' = L + \lambda + \frac{G^2}{\nu}$*
.

Bounding the gradient dissimilarity. Bounding of the variance of gradient estimators is a key assumption in the analysis of stochastic gradient methods (see e.g. the textbook (Bottou et al., 2018)). In the centralized setting, when a stochastic objective $\mathbb{E}_{\xi}[f(w,\xi)]$, it is standard to assume for a given estimator g_w of $\nabla_w \mathbb{E} f(w,\xi)$ that there exists some constants $M_1, M_2 > 0$ such that for all $w \in \mathbb{R}^d$,

$$\left\|\mathbb{E}\left[g_w\right]\right\|^2 \leq M_1 \quad \text{or} \quad \left\|\mathbb{E}\left[g_w\right]\right\|^2 \leq M_1 + M_2\left\|\nabla_w \mathbb{E}\left[f(w,\xi\right]\right\|^2.$$

In the federated setting, the use of a subset $S \subset [n]$ of clients in each round induces noise on the estimation of the average gradient over the whole network. Thus, such assumption translates into a *bound on the gradient dissimilarity* among the clients (Karimireddy et al., 2020; Wang et al., 2019):

$$\frac{1}{n}\sum_{i\in[n]} \left\|\nabla \tilde{F}_i(w)\right\|^2 \leq M_1 + M_2\left\|\frac{1}{n}\sum_{i\in[n]} \nabla \tilde{F}_i(w)\right\|^2.$$

In this work, we also consider the minimization of the global loss F^{ν}_{θ} by a stochastic algorithm based on partial participation of the clients, with the additional difficulty that we only have access to a biased estimator $\overline{F}^{\nu}_{\theta}$ of the loss F^{ν}_{θ} and its gradient. In particular, the adaptive reweighting of the clients selected at each round does not permit the direct use of such an assumption. We show instead in the next lemma that the variance of the stochastic gradient estimator can also be bounded, thanks to the Lipschitz assumption.

Proposition 11 (Gradient dissimilarity) *Consider the quantities $\pi^{(t)}, w^{(t)}$ from Algorithm* 3. *We have*,

$$\mathbb{E}\left[\sum_{i\in S}\pi_i^{(t)}\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2\middle|\mathcal{F}_t\right]\leq\left(4+\frac{8}{\theta m}\right)G^2+\left\|\nabla\overline{F}_\theta^\nu(w^{(t)})\right\|^2.$$

Proof We drop the superscript t throughout this proof. By centering the second moment (cf. (A24)), we have:

$$\begin{aligned}\sum_{i\in S}\pi_i\left\|\nabla\tilde{F}_j(w)\right\|^2&=\sum_{i\in S}\pi_i\left\|\nabla\tilde{F}_i(w)-\nabla F_{\theta,S}^\nu(w)\right\|^2+\left\|\nabla F_{\theta,S}^\nu(w)\right\|^2\\&=\sum_{i\in S}\pi_i\left\|\nabla F_i(w)-\sum_{j\in S}\pi_j\nabla F_j(w)\right\|^2+\left\|\nabla F_{\theta,S}^\nu(w)\right\|^2.\end{aligned}$$

Now since the weights π_i sum to one, we may use the convexity of $\|\cdot\|^2$ to get:

$$\sum_{i\in S}\pi_i\left\|\nabla\tilde{F}_j(w)\right\|^2\leq\sum_{i,j\in S}\pi_i\pi_j\left\|\nabla F_j(w)-\nabla F_i(w)\right\|^2+\left\|\nabla F_{\theta,S}^\nu(w)\right\|^2.$$

The squared triangle inequality (cf. (A23)) together with the Lipschitz assumption on the functions F_i yields:

$$\begin{aligned}\sum_{i\in S}\pi_i\left\|\nabla\tilde{F}_i(w)\right\|^2&\leq 2\sum_{i,j\in S}\pi_i\pi_i\left(\left\|\nabla F_i(w)\right\|^2+\left\|\nabla F_j(w)\right\|^2\right)+\left\|\nabla F_{\theta,S}^\nu(w)\right\|^2\\&\leq 4\,G^2+\left\|\nabla F_{\theta,S}^\nu(w)\right\|^2.\end{aligned}$$

Thus, taking an expectation over $S\sim U_m$ gives

$$\mathbb{E}\left[\sum_{i\in S}\pi_i\left\|\nabla\tilde{F}_j(w)\right\|^2\middle|\mathcal{F}_t\right]\leq 4\,G^2+\mathbb{E}_{S\sim U_m}\left[\left\|\nabla F_{\theta,S}^\nu(w)\right\|^2\right].$$

By centering (cf. (A24)), we get,

$$\begin{aligned}\mathbb{E}\left[\sum_{i\in S}\pi_i\left\|\nabla\tilde{F}_i(w)\right\|^2\middle|\mathcal{F}_t\right]\leq&\,4\,G^2+\left\|\nabla\overline{F}_\theta^\nu(w)\right\|^2\\&+\mathbb{E}\left[\left\|\nabla F_{\theta,S}^\nu(w)-\nabla\overline{F}_\theta^\nu(w)\right\|^2\middle|\mathcal{F}_t\right].\end{aligned}\tag{A22}$$

Finally, substituting the variance bound from Property 9 into (A22) yields the stated result. □

Bounding the Client Drift. During federated learning, each client takes multiple local steps. This causes the resulting update to be a biased estimator of a descent direction for the global objective. This phenomenon has been referred to as "client drift" (Li et al., 2020; Karimireddy et al., 2020). Current proof techniques rely on treating this as a "noise" term that is to be controlled. In the context of this work, the reweighting by $\pi^{(t)}$ requires us to adapt this typical definition of client drift to our setting. In particular, recall that we define the client drift $d^{(t)}$ in outer iteration t of the algorithm as

$$d^{(t)} := \mathbb{E}_{S\sim U_m}\left[\sum_{i\in S}\pi_i^{(t)}\sum_{k=0}^{\tau-1}\|w_{i,k}^{(t)}-w^{(t)}\|^2\,\middle|\,\mathcal{F}_t\right].$$

Proposition 12 (Client Drift) *If* $\gamma \le \frac{1}{4\tau(L+\lambda)}$, *we have for any* $t \ge 0$ *that*

$$d^{(t)} \le 8\tau^2(\tau-1)\gamma^2\left(\left(4+\frac{8}{\theta m}\right)G^2 + 2L'\left(\overline{F}_\theta^\nu(w^{(t)}) - \overline{F}_\theta^\nu(\overline{w}^\star)\right)\right).$$

Furthermore, if $\lambda = 0$, *we have the bound*

$$d^{(t)} \le 8\tau^2(\tau-1)\gamma^2G^2\,.$$

The last bound also works without smoothing, i.e., $\nu = 0$.

Proof We absorb the regularization into the superquantile by defining $\tilde{F}_i(w) = F_i(w) + (\lambda/2)\|w\|^2$. If $\tau = 1$, there is nothing to prove as both sides of the inequality are 0. We assume now that $\tau > 1$. Let us first fix $S \subset [n]$ of size $|S| = m$. For any $k \in S$ and $j \in \{1,\dots,\tau-1\}$, by the squared triangle inequality (cf. (A23)), we have:

$$\begin{aligned}\left\|w_{i,k}^{(t)} - w^{(t)}\right\|^2 &= \left\|w_{i,k-1}^{(t)} - \gamma\nabla\tilde{F}_i(w_{i,k-1}^{(t)}) - w^{(t)}\right\|^2\\ &\le \left(1+\frac{1}{\tau-1}\right)\left\|w_{i,k-1}^{(t)} - w^{(t)}\right\|^2 + \tau\gamma^2\left\|\nabla\tilde{F}_i(w_{i,k-1}^{(t)})\right\|^2.\end{aligned}$$

The squared triangle inequality (cf. (A23)) together with the smoothness of the local losses gives:

$$\begin{aligned}\left\|w_{i,k}^{(t)} - w^{(t)}\right\|^2 &\le \left(1+\frac{1}{\tau-1}\right)\left\|w_{i,k-1}^{(t)} - w^{(t)}\right\|^2\\ &\quad + 2\tau\gamma^2\left(\left\|\nabla\tilde{F}_i(w_{i,k-1}^{(t)}) - \nabla\tilde{F}_i(w^{(t)})\right\|^2 + \left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2\right)\\ &\le \left(1+\frac{1}{\tau-1}\right)\left\|w_{i,k-1}^{(t)} - w^{(t)}\right\|^2\\ &\quad + 2\tau\gamma^2(L+\lambda)^2\left\|w_{i,k-1}^{(t)} - w^{(t)}\right\|^2 + 2\tau\gamma^2\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2.\end{aligned}$$

Hence, for $\gamma \le \frac{1}{4\tau(L+\lambda)}$, we get:

$$\left\|w_{i,k}^{(t)} - w^{(t)}\right\|^2 \le \left(1+\frac{2}{\tau-1}\right)\left\|w_{i,k-1}^{(t)} - w^{(t)}\right\|^2 + 2\tau\gamma^2\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2.$$

Unrolling this recursion yields for any $j \le \tau - 1$

$$\begin{aligned}\left\|w_{i,k}^{(t)} - w^{(t)}\right\|^2 &\leq \sum_{i=0}^{k-1}\left(1+\frac{2}{\tau-1}\right)^i\left(2\tau\gamma^2\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2\right)\\ &\leq \frac{\tau-1}{2}\left(1+\frac{2}{\tau-1}\right)^k\left(2\tau\gamma^2\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2\right)\\ &\leq \frac{\tau-1}{2}\left(1+\frac{2}{\tau-1}\right)^{\tau-1}\left(2\tau\gamma^2\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2\right)\\ &\leq 8\tau(\tau-1)\gamma^2\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2,\end{aligned}$$

where we use $(1+2/x)^x \leq e^2 < 8$ for any $x > 0$. If $\lambda = 0$ we have that $\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2 = \left\|\nabla F_i(w^{(t)})\right\|^2 \leq G^2$ since F_i is G-Lipschitz; this gives us the final bound in the statement. When $\lambda \neq 0$, this does not hold. In this case, we apply Proposition 11 to get

$$\begin{aligned}d^{(t)} &\leq 8\tau^2(\tau-1)\gamma^2\,\mathbb{E}_{S\sim U_m}\left[\sum_{i\in S}\pi_i^{(t)}\left\|\nabla\tilde{F}_i(w^{(t)})\right\|^2\,\middle|\,\mathcal{F}^{(t)}\right]\\ &\leq 8\tau^2(\tau-1)\gamma^2\left(\left(4+\frac{8}{\theta m}\right)G^2+\left\|\nabla\overline{F}_\theta^{\nu}(w^{(t)})\right\|^2\right).\end{aligned}$$

Invoking smoothness (cf. (A26)) completes the proof. □

A.5: Useful inequalities and technical results

We recall a few standard inequalities:

- Squared Triangle inequality: For any $x, y \in \mathbb{R}^d$ and $\alpha > 0$ we have:

$$\|x+y\|^2 \leq (1+\alpha)\|x\|^2 + \left(1+\frac{1}{\alpha}\right)\|y\|^2. \tag{A23}$$

- Centering the second moment: For any $\mathbb{R}^d$-valued random vector X such that $\mathbb{E}\|X\|^2 < \infty$,

$$\mathbb{E}\|X\|^2 = \mathbb{E}\left\|X - \mathbb{E}[X]\right\|^2 + \left\|\mathbb{E}[X]\right\|^2. \tag{A24}$$

- Strong convexity: Let $F : \mathbb{R}^d \to \mathbb{R}$ be μ-strongly convex. Then for any $x, y \in \mathbb{R}^d$, we have:

$$\langle\nabla F(x), x-y\rangle \geq F(x) - F(y) + \frac{\mu}{2}\|x-y\|^2. \tag{A25}$$

- Smoothness: Let $F : \mathbb{R}^d \to \mathbb{R}$ be L-smooth and let $F^\star$ be the minimum value of F (assuming it exists). Then for any $x \in \mathbb{R}^d$, we have:

$$\|\nabla F(x)\|^2 \leq 2L\big(F(x) - F^\star\big). \tag{A26}$$

Lemma 13 *Consider the maps $\varphi, \psi : (0, \Gamma] \to \mathbb{R}_+$ given by*

$$\varphi(\gamma) = \frac{A}{\exp(\lambda\gamma T) - 1} + B\gamma + C\gamma^2\,, \quad \psi(\gamma) = 2A\exp(-\lambda\gamma T) + B\gamma + C\gamma^2\,,$$

where $\lambda, \Gamma, A, B, C, T > 0$ *are given and* $\lambda\Gamma \le 1$. *If* $T \ge (\lambda\Gamma)^{-1}$, *then, we have*,

$$\varphi(\gamma^\star) \le \psi(\gamma^\star) \le 2A\exp(-\lambda\Gamma T) + \frac{3B}{\lambda T}\left(1 \vee \log\frac{A\lambda T}{B}\right) + \frac{3C}{\lambda^2 T^2}\left(1 \vee \log\frac{A\lambda^2 T^2}{C}\right)^2,$$

where $\gamma^\star$ *is given by*

$$\gamma^\star = \min\left\{\Gamma, \frac{1}{\lambda T}\left(1 \vee \log\frac{A\lambda T}{B}\right), \frac{1}{\lambda T}\left(1 \vee \log\frac{A\lambda^2 T^2}{C}\right)\right\}.$$

Furthermore, we also have that $\big(\exp(\lambda\gamma^\star T) - 1\big)^{-1} \le 1$.

Proof Since $\lambda\Gamma T \ge 1$, we have that $\lambda\gamma^\star T \ge 1$. Then, $\exp(-\lambda\gamma^\star T) \le \exp(-1) < 1/2$ so that

$$\frac{1}{\exp(\lambda\gamma^\star T) - 1} = \frac{\exp(-\lambda\gamma^\star T)}{1 - \exp(-\lambda\gamma^\star T)} \le 2\exp(-\lambda\gamma^\star T) \le 1\,.$$

Therefore, we have,

$$\varphi(\gamma^\star) \le 2A\exp(-\lambda\gamma^\star T) + B\gamma^\star + C(\gamma^\star)^2 = \psi(\gamma^\star)\,.$$

Next, define $\gamma_1 = (\lambda T)^{-1}\log(1 \vee A\lambda T/B)$ and $\gamma_2 = (\lambda T)^{-1}\log(1 \vee A\lambda^2 T^2/C)$, so that $\gamma^\star = \min\{\Gamma, \gamma_1, \gamma_2\}$. We have three cases:

- If $\gamma^\star = \Gamma$, we have that $\Gamma \le \gamma_1$ and $\Gamma \le \gamma_2$ so that
  $$\psi(\gamma^\star) = 2A\exp(-\lambda\Gamma T) + B\Gamma + C\Gamma^2 \le 2A\exp(-\lambda\Gamma T) + B\gamma_1 + C\gamma_2^2\,.$$
- If $\gamma^\star = \gamma_1$, we have $\gamma_1 \le \gamma_2$. In this case,
  $$\psi(\gamma^\star) = A\exp(-\lambda\gamma_1 T) + B\gamma_1 + C\gamma_1^2 \le \frac{2B}{\lambda T} + \frac{B}{\lambda T}\left(1 \vee \log\frac{A\lambda T}{B}\right) + C\gamma_2^2\,.$$
- If $\gamma^\star = \gamma_2$, we have $\gamma_2 \le \gamma_1$, so that
  $$\psi(\gamma^\star) = 2A\exp(-\lambda\gamma_2 T) + B\gamma_2 + C\gamma_2^2 \le \frac{2C}{\lambda^2 T^2} + B\gamma_1 + \frac{C}{\lambda^2 T^2}\left(1 \vee \log\frac{A\lambda^2 T^2}{C}\right)^2.$$

□

The proof of the next lemma is elementary and is omitted.

Lemma 14 *Consider the map* $\varphi : (0, \Gamma] \to \mathbb{R}_+$ *given by*

$$\varphi(\gamma) = \frac{A}{\gamma T} + B\gamma + C\gamma^2\,,$$

where $\Gamma, A, B, C > 0$ *are given. Then, we have*,

$$\varphi(\gamma^\star) \le \frac{A}{\Gamma T} + 2\left(\frac{AB}{T}\right)^{1/2} + 2C^{1/3}\left(\frac{A}{T}\right)^{2/3},$$

where $\gamma^\star$ *is given by*

$$\gamma^\star = \min\left\{\Gamma, \sqrt{\frac{A}{BT}}, \left(\frac{A}{CT}\right)^{1/3}\right\}.$$

Appendix B: Privacy analysis

B.1: Preliminaries

The discrete Gaussian mechanism was introduced in Canonne et al. (2020) as an extension of the Gaussian mechanism to integer data. A random variable ξ is said to satisfy the discrete Gaussian distribution with mean μ and variance proxy σ^2 if

$$\mathbb{P}(\xi = n) = C\,\exp\left(-\frac{(n-\mu)^2}{2\sigma^2}\right) \quad \text{for all } n \in \mathbb{Z},$$

where C is an appropriate normalizing constant. We denote it by $\mathcal{N}_{\mathbb{Z}}(\mu, \sigma^2)$. We need the following property of the discrete Gaussian.

Property 15 *Let* ξ *be distributed according to* $\mathcal{N}_{\mathbb{Z}}(\mu, \sigma^2)$. *Then,* $\mathbb{E}[\xi] = \mu$. *Furthermore, if* $\mu = 0$, *then* ξ *is sub-Gaussian with variance proxy* σ^2, *i.e.*, $\mathbb{E}[\exp(\lambda\xi)] \le \exp(\lambda^2\sigma^2/2)$ *for all* $\lambda > 0$.

B.2: Privacy-utility analysis of quantile computation

We now give the full proof of Theorem 6.

Proof of Theorem 6 We start by defining and controlling the probabilities of some events. Throughout, let $\delta > 0$ be fixed. Define the event

$$E_{\text{mod}} = \bigcap_{i=1}^{n} \bigcap_{r=0}^{\log_2 b - 1} \bigcap_{j=1}^{b/2^r} \left\{-\frac{M-2}{2n} \le cx_i(r,j) + \xi_i(r,j) \le \frac{M-2}{2n}\right\}. \tag{B27}$$

Note that under E_{mod}, no modular wraparound occurs in the algorithm. Thus, for all valid levels r and indices j, we have $\tilde{x}_i(r,j) = cx_i(r,j) + \xi_i(r,j)$ and

$$\hat{h}(r,j) = \sum_{i=1}^{n} \frac{\tilde{x}_i(r,j)}{c} = \sum_{i=1}^{n}\left(x_i(r,j) + \frac{\xi_i(r,j)}{c}\right).$$

Next, we define the event

$$E_{\text{diff}} = \bigcap_{j=1}^{b} \left\{ \left| H(j) - \hat{H}(j) \right| \le \sqrt{2\sigma^2 n \log_2(b) \log(4b/\delta)} \right\}. \tag{B28}$$

We will show later that E_{mod} and E_{diff} holds with high probability; for now, we assume that they hold.

Privacy analysis We start by establishing the sensitivity of the sum query over x_i's as $\log_2 b$, one for each level in the hierarchical histogram. Define the input space $\mathcal{X}$ to be the space of hierarchical histograms with one non-zero entry in the leaf nodes with consistent counts (i.e., the count of a parent node in the hierarchical histogram equals the sum of its child nodes). Let $\mathcal{X}^* = \cup_{m=1}^{\infty} \mathcal{X}^m$ denote the set of all sequences of elements of $\mathcal{X}$. We consider the rescaled sum query $A((x_1, \dots, x_N)) = \sum_{i=1}^{n} c x_i$. The L_2 sensitivity $S(A)$ of this query A is supremum over all $X \in \mathcal{X}^*$ and X' which is obtained by concatenating x' to X:

$$S(A) = \sup_{X, X'} \left\| A(X) - A(X') \right\|_2 = \sup_{x' \in \mathcal{X}} c \left\| x' \right\|_2 = c \log_2 b \,.$$

We invoke the privacy bound of sums of discrete Gaussians (Lemma 18) to claim that an algorithm $\mathcal{A}$ returning $A(x) + \sum_{i=1}^{n} \xi_i$ satisfies $(1/2)\epsilon^2$-concentrated DP where ϵ is as in the theorem statement. The fact that the quantile and all further functions of it remain private follows from the post-processing property of DP (also known as the data-processing inequality).

Utility analysis Using the triangle inequality, we get,

$$\begin{aligned}
\Delta_\theta(\hat{H}, H) &= \left| \frac{H\left(j^*_\theta(\hat{H})\right)}{n} - (1-\theta) \right| \\
&\le \frac{1}{n} \left| H\left(j^*_\theta(\hat{H})\right) - \hat{H}\left(j^*_\theta(\hat{H})\right) \right| + \left| \frac{1}{n} \hat{H}\left(j^*_\theta(\hat{H})\right) - (1-\theta) \right| \\
&\le \max_{j \in [b]} \left\{ \frac{1}{n} \left| H(j) - \hat{H}(j) \right| \right\} + R^*_\theta(\hat{H}) \,.
\end{aligned}$$

The first term is bounded under E_{diff}, and this gives the utility bound.

Bounding the failure probability The algorithm fails when at least one of E_{mod} or E_{diff} fail to hold. We have from Claim 16 that $\mathbb{P}(E_{\text{mod}}) \ge 1 - \delta/4$ under the given assumptions. From, Claim 17, we have $\mathbb{P}(E_{\text{diff}} | E_{\text{mod}}) \ge 1 - \delta/2$. We bound the total failure probability of the algorithm with a union bound as

$$\begin{aligned}
\mathbb{P}(\bar{E}_{\text{diff}} \cup \bar{E}_{\text{mod}}) &\le \mathbb{P}(\bar{E}_{\text{diff}} | E_{\text{mod}}) \, \mathbb{P}(E_{\text{mod}}) + \mathbb{P}(\bar{E}_{\text{diff}} | \bar{E}_{\text{mod}}) \, \mathbb{P}(\bar{E}_{\text{mod}}) + \mathbb{P}(\bar{E}_{\text{mod}}) \\
&\le \mathbb{P}(\bar{E}_{\text{diff}} | E_{\text{mod}}) + 2 \, \mathbb{P}(\bar{E}_{\text{mod}}) \le \delta \,.
\end{aligned}$$

□

We state and prove bounds on probabilities of the events $E_{\text{mod}}, E_{\text{diff}}$ defined above.

Claim 16 *If* $M \ge 2 + 2cn + 2n\sqrt{2\sigma^2 \log(16nb/\delta)}$, *then* $\mathbb{P}(E_{\text{mod}}) \ge 1 - \delta/4$.

Proof Each discrete Gaussian random variable $\xi_i(r, j)$ is centered and sub-Gaussian with variance proxy σ^2 (cf. Property 15). A Cramér-Chernoff bound (cf. Lemma 19) gives us the exponential tail bound

$$\mathbb{P}\left(|\xi_i(r,j)| > \sqrt{2\sigma^2 \log(16nb/\delta)}\right) \le \frac{\delta}{8nb}.$$

Applying the union bound over $i = 1, \ldots, n$ and the $2b - 2$ nodes in each hierarchical histogram x_i (each node corresponding to one (r, j) pair) completes the proof. □

Claim 17 *We have* $\mathbb{P}(E_{\text{diff}}|E_{\text{mod}}) \ge 1 - \delta/2$.

Proof Under E_{mod}, we have that $\hat{H}(j) = H(j) + \sum_{i=1}^{n} \sum_{(r,o)\in P_j} \xi_i(r,o)$, where P_j is the maximal dyadic partition of $[1, j]$ with $|P_j| \le \log_2 b$. Thus, $\zeta_j := \hat{H}(j) - H(j)$ is sub-Gaussian with variance proxy $n|P_j|\sigma^2 \le n\sigma^2 \log_2 b$. A Cramér-Chernoff bound (cf. Lemma 19) gives us

$$\mathbb{P}\left(|\zeta_j| > \sqrt{2\sigma^2 n \log_2(b) \log(4b/\delta)}\right) \le \frac{\delta}{2b}.$$

Applying a union bound over $j = 1, \ldots, b$ completes the proof. □

B.3: Useful results

The distributed discrete Gaussian mechanism gets privacy guarantees by adding a sum of discrete Gaussian random variables. We give a bound on its privacy. The following lemma is due to Kairouz et al. (2021).

Lemma 18 (Privacy of Sum of Discrete Gaussians) *Fix* $\sigma \ge 1/2$. *Let* $A : \mathcal{X} \to^d$ *be a deterministic algorithm with* ℓ_2*-sensitivity* S *for some input space* $\mathcal{X}$. *Define a randomized algorithm* $\mathcal{A}$, *which when given an input* $x \in \mathcal{X}$, *samples* $\xi_1, \ldots, \xi_n \sim \mathcal{N}_{\mathbb{Z}}(0, \sigma^2 I_d)$ *and returns* $A(x) + \sum_{i=1}^{n} \xi_i$. *Then,* $\mathcal{A}$ *satisfies* $\varepsilon^2/2$*-concentrated DP with*

$$\varepsilon = \min\left\{\sqrt{\frac{S^2}{n\sigma^2} + \frac{\psi d}{2}}, \frac{S}{\sqrt{n}\sigma} + \psi\sqrt{d}\right\},$$

where $\psi = 10\sum_{i=1}^{n-1} \exp\left(-2\pi^2\sigma^2 i/(k+1)\right) \le 10(n-1)\exp(-2\pi^2\sigma^2)$.

Next, we record a standard concentration result.

Lemma 19 (Cramér-Chernoff) *Let* ξ *be a real-valued and centered sub-Gaussian random variable with variance proxy* σ^2, *i.e.*, $\mathbb{E}[\xi] = 0$ *and* $\mathbb{E}[\exp(\lambda\xi)] \le \exp(\lambda^2\sigma^2/2)$ *for all* $\lambda > 0$. *Then, we have for any* $t > 0$,

$$\mathbb{P}(|\xi| > t) \le 2\exp\left(-\frac{t^2}{2\sigma^2}\right).$$

Appendix C: Numerical experiments: extended results

We conduct our experiments on two datasets from computer vision and natural language processing. These datasets contain a natural, non-iid split of data which is reflective of data heterogeneity encountered in federated learning. In this section, we describe in detail the experimental setup and the results. Here is its outline:

- Section C.1 describes the datasets and tasks.
- Section C.2 presents the algorithm and the hyperparameters used.
- Section C.3 details the evaluation methodology.
- Section C.4 gives the experimental comparison of Δ-FL to baselines.

Since each client has a finite number of datapoints in the examples below, we let its probability distribution q_i to be the empirical distribution over the available examples, and the weight α_i to be proportional to the number of datapoints available on the client.

C.1: Datasets and tasks

We use the two following datasets, described in detail below. The data was preprocessed using LEAF (Caldas et al., 2018).

EMNIST for handwritten-letter recognition

Dataset EMNIST (Cohen et al., 2017) is a character recognition dataset. This dataset contains images of handwritten digits or letters, labeled with their identification (a-z, A-Z, 0-9). The images are grey-scaled pictures of $28 \times 28 = 784$ pixels.

Train and test devices Each image is also annotated with the "writer" of the image, i.e., the human subject who hand-wrote the digit/letter during the data collection process. Each client corresponds to one writer. From this set of clients, we discard all clients containing less than 100 images. The remaining clients were partitioned into two groups — 1730 training and 1730 testing clients. For each experiment, we subsampled 865 training and 865 testing clients for computational tractability, where the sampled clients vary based on the random seed of each experiment.

Model We consider the following models for this task.

- **Linear model**: We use a linear softmax regression model. In this case, each F_i is convex. We train parameters $w \in \mathbb{R}^{62\times 784}$. Given an input image $x \in \mathbb{R}^{784}$, the score of each class $c \in [62]$ is the dot product $\langle w_c, x\rangle$. The probability p_c assigned to each class is then computed as a softmax: $p_c = \exp\langle w_c, x\rangle / \sum_{c'} \exp\langle w_{c'}, x\rangle$. The prediction for a given image is then the class with the highest probability.
- **ConvNet**: We also consider a convolutional neural network with two convolutional layers with max-pooling and one fully connected layer (F.C) which outputs a vector in $\mathbb{R}^{62}$. The outputs of the ConvNet are scores with respect to each class. They are also used with a softmax operation to compute probabilities.

The loss used to train both models is the multinomial logistic loss $L(p, y) = -\log p_y$ where p denotes the vector of probabilities computed by the model and p_y denotes its y^{th} component. In the convex case, we add a quadratic regularization term of the form $(\lambda/2)\|w\|_2^2$.

Sent140 for sentiment analysis

Dataset Sent140 (Go et al., 2009) is a text dataset of 1,600,498 tweets produced by 660,120 Twitter accounts. Each tweet is represented by a character string with emojis redacted. Each tweet is labeled with a binary sentiment reaction (i.e., positive or negative), which is inferred based on the emojis in the original tweet.

Train and test devices Each client represents a Twitter account and contains only tweets published by this account. From this set of clients, we discarded all clients containing less than 50 tweets and split the 877 remaining clients into a train set and a test set of sizes 438 and 439 respectively. This split was held fixed for all experiments. Each word in the tweet is encoded by its 50-dimensional GloVe embedding (Pennington et al., 2014).

Model We consider the following models.

- **Linear model**: We consider a l_2-regularized linear logistic regression model where the parameter vector w is of dimension 50. In this case, each F_i is convex. We summarize each tweet by the average of the GloVe embeddings of the words of the tweet.
- **RNN**: The nonconvex model is a Long Short Term Memory (LSTM) model (Hochreiter & Schmidhuber, 1997) built on the GloVe embeddings of the words of the tweet. The hidden dimension of the LSTM is the same as the embedding dimension, i.e., 50. We refer to it as "RNN".

The loss function is the binary logistic loss.

C.2: Algorithms and hyperparameters

Algorithm and baselines

The proposed Δ-FL is run for three values of $\theta \in \{0.8, 0.5, 0.1\}$. We compare it to the following baselines:

- FedAvg (McMahan et al., 2017): It is the de facto standard for the vanilla federated learning objective.
- FedAvg, θ: We also consider FedAvg with a random client subselection step: local updates are run on a fraction of the initial number of clients randomly selected per round. For each dataset, we try three values, corresponding to the average number of clients selected by Δ-FL for the three values of θ used.

  In the main paper, we report as FedAvg-Sub the performance of FedAvg, θ with $\theta \in \{0.8, 0.5, 0.1\}$ which gives the best performance on Δ-FL (i.e., lowest 90th percentile of test misclassification error). Here we report numbers for all values of θ considered.
- FedProx (Li et al., 2020): It augments FedAvg with a proximal term but still minimizes the vanilla federated learning objective.
- q-FFL (Li et al., 2020): It raises the per-client losses to the power $(1 + q)$, where $q \geq 0$ is a parameter, in order to focus on clients with higher loss. We run q-FFL for values of q in $\{10^j, j \in \{-3, \ldots, 1\}\}$.
- AFL (Mohri et al., 2019): It aims to minimize the worst per-client loss. We implement it as an asymptotic version of q-FFL, using a large value of q, as this was found to yield better convergence with comparable performance (Li et al., 2020). In the experiments, we take $q = 10.0$.

The experiments are conducted on the datasets described in Section C.1.

Hyperparameters.

Rounds We measure the progress of each algorithm by the number of calls to secure aggregation routine for weight vectors, i.e., the number of communication rounds.

For the experiments, we choose the number of communication rounds depending on the convergence of the optimization for FedAvg. For the EMNIST dataset, we run the algorithm for 3000 communication rounds with the linear model and 1000 for the ConvNet. For the Sent140 dataset, we run the 1000 communication rounds for the linear model and 600 for the RNN.

Devices per round We choose the same number of clients per round for each method, with the exception of $FedAvg, \theta$.

All clients are assumed to be available and selections are made uniformly at random. In particular, we select 100 clients per round for all experiments with the exception of Sent140 RNN for which we used 50 clients per round.

Local updates and minibatch size Each selected client locally runs 1 epoch of minibatch stochastic gradient descent locally.

We used the default mini-batch of 10 for all experiments (McMahan et al., 2017), except for 16 for EMNIST ConvNet. This is because the latter experiments were run using on a GPU, as we describe in the section on the hardware.

Learning rate scheme We now describe the learning rate γ_t used during *LocalUpdate*. For the linear model, we used a constant fixed learning rate $\gamma_t \equiv \gamma_0$, while for the neural network models, we used a step decay scheme of the learning rate $\gamma_t = \gamma_0 c^{-\lfloor t/t_0 \rfloor}$ for some t_0 where γ_0 and $0 < c \leq 1$ are tuned. We tuned the learning rates only for the baseline FedAvg and used the same learning rate for the other baselines and Δ-FL at all values of θ.

For the neural network models, we fixed t_0 so that the learning rate decayed once or twice during the fixed time horizon T. In particular, we used $t_0 = 400$ for EMNIST ConvNet (where $T = 1000$) and $t_0 = 200$ for Sent140 RNN (where $T = 600$). We tuned c from the set $\{2^{-3}, 2^{-2}, 2^{-1}, 1\}$, while the choice of the range of γ_0 depended on the dataset-model pair. The tuning criterion we used was the mean of the loss distribution over the training clients (with client i weighted by α_i) at the end of the time horizon. That is, we chose the γ_0, c which gave the best terminal training loss.

Tuning of the regularization parameter The regularization parameter λ for linear models was tuned with cross validation from the set $\{10^{-k} : k \in \{3, \ldots, 8\}\}$. This was performed as described below.

For each dataset, we held out half the training clients as validation clients. Then, for different values of the regularization parameter, we trained a model with the (smaller subset of) training clients and evaluate its performance on the validation clients. We selected the value of the regularization parameter as the one which gave the smallest 90th percentile of the misclassification error on the validation clients.

Baselines parameters We tune the proximal parameter of FedProx with cross validation. The procedure we followed is identical to the procedure we described above for the regularization parameter λ. The set of parameters tested is $\{10^{-j}, j \in \{0, \ldots, 3\}\}$.

We did not attempt to tune the parameter q of q-FFL and report the performance of all values of q which we tried.

Hyperparameters of Δ-FL We optimize Δ-FL via Algorithm 3 with a fixed number of local steps, corresponding to one epoch. For simplicity, we calculate the quantile exactly, assuming client losses are available to the server.

C.3: Evaluation strategy and other details

Evaluation metrics We record the loss of each training client and the misclassification error of each testing client, as measured on its local data.

The evaluation metrics noted in Sect. C.4 are the following: the weighted mean of the loss distribution over the training clients, the (unweighted) mean misclassification error over the testing clients, the weighted τ-percentile of the loss over the training client and the (unweighted) τ-percentile of the misclassification error over the testing clients for values of τ among $\{20, 50, 60, 80, 90, 95\}$. We also present the 90th and 95th superquantile of the test misclassification error (i.e., average misclassification error of the worst 10% and 5% of the clients respectively), as well as the average test misclassification error of the best 10% clients. The weight α_i used for training client i was set as proportional to the number of datapoints on the client.

Evaluation times We evaluate the model during the training process once every l communication rounds. The value of l used was $l = 50$ for EMNIST linear model, $l = 10$ for EMNIST ConvNet, $l = 20$ for Sent140 linear model and $l = 25$ for Sent140 RNN.

Hardware We run each experiment as a simulation as a single process. The linear models were trained on m5.8xlarge AWS instances, each with an Intel Xeon Platinum 8000 series processor with 128 GB of memory running at most 3.1 GHz. The neural network experiments were trained on workstation with an Intel i9 processor with 128 GB of memory at 1.2 GHz, and two Nvidia Titan Xp GPUs. The Sent140 RNN experiments were run on a CPU while the other neural network experiments were run using GPUs.

Software packages Our implementation is based on NumPy using the Python language. In the neural network experiments, we use PyTorch to implement the *LocalUpdate* procedure, i.e., the model itself and the automatic differentiation routines provided by PyTorch to make SGD updates.

Randomness Since several sampling routines appear in the procedures such as the selection of clients or the local stochastic gradient, we carry out our experiments with five different seeds and plot the average metric value over these seeds. Each simulation is run on a single process. Where appropriate, we report one standard deviation from the mean.

C.4: Experimental results

We now present the experimental results of the paper.

- We present different metrics on the distribution of test misclassification error over the clients, comparing Δ-FL to baselines.
- We study the convergence of Algorithm 3 for Δ-FL over the course of the optimization, and compare it with FedAvg.
- We plot the histograms of the distribution of losses over train clients as well as the test misclassification errors over test clients at the end of the training process.
- We present in the form of scatter plots the training loss and test misclassification error across clients achieved at the end of the training, versus the number of local data points on the client.
- We present the number of clients having a loss greater than the quantile at each communication round for Δ-FL. This gives the effective number of clients selected in each round, cf. Proposition 3 and Remark 1.

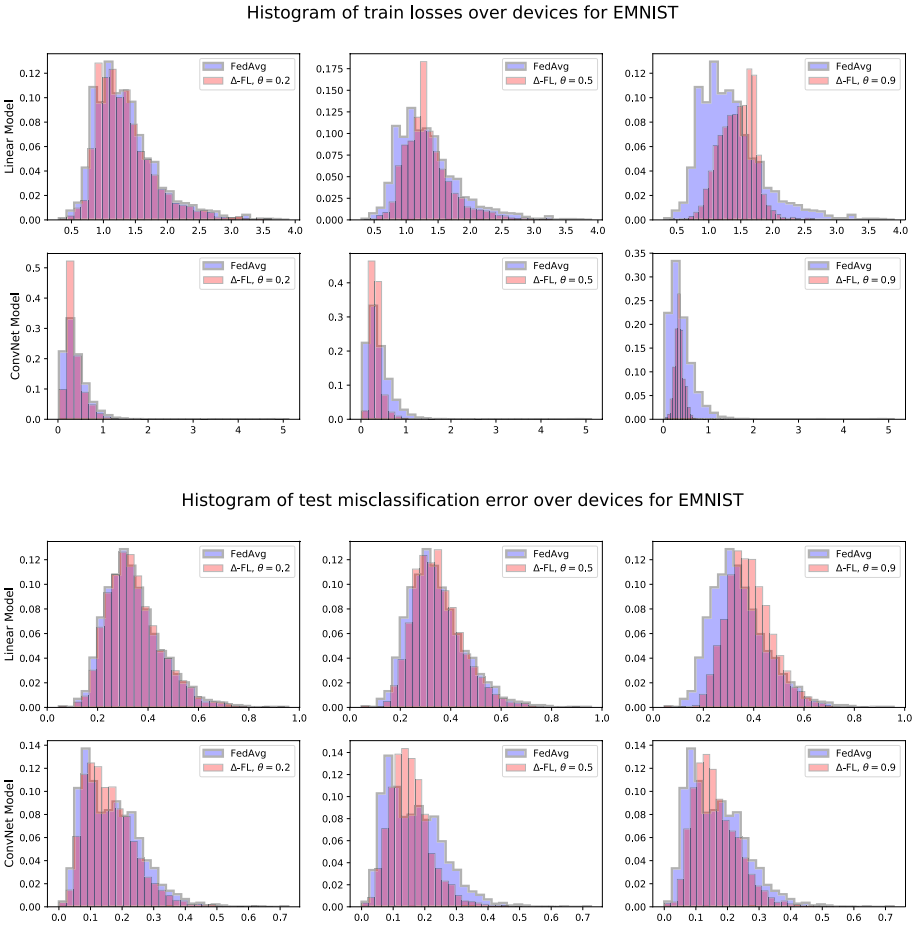

Fig. 7 Histogram of loss distribution over training clients and misclassification error distribution over testing clients for EMNIST. The identification of the model (linear or ConvNet) is given on the *y*-axis of the histograms

Comparison to baselines We now present a detailed comparison of various statistics of the test misclassification error distribution for different methods in Table 6-For each column, the smallest mean over five random runs is highlighted in bold. Further, if no other method is within one standard deviation of this method, the entire entry (i.e., mean ± std) is highlighted in bold.

Histograms of loss and test misc. Error over Devices Here, we plot the histograms of the loss distribution over training clients and the misclassification error distribution over testing clients. We report the losses and errors obtained at the end of the training process. Each metric is averaged per client over 5 runs of the random seed. Figure 7 shows the histograms for EMNIST, while Fig. 8 shows the histograms for Sent140 dataset. for Sent140. We note that Δ-FL tends to exhibit thinner upper tails at multiple values of θ and a lower variance of the distribution in most of the cases. This is also confirmed by the figures in Tables 6 , 7, 8 and 9. This shows the benefit of using Δ-FL over vanilla FedAvg.

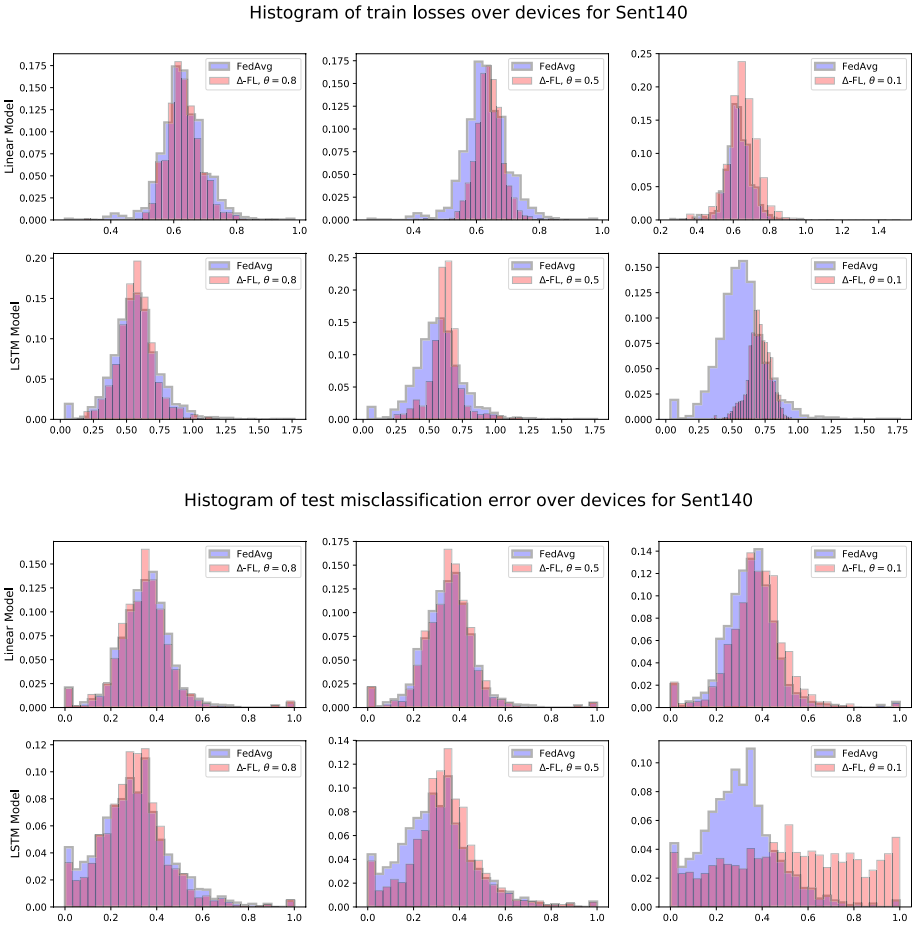

Fig. 8 Histogram of loss distribution over training clients and misclassification error distribution over testing clients for Sent140. The identification of the model (linear or RNN) is given on the *y*-axis of the histograms

Performance compared to local data size Next, we plot the loss on training clients versus the amount of local data on the client and the misclassification error on the test clients versus the amount of local data on the client. See Fig. 9 for EMNIST and Fig. 10 for Sent140.

Observe firstly that improvement over the worst cases is achieved regardless of the local data size of the clients. Indeed, the client re-weighting step operates a sorting of the loss of the clients which does not prevent small clients from being selected. In contrast, FedAvg, by averaging with respect to the weights of the clients is likely to put more weight on the clients with larger local data size. Secondly, Δ-FL appears to reduce the variance of of the loss on the train clients. Lastly, note that amongst test clients with a small number of data points (e.g., < 200 for EMNIST or < 100 for Sent140), Δ-FL reduces the variance of the misclassification error. Both effects are more pronounced on the neural network models.

Table 6 Metrics for the test misclassification error for EMNIST (Linear Model)

Method	Mean	Standard deviation	10th Percentile	Median	90th Percentile
FedAvg	34.38 ± 0.38	18.39 ± 0.33	21.54 ± 0.35	32.61 ± 0.39	49.65 ± 0.67
FedAvg $\theta = 0.8$	34.20 ± 0.45	18.25 ± 0.22	$\mathbf{21.37} \pm 0.26$	32.10 ± 0.34	49.92 ± 1.16
FedAvg $\theta = 0.5$	34.51 ± 0.47	18.21 ± 0.30	21.40 ± 0.36	32.36 ± 0.59	50.28 ± 0.77
FedAvg $\theta = 0.1$	34.60 ± 0.46	18.58 ± 0.31	21.71 ± 0.37	32.54 ± 0.37	50.33 ± 1.28
FedProx	$\mathbf{33.82} \pm 0.30$	18.25 ± 0.23	21.37 ± 0.35	$\mathbf{31.75} \pm 0.20$	49.15 ± 0.74
q-FFL (Best $q = 1.0$)	34.71 ± 0.27	19.34 ± 0.30	22.33 ± 0.41	32.80 ± 0.23	49.90 ± 0.58
Tilted-ERM (Best $t = 1.0$)	34.15 ± 0.25	10.78 ± 0.30	22.43 ± 0.29	32.36 ± 0.23	48.59 ± 0.62
AFL	39.32 ± 0.27	25.42 ± 0.27	28.64 ± 0.43	38.16 ± 0.34	51.62 ± 0.28
Δ-FL $\theta = 0.8$	34.48 ± 0.26	19.16 ± 0.32	22.24 ± 0.32	32.85 ± 0.31	49.10 ± 0.24
Δ-FL $\theta = 0.5$	35.01 ± 0.20	20.46 ± 0.34	23.64 ± 0.22	33.83 ± 0.34	$\mathbf{48.44} \pm 0.38$
Δ-FL $\theta = 0.1$	38.32 ± 0.48	23.86 ± 0.59	27.27 ± 0.64	37.52 ± 0.67	50.34 ± 0.95

The boldfaced entries indicate the smallest error in each column

Table 7 Metrics for the test misclassification error for EMNIST (ConvNet Model)

Method	Mean	Standard deviation	10th Percentile	Median	90th Percentile
FedAvg	16.63 ± 0.50	$\mathbf{4.94} \pm 0.14$	$\mathbf{6.43} \pm 0.24$	15.34 ± 0.37	28.46 ± 1.07
FedAvg $\theta = 0.8$	15.95 ± 0.42	5.25 ± 0.19	6.86 ± 0.38	14.84 ± 0.24	26.82 ± 1.28
FedAvg $\theta = 0.5$	16.22 ± 0.23	5.06 ± 0.17	6.47 ± 0.28	15.05 ± 0.25	27.56 ± 0.81
FedAvg $\theta = 0.1$	15.97 ± 0.43	5.40 ± 0.42	7.10 ± 0.64	14.76 ± 0.20	26.35 ± 2.08
FedProx	16.01 ± 0.54	5.16 ± 0.32	6.68 ± 0.44	14.88 ± 0.29	27.01 ± 1.86
q-FFL (Best $q = 0.001$)	16.58 ± 0.30	5.05 ± 0.21	6.53 ± 0.20	15.40 ± 0.43	28.02 ± 0.80
Tilted-ERM (Best $t = 1.0$)	15.69 ± 0.38	7.31 ± 0.68	7.26 ± 0.51	$\mathbf{14.66} \pm 0.16$	25.46 ± 1.49
AFL	33.00 ± 0.37	20.38 ± 0.23	22.92 ± 0.23	31.58 ± 0.27	45.07 ± 1.00
Δ-FL $\theta = 0.8$	16.08 ± 0.40	5.60 ± 0.14	7.31 ± 0.29	14.85 ± 0.48	26.23 ± 1.15
Δ-FL $\theta = 0.5$	$\mathbf{15.48} \pm 0.30$	6.13 ± 0.15	8.08 ± 0.16	14.73 ± 0.22	$\mathbf{23.69 \pm 0.94}$
Δ-FL $\theta = 0.1$	16.37 ± 1.03	6.61 ± 0.42	8.28 ± 0.65	15.49 ± 1.03	25.45 ± 2.77

The boldfaced entries indicate the smallest error in each column

Appendix D: Numerical experiments: End-to-end differential privacy

We consider a synthetic classification dataset to evaluate the privacy-utility tradeoff of Δ-FL under end-to-end differential privacy.

D.1: End-to-end differential privacy with Δ-FL

To obtain an end-to-end differentially private version of Δ-FL, we modify the weight aggregation step of Algorithm 1 (line 11). Specially, we clip the weight updates and add

Table 8 Metrics for the test misclassification error for Sent140 (Linear Model)

Method	Mean	Standard deviation	10th Percentile	Median	90th Percentile
FedAvg	34.74 ± 0.31	12.16 ± 0.15	21.89 ± 0.24	34.81 ± 0.38	46.83 ± 0.54
FedAvg $\theta = 0.8$	34.47 ± 0.03	12.08 ± 0.16	21.69 ± 0.26	34.62 ± 0.17	46.59 ± 0.38
FedAvg $\theta = 0.5$	34.46 ± 0.07	12.11 ± 0.24	$\mathbf{21.55} \pm 0.51$	$\mathbf{34.48} \pm 0.20$	47.00 ± 0.40
FedAvg $\theta = 0.1$	34.79 ± 0.32	11.97 ± 0.37	22.08 ± 0.75	34.93 ± 0.35	46.69 ± 0.84
FedProx	34.74 ± 0.31	12.16 ± 0.15	21.89 ± 0.24	34.82 ± 0.39	46.83 ± 0.54
q-FFL (Best $q = 1.0$)	34.48 ± 0.06	11.96 ± 0.14	21.61 ± 0.24	34.57 ± 0.16	$\mathbf{46.38} \pm 0.40$
Tilted-ERM (Best $t = 1.0$)	34.71 ± 0.31	12.00 ± 0.14	21.83 ± 0.34	34.91 ± 0.39	46.70 ± 0.50
AFL	35.97 ± 0.08	11.83 ± 0.09	23.58 ± 0.28	36.09 ± 0.17	47.51 ± 0.32
Δ-FL $\theta = 0.8$	$\mathbf{34.41} \pm 0.22$	12.17 ± 0.11	21.77 ± 0.34	34.64 ± 0.25	46.44 ± 0.38
Δ-FL $\theta = 0.5$	35.28 ± 0.25	$\mathbf{11.68} \pm 0.40$	23.03 ± 0.38	35.55 ± 0.53	46.64 ± 0.41
Δ-FL $\theta = 0.1$	37.78 ± 0.89	12.86 ± 0.52	23.93 ± 0.99	37.80 ± 1.30	51.38 ± 1.07

The boldfaced entries indicate the smallest error in each column

Table 9 Metrics for the test misclassification error for Sent140 (RNN Model)

Method	Mean	Standard deviation	10th Percentile	Median	90th Percentile
FedAvg	30.16 ± 0.44	4.36 ± 1.26	10.06 ± 2.06	29.51 ± 0.33	49.66 ± 3.95 1
FedAvg $\theta = 0.8$	$\mathbf{29.85} \pm 0.46$	5.39 ± 1.32	11.90 ± 2.27	29.57 ± 0.31	46.93 ± 3.84 1
FedAvg $\theta = 0.5$	31.06 ± 1.01	$\mathbf{4.33} \pm 2.73$	$\mathbf{9.69} \pm 4.89$	30.14 ± 0.71	53.10 ± 7.22 1
FedAvg $\theta = 0.1$	31.96 ± 1.47	4.82 ± 2.09	11.65 ± 4.83	31.55 ± 1.13	52.87 ± 8.41 1
FedProx	30.20 ± 0.48	4.35 ± 1.23	10.37 ± 2.08	$\mathbf{29.51} \pm 0.32$	49.85 ± 4.07
q-FFL (Best $q = 0.01$)	29.99 ± 0.56	4.90 ± 1.66	10.98 ± 2.88	29.56 ± 0.39	48.65 ± 4.68
Tilted-ERM (Best $t = 1.0$)	30.13 ± 0.49	14.17 ± 2.10	13.18 ± 3.33	29.96 ± 0.84	46.54 ± 3.27
AFL	37.74 ± 0.65	9.90 ± 1.46	18.19 ± 1.99	36.95 ± 1.03	57.78 ± 1.19
Δ-FL $\theta = 0.8$	30.30 ± 0.33	6.75 ± 2.68	13.05 ± 3.87	29.92 ± 0.38	$\mathbf{46.46} \pm 4.39$
Δ-FL $\theta = 0.5$	33.58 ± 2.44	8.74 ± 3.98	16.77 ± 6.62	33.28 ± 2.27	50.47 ± 8.24
Δ-FL $\theta = 0.1$	51.97 ± 11.81	9.11 ± 5.47	16.67 ± 9.15	52.44 ± 13.21	86.44 ± 10.95

The boldfaced entries indicate the smallest error in each column

Gaussian noise to obtain differential privacy via the Gaussian mechanism. The overall algorithm is given in Algorithm 4.

Privacy accounting We now discuss the privacy spent in each communication round. For simplicity, we assume the number $m^{(t)} = \sum_{i \in S} \mathbb{I}(F_i(w^{(t)}) \geq Q^{(t)})$ of selected clients is publicly known.

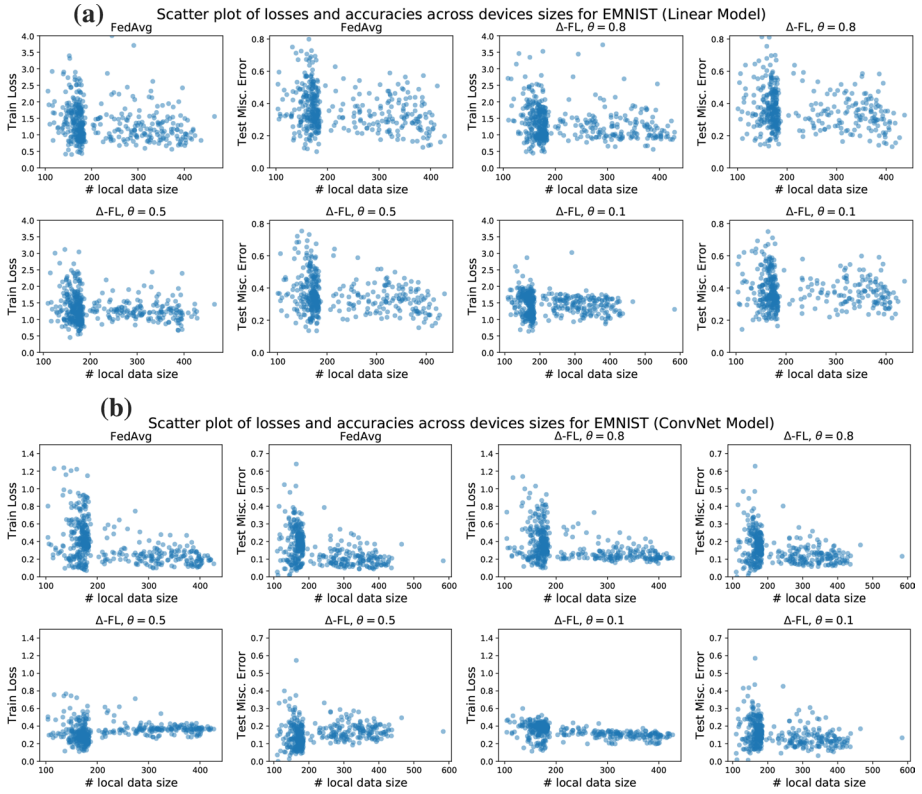

Fig. 9 Scatter plot of **a** loss on training client versus amount of local data, and **b** misclassification error on testing client vs. amount of local data for EMNIST

Claim 20 *Consider the setting of Algorithm* 4 *with noise scale* σ_w*, norm bound* C *and Algorithm* 2 *with* b *bins and noise scale* $\sigma = \sigma_q$*. Each round of Algorithm* 4 *satisfies* $(1/2)\varepsilon^2$ *-concentrated DP where*

$$\frac{1}{2}\varepsilon^2 = \frac{1}{2}\min\left\{\frac{c^2\log_2^2 b}{m\sigma_q^2} + \psi b\,,\left(\frac{c\log_2 b}{\sqrt{m}\sigma_q} + \psi\sqrt{2b}\right)^2\right\} + \frac{\sigma_w^2}{2C^2}\,,$$

where $\psi = 10\sum_{i=1}^{m-1}\exp\left(-2\pi^2\sigma_q^2 i/(i+1)\right)$.

Proof The privacy bound of the quantile computation from Algorithm 2 is given by Theorem. Since the contribution $\delta_i^{(t)}$ of each client has ℓ_2 norm $\left\|\delta_i^{(t)}\right\| \leq C$ and we add

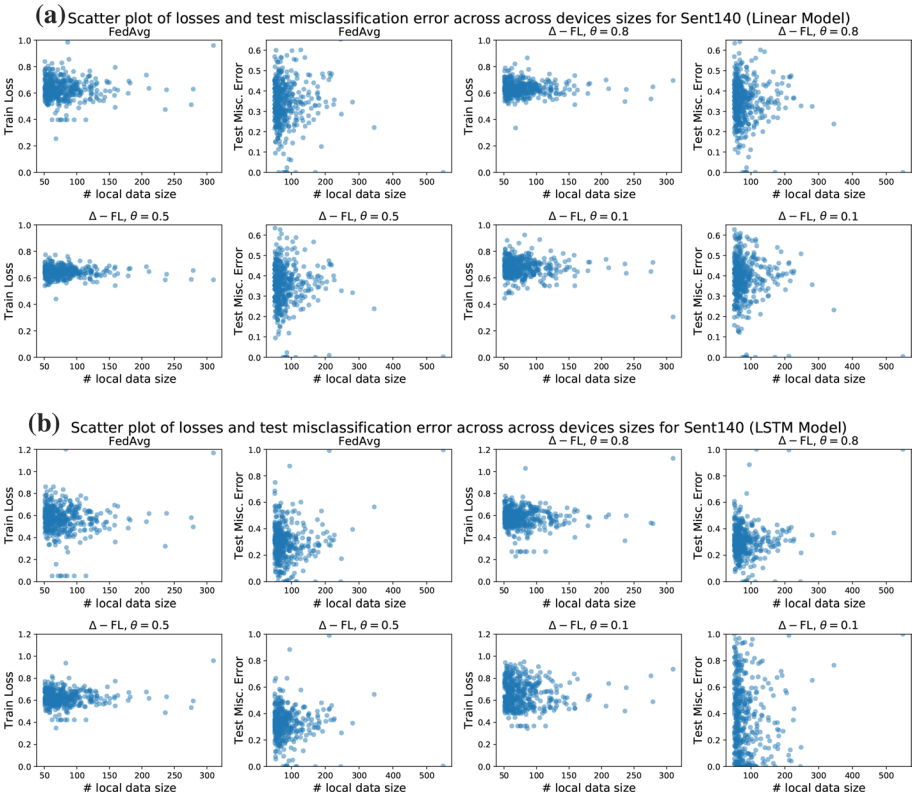

Fig. 10 Scatter plot of **a** loss on training client versus amount of local data, and **b** misclassification error on testing client vs. amount of local data for Sent140

Gaussian noise $\mathcal{N}(0, \sigma_w^2 I_d)$, the weight update step satisfies $\sigma_w^2/(2C^2)$-concentrated DP. The proof is completed by noting that concentrated differential privacy composes additively. □

To obtain a bound on the concentrated DP of the entire algorithm, we rely on generic upper bounds of Zhu and Wang (2019) for privacy amplification by subsampling.

Algorithm 4 The Δ-FL Algorithm with End-to-End Differential Privacy

Input: Initial iterate $w^{(0)}$, number of communication rounds T, number of clients per round m, number of local updates τ, local step size γ, ℓ_2 norm bound C for weight updates, noise variance σ_w^2

for $t = 0, 1, \ldots, T-1$ **do**
  Sample m clients from $[n]$ without replacement in S
  Estimate the $(1-\theta)$-quantile of $F_i(w^{(t)})$ for $i \in S$ with distributed differential privacy (Algorithm 2); call this $Q^{(t)}$
  Set $m^{(t)} = \sum_{i \in S} \mathbb{I}\left(F_i(w^{(t)}) \geq Q^{(t)}\right)$
  for each selected client $i \in S$ in parallel **do**
    Initialize $w_{k,0}^{(t)} = w^{(t)}$
    for $k = 0, \ldots, \tau - 1$ **do**
      $w_{i,k+1}^{(t)} = (1 - \gamma\lambda) w_{i,k}^{(t)} - \gamma \nabla F_i(w_{i,k}^{(t)})$
    end for
    Define the norm-clipped update contributed by the client

$$\delta_i^{(t)} = \begin{cases} \frac{C\left(w_{i,\tau}^{(t)} - w^{(t)}\right)}{\max\left\{C, \left\|w_{i,\tau}^{(t)} - w^{(t)}\right\|_2\right\}}, & \text{if } F_i(w^{(t)}) \geq Q^{(t)} \\ \mathbf{0}_d, & \text{else} \end{cases}$$

  end for
  Sample Gaussian noise $\xi^{(t)} \sim \mathcal{N}(0, \sigma_w^2 I_d)$ and update

$$w^{(t+1)} = w^{(t)} + \frac{1}{m^{(t)}} \sum_{i \in S} \delta_i^{(t)} + \xi^{(t)}$$

end for
return w_T

D.2: Experimental setup

We consider a synthetic classification dataset and train a linear model on it.

Dataset description We create a 10-class classification dataset in $d = 20$ dimensions, inspired by Guyon (2003). The input x for each class k is drawn from a Gaussian of mean μ_i and identity covariance in $\mathbb{R}^{15}$. The means μ_i's are the corners of a random polytope in $\mathbb{R}^{15}$. We add 2 features that are linear combinations of the 15 informative ones and 3 features that are pure noise. Overall, the dataset can be generated using the `make_classification` function of scikit-learn (Pedregosa et al., 2011) as

```
x, y = make_classification(
        n_samples=int(5e5), n_features=20,
        n_informative=15, n_redundant=2, n_repeated=0,
        n_classes=10, n_clusters_per_class=1,
        class_sep=5.0, hypercube=False, random_state=2345
)
```

We now split this dataset into a federated dataset with $n = 2500$ training clients and $n' = 500$ validation and $n'' = 500$ test clients. The data distribution $q_i(x, y) = q_i(y)q_i(x|y)$ across the clients is designed to exhibit a label shift, i.e., the distribution $q_i(y)$ over labels for each client is different while the class-conditional distribution $q_i(x|y = k) = \mathcal{N}(\mu_k, I_d)$ is the same across clients. The class distribution $q_i(y)$ on each training client i is drawn from a Dirichlet distribution Dir(0.5), while that for a validation or test client is drawn from Dir(0.01). We sample 100 input-output pairs for each training, validation, and test client.

Model and per-client objective We use a linear model (with intercept) on each client and the multinomial logistic loss, also known as the cross entropy loss, to define the per-client objective.

Algorithms and hyperparameters We compare Algorithm 4 with DP-FedAvg (McMahan et al., 2018), a version of FedAvg with differential privacy.

Both algorithms used a single full gradient step per client with a fixed learning rate of 0.1. For each algorithm, we sample 100 clients per round and run the training for a total of 1000 rounds. We vary the privacy budget $\varepsilon \in \{3, 5, 10, 15, 20\}$ and tune the following hyperparameters for each algorithm.

For DP-FedAvg, we tune the ℓ_2 norm bound (analogous to C in Algorithm 4) and set the noise scale σ_w depending on the target privacy budget ε and the norm bound C. For Algorithm 4, we allocate r-times the privacy budget of the weight updates to the quantile updates. In addition, we also tune:

- The loss upper bound B, so that all losses are truncated to $[0, B]$,
- The number of bins b in the hierarchical histogram,
- The ℓ_2 norm bound C for the weight update.

We tune all 4 hyperparameters with a grid search and set the noise scale σ_w for the weight update, and σ_q/c for the quantile update depending on the selected hyperparameters and the privacy budget ε. The objective of the grid search was to minimize the 90th percentile of the misclassification errors across all validation clients.

The ranges of the hyperparameters considered are quantile privacy ratio $r \in \{0.1, 0.25, 0.5, 0.75\}$, loss upper bound $B \in \{0.7, 0.9, 1.1, 1.3, 1.5\}$,[4], number of bins $b \in \{16, 32, 64\}$, and update norm $C \in \{0.9, 1.1, 1.3, 1.5\}$.[5]

Author contributions KP and YL jointly and equally contributed to the theoretical analyses, designing, and running experiments. ZH and JM conceived the original idea and supervised the project. All authors discussed the theoretical and experimental results and contributed to the final manuscript.

[4] The loss at convergence was around 0.7, while that at random guessing is $\log 10 \approx 2.3$.

[5] These correspond approximately to the 0.3, 0.5, 0.7, 0.9 quantiles of the update norms of FedAvg without differential privacy, during the latter half of training.

Funding We acknowledge support from NSF DMS 2023166, DMS 1839371, CCF 2019844, the CIFAR program "Learning in Machines and Brains", faculty research awards, and a JP Morgan Ph.D. fellowship. This work has been partially supported by MIAI – Grenoble Alpes, (ANR-19-P3IA-0003). The work was mainly performed while Krishna Pillutla was at the University of Washington, and Yassine Laguel was at the Université Grenoble Alpes.

Data availibility The data we use is publicly available and can be downloaded from the LEAF benchmark, cf. Caldas et al. (2018).

Code availability The code and scripts to reproduce our experiments can be found at https://github.com/krishnap25/simplicial-fl.

Declarations

Conflict of interest The authors have no further financial or non-financial interests to disclose.

References

Agarwal, N., Kairouz, P., & Liu, Z. (2021). The skellam mechanism for differentially private federated learning. In: NeurIPS.

Artzner, P., Delbaen, F., Eber, J.-M., & Heath, D. (1999). Coherent measures of risk. *Mathematical Finance, 9*(3), 203–228.

Beck, A., & Teboulle, M. (2012). Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization, 22*(2), 557–580.

Ben-Tal, A., den Hertog, D., Waegenaere, A. D., Melenberg, B., & Rennen, G. (2013). Robust solutions of optimization problems affected by uncertain probabilities. *Management Science, 59*(2), 341–357.

Ben-Tal, A., & Teboulle, M. (2007). An old-new concept of convex risk measures: The optimized certainty equivalent. *Mathematical Finance, 17*(3), 449–476.

Bertsekas, D.P. (1999) Nonlinear Programming.

Bietti, A., Wei, C., Dudík, M., Langford, J., & Wu, Z.S. (2022). Personalization improves privacy-accuracy tradeoffs in federated learning. In: ICML, vol. 162, pp. 1945–1962.

Bonawitz, K.A., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečný, J., Mazzocchi, S., McMahan, B., Overveldt, T.V., Petrou, D., Ramage, D., Roselander, & J. (2019). Towards federated learning at scale: system design. In: Proceedings of machine learning and systems 2019, MLSys 2019.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In: ACM SIGSAC conference on computer and communications security, pp. 1175–1191

Bottou, L., Curtis, F. E., & Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review, 60*(2), 223–311.

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., & Shi, W. (2018). Federated learning of predictive models from federated electronic health records. *International Journal of Medical Informatics, 112*, 59–67.

Bun, M., & Steinke, T. (2016). Concentrated differential privacy: simplifications, extensions, and lower bounds. In: Hirt, M., Smith, A.D. (eds.) Theory of cryptography conference, vol. 9985, pp. 635–658.

Caldas, S., Duddu, S.M.K., Wu, P., Li, T., Konečný, J., McMahan, H.B., Smith, V., & Talwalkar, A. (2018). LEAF: A benchmark for federated settings. arXiv Preprint.

Canonne, C.L., Kamath, G., & Steinke, T. (2020). The discrete gaussian for differential privacy. In: Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual.

Cassel, A., Mannor, S., & Zeevi, A. (2018). A general approach to multi-armed bandits under risk criteria. In: Conference On learning theory. Proceedings of machine learning research, vol. 75, pp. 1295–1306.

Chan, T.-H., Shi, E., & Song, D. (2011). Private and continual release of statistics. *ACM Transactions on Information and System Security, 14*(3), 26–12624.

Chow, Y., Ghavamzadeh, M., Janson, L., & Pavone, M. (2017). Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research, 18*, 167–116751.

Chow, Y., Tamar, A., Mannor, S., & Pavone, M. (2015). Risk-sensitive and robust decision-making: A CVaR optimization approach. *Advances in Neural Information Processing Systems, 28*, 1522–1530.
Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: Extending MNIST to handwritten letters. In: International joint conference on neural networks, pp. 2921–2926.
Collins, L., Hassani, H., Mokhtari, A., & Shakkottai, S. (2021). Exploiting shared representations for personalized federated learning. In: International conference on machine learning, vol. 139, pp. 2089–2099.
Cormode, G., Kulkarni, T., & Srivastava, D. (2019). Answering range queries under local differential privacy. *VLDB, 12*(10), 1126–1138.
Curi, S., Levy, K. Y., Jegelka, S., & Krause, A. (2020). Adaptive sampling for stochastic risk-averse learning. *Advances in Neural Information Processing Systems., 33*(2020), 1036–1047.
Davis, D., & Drusvyatskiy, D. (2019). Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization, 29*(1), 207–239.
Deng, Y., Kamani, M. M., & Mahdavi, M. (2020). Distributionally robust federated averaging. *Advances in Neural Information Processing Systems., 33*, 15111–15122.
Devolder, O., Glineur, F., & Nesterov, Y. E. (2014). First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming, 146*(1–2), 37–75.
Dieuleveut, A., & Patel, K.K. (2019). Communication trade-offs for local-sgd with large step size. In: Advances in neural information processing systems, pp. 13579–13590.
Dinh, C.T., Tran, N., & Nguyen, J. (2020). Personalized Federated Learning with Moreau Envelopes. In: Proc. of NeurIPS, vol. 33, pp. 21394–21405.
Drusvyatskiy, D., & Paquette, C. (2019). Efficiency of minimizing compositions of convex functions and smooth maps. *Mathematical Programming, 178*(1), 503–558.
Duchi, J. C., & Namkoong, H. (2019). Variance-based regularization with convex objectives. *Journal of Machine Learning Research, 20*(68), 1–55.
Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., & Naor, M. (2006b). Our data, ourselves: Privacy via distributed noise generation. In: EUROCRYPT. Lecture notes in computer science, vol. 4004, pp. 486–503.
Dwork, C., McSherry, F., Nissim, K., & Smith, A.D. (2006a). Calibrating noise to sensitivity in private data analysis. In: Theory of cryptography conference. lecture notes in computer science, vol. 3876, pp. 265–284.
Dwork, C., Naor, M., Pitassi, T., & Rothblum, G.N. (2010). Differential privacy under continual observation. In: STOC, pp. 715–724.
Dwork, C., McSherry, F., Nissim, K., & Smith, A. D. (2016). Calibrating noise to sensitivity in private data analysis. *Journal of Privacy and Confidentiality, 7*(3), 17–51.
Eichner, H., Koren, T., McMahan, B., Srebro, N., & Talwar, K. (2019). *Semi-cyclic stochastic gradient descent*. In: ICML, *97*, 1764–1773.
Evans, D., Kolesnikov, V., Rosulek, M., et al. (2018). A pragmatic introduction to secure multi-party computation. *Foundations and Trends in Privacy and Security, 2*(2–3), 70–246.
Föllmer, H., & Schied, A. (2002). Convex measures of risk and trading constraints. *Finance Stochastics, 6*, 429–447.
Föllmer, H., & Schied, A. (2016). Stochastic finance: An introduction in discrete time. *The Mathematical Intelligencer*. https://doi.org/10.1515/9783110463453
Gafni, T., Shlezinger, N., Cohen, K., Eldar, Y. C., & Poor, H. V. (2022). Federated learning: A signal processing perspective. *IEEE Signal Processing Magazine, 39*(3), 14–41. https://doi.org/10.1109/MSP.2021.3125282
Go, A., Bhayani, R., & Huang, L. (2009). Twitter Sentiment Classification using Distant Supervision. CS224N Project Report, Stanford.
Guyon, I. (2003). Design of experiments of the neurips 2003 variable selection benchmark. In: NeurIPS 2003 Workshop on Feature Extraction and Feature Selection, vol. 253, p. 40.
Haddadpour, F., Kamani, M. M., Mahdavi, M., & Cadambe, V. (2019). Local SGD with periodic averaging: Tighter analysis and adaptive synchronization. *Advances in Neural Information Processing Systems, 32*, 11080–11092.
Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., & Ramage, D. (2018). Federated learning for mobile keyboard prediction. arXiv Preprint.
Hay, M., Rastogi, V., Miklau, G., & Suciu, D. (2010). Boosting the accuracy of differentially private histograms through consistency. *VLDB, 3*(1), 1021–1032.
Hiriart-Urruty, J.-B., & Lemaréchal, C. (1996). *Convex analysis and minimization algorithms I: Fundamentals*. Springer.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.
Huang, L., Shea, A. L., Qian, H., Masurkar, A., Deng, H., & Liu, D. (2019). Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *Journal of Biomedical Informatics, 99*, 103291.
Jain, P., Rush, J., Smith, A.D., Song, S., & Thakurta, A.G. (2021). Differentially private model personalization. In: NeurIPS, pp. 29723–29735.
Jhunjhunwala, D., Gadhikar, A., Joshi, G., & Eldar, Y.C. (2021). Adaptive quantization of model updates for communication-efficient federated learning. In: IEEE International conference on acoustics, speech and signal processing, pp. 3110–3114.
Kairouz, P., Liu, Z., & Steinke, T. (2021). The distributed discrete gaussian mechanism for federated learning with secure aggregation. In: ICML, vol. 139, pp. 5201–5212.
Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K. A., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., … Zhao, S. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning, 14*(1–2), 1–210.
Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S.J., Stich, S.U., & Suresh, A.T. (2020). SCAFFOLD: stochastic controlled averaging for federated learning. In: International conference on machine learning. In: Proceedings of machine learning research, vol. 119, pp. 5132–5143.
Khaled, A., Mishchenko, K., & Richtárik, P. (2020). Tighter theory for local SGD on identical and heterogeneous data. In: International conference on artificial intelligence and statistics.
Koloskova, A., Loizou, N., Boreiri, S., Jaggi, M., & Stich, S. (2020). A unified theory of decentralized sgd with changing topology and local updates. In: ICML.
Kubiak, W. (2008). Proportional optimization and fairness. In: International series in operations research & management science.
Kuhn, D., Esfahani, P.M., Nguyen, V.A., & Shafieezadeh-Abadeh, S. (2019). Wasserstein distributionally robust optimization: theory and applications in machine learning. In: Operations research & management science in the age of analytics, pp. 130–166.
Laguel, Y., Malick, J., & Harchaoui, Z. (2020). First-order optimization for superquantile-based supervised learning. In: IEEE international workshop on machine learning for signal processing, pp. 1–6.
Laguel, Y., Pillutla, K., Malick, J., & Harchaoui, Z. (2020). Device heterogeneity in federated learning: A superquantile approach. arXiv preprint.
Laguel, Y., Pillutla, K., Malick, J., & Harchaoui, Z. (2021). A Superquantile approach to federated learning with heterogeneous devices. In: IEEE CISS.
Lee, J., & Raginsky, M. (2018). Minimax statistical learning with Wasserstein distances. In: Advances in neural information processing systems, pp. 2687–2696.
Levy, D., Carmon, Y., Duchi, J.C., & Sidford, A. (2020). Large-scale methods for distributionally robust optimization. In: Advances in neural information processing systems.
Levy, D., Carmon, Y., Duchi, J. C., & Sidford, A. (2020). Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems, 33*, 8847–8860.
Li, T., Beirami, A., Sanjabi, M., & Smith, V. (2021). Tilted empirical risk minimization. In: International conference on learning representations.
Li, X., Huang, K., Yang, W., Wang, S., & Zhang, Z. (2020). On the convergence of FedAvg on Non-IID data. In: ICLR
Li, T., Sanjabi, M., & Smith, V. (2020). Fair resource allocation in federated learning. In: International conference on learning representations.
Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine, 37*(3), 50–60.
Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2020). Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems, 2*, 429–450.
McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B.A. (2017). Communication-efficient learning of deep networks from decentralized data. In: AISTATS, pp. 1273–1282.
McMahan, H.B., Ramage, D., Talwar, K., & Zhang, L. (2018). Learning differentially private recurrent language models. In: ICLR.
Mills, J., Hu, J., & Min, G. (2020). Communication-efficient federated learning for wireless edge intelligence in IoT. *IEEE Internet Things Journal, 7*(7), 5986–5994.
Mohammadi Amiri, M., & Gündüz, D. (2020). Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air. *IEEE Transactions on Signal Processing, 68*, 2155–2169.
Mohri, M., Sivek, G., & Suresh, A.T. (2019). Agnostic federated learning. In: ICML.
Nesterov, Y. (2005). Smooth minimization of non-smooth functions. *Mathematical Programming, 103*(1), 127–152.

Nurminskii, E. (1973). The quasigradient method for the solving of the nonlinear programming problems. *Cybernetics, 9*(1), 145–150.

Pantelidou, A., & Ephremides, A. (2011). Scheduling in wireless networks. *Foundations and Trends in Networking, 4*(4), 421–511.

Paulik, M., Seigel, M., Mason, H., Telaar, D., Kluivers, J., van Dalen, R.C., Lau, C.W., Carlson, L., Granqvist, F., Vandevelde, C., Agarwal, S., Freudiger, J., Byde, A., Bhowmick, A., Kapoor, G., Beaumont, S., Cahill, Á., Hughes, D., Javidbakht, O., Dong, F., Rishi, R., & Hung, S. (2021). Federated evaluation and tuning for on-device personalization: system design & applications. arXiv Preprint.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

Pennington, J., Socher, R., & Manning, C.D. (2014). GloVe: Global vectors for word representation. In: Empirical methods in natural language processing, pp. 1532–1543.

Pillutla, K., Malik, K., Mohamed, A., Rabbat, M., Sanjabi, M., & Xiao, L. (2022). Federated learning with partial model personalization. In: Proccedings of of ICML, vol. 162, pp. 17716–17758.

Ramaswamy, S., Thakkar, O., Mathews, R., Andrew, G., McMahan, H.B., & Beaufays, F. (2020). Training production language models without memorizing user data. arXiv Preprint.

Reddi, S.J., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečný, J., Kumar, S., & McMahan, H.B. (2021). Adaptive federated optimization. In: International conference on learning representations.

Reisizadeh, A., Farnia, F., Pedarsani, R., & Jadbabaie, A. (2020). Robust federated learning: The case of affine distribution shifts. *Advances in Neural Information Processing Systems, 33*, 21554–21565.

Rezaei, A., Liu, A., Memarrast, O., & Ziebart, B.D. (2021). Robust fairness under covariate shift. In: AAAI conference on artificial intelligence, pp. 9419–9427.

Rockafellar, R.T., & Wets, R.J.-B. (2009). Variational analysis vol. 317.

Rockafellar, R. T., & Uryasev, S. (2000). Optimization of conditional value-at-risk. *Journal of Risk, 2*, 21–42.

Rockafellar, R. T., & Uryasev, S. (2002). Conditional value-at-risk for general loss distributions. *Journal of Banking & Finance, 26*(7), 1443–1471.

Rockafellar, R. T., & Uryasev, S. (2013). The fundamental risk quadrangle in risk management, optimization and statistical estimation. *Surveys in Operations Research and Management Science, 18*(1–2), 33–53.

Rockafellar, R. T., Uryasev, S., & Zabarankin, M. (2008). Risk tuning with generalized linear regression. *Mathematics of Operations Research, 33*(3), 712–729.

Sani, A., Lazaric, A., & Munos, R. (2012). Risk-aversion in multi-armed bandits. *Advances in Neural Information Processing Systems, 25*, 3284–3292.

Sattler, F., Müller, K.-R., & Samek, W. (2020). Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems, 32*, 3710–3722.

Sery, T., Shlezinger, N., Cohen, K., & Eldar, Y. C. (2021). Over-the-air federated learning from heterogeneous data. *IEEE Transactions on Signal Processing, 69*, 3796–3811.

Shlezinger, N., Chen, M., Eldar, Y. C., Poor, H. V., & Cui, S. (2021). UVeQFed: Universal vector quantization for federated learning. *IEEE Transactions on Signal Processing, 69*, 500–514.

Smith, A.D. (2011). Privacy-preserving statistical estimation with optimal convergence rates. In: STOC, pp. 813–822.

Smith, A.D., Thakurta, A., & Upadhyay, J. (2017). Is interaction necessary for distributed private learning? In: IEEE symposium on security and privacy, pp. 58–77.

Stanczak, S., Wiczanowski, M., & Boche, H. (2009). Fundamentals of resource allocation in wireless networks: Theory and algorithms. *Communications and networking: foundations in signal processing*. Springer.

Stich, S.U. (2019). Local SGD converges fast and communicates little. In: International conference on learning representations.

Tamar, A., Chow, Y., Ghavamzadeh, M., & Mannor, S. (2015). Policy gradient for coherent risk measures. *Advances in Neural Information Processing Systems, 28*, 1468–1476.

Wang, J., Charles, Z., Xu, Z., Joshi, G., McMahan, H.B., Al-Shedivat, M., Andrew, G., Avestimehr, S., Daly, K., & Data, D., et al. (2021). A Field Guide to Federated Optimization. arXiv Preprint.

Wang, J., Liu, Q., Liang, H., Joshi, G., & Poor, H. V. (2020). Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in Neural Information Processing Systems., 33*, 7611–7623.

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

Wei, K., Li, J., Ding, M., Ma, C., Yang, H. H., Farokhi, F., Jin, S., Quek, T. Q. S., & Poor, H. V. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security, 15*, 3454–3469.

Williamson, R.C., & Menon, A.K. (2019). Fairness risk measures. In: International conference on machine learning.

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., & Beaufays, F. (2018). Applied federated learning: improving google keyboard query suggestions. arXiv Preprint.

Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., & Khazaeni, Y. (2019). Bayesian nonparametric federated learning of neural networks. In: International conference on machine learning, pp. 7252–7261.

Zhou, F., & Cong, G. (2018). On the convergence properties of a K-step averaging stochastic gradient descent algorithm for nonconvex optimization. In: International joint conference on artifical intelligence, pp. 3219–3227.

Zhu, Y., & Wang, Y. (2019). Poission Subsampled Rényi Differential Privacy. In: Proceedings of ICML, vol. 97, pp. 7634–7642.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.