Revisiting the Moment Accountant Method for DP-SGD

Marten van Dijk[1,2,3*], **Nhuong V. Nguyen**[4†*], **Toan N. Nguyen**[4†],
Lam M. Nguyen[5], **Phuong Ha Nguyen**[6]
[1] CWI Amsterdam, The Netherlands
[2] Department of Computer Science, Vrije Universiteit Amsterdam, The Netherlands
[3] Department of Electrical and Computer Engineering, University of Connecticut, CT, USA
[4] Department of Computer Science and Engineering, University of Connecticut, CT, USA
[5] IBM Research, Thomas J. Watson Research Center, Yorktown Heights, NY, USA
[6] eBay, CA, USA

marten.van.dijk@cwi.nl, nhuong.nguyen@uconn.edu, toan.nguyen@uconn.edu,
LamNguyen.MLTD@ibm.com, phuongha.ntu@gmail.com

Abstract

In order to provide differential privacy, Gaussian noise with standard deviation σ is added to local SGD updates after performing a clipping operation in Differential Private SGD (DP-SGD). By non-trivially improving the account method we prove a simple and easy to evaluate closed form (ϵ, δ)-DP guarantee: DP-SGD is (ϵ, δ)-DP if $\sigma = \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}$ and T is at least $\approx 2k^2/\epsilon$, where T is the total number of rounds, and $K = kN$ is the total number of gradient computations where k measures K in number of epochs of size N of the local data set. We prove that our expression is close to tight in that if T is more than a constant factor ≈ 8 smaller than the lower bound $\approx 2k^2/\epsilon$, then the (ϵ, δ)-DP guarantee is violated. Choosing the smallest possible value $T \approx 2k^2/\epsilon$ not only leads to a close to tight DP guarantee, but also minimizes the total number of communicated updates and this means that the least amount of noise is aggregated into the global model and accuracy is optimized as confirmed by simulations. In addition this minimizes round communication.

1 Introduction

Privacy leakage is a big problem in the big-data era. Solving a learning task based on big data intrinsically means that only through a collaborative effort sufficient data is available for training a global model with sufficient clean accuracy (utility). Federated learning is a framework where a learning task is solved by a loose federation of participating devices/clients which are coordinated by a central server [21]. Clients, who use own local data to participate in a learning task by training a global model, want privacy guarantees for their local proprietary data.

For this reason DP-SGD [1] was introduced as it adapts distributed Stochastic Gradient Descent (SGD) with Differential Privacy (DP). Here, differential privacy comes from adding Gaussian noise $\mathcal{N}(0, C^2\sigma^2\mathbf{I})$ to local (client-computed) mini-batch SGD updates after performing a clipping operation $x \to [x]_C = x/\max\{1, \|x\|/C\}$. This wrings with being able to learn an accurate global model; even though more Gaussian noise by increasing σ leads to better differential privacy guarantees, this hurts the accuracy of the final global model and slows convergence of SGD toward the final

[*] These authors contributed equally.
[†] Supported by NSF grant CNS-1413996 "MACS: A Modular Approach to Cloud Security."
The previous version of this eprint introduces proactive DP-SGD together with an analysis and discussion.

In memory of our dear friend and author of this paper Nhuong Nguyen.

global model. We can also not make the clipping constant C too small, otherwise, convergence of SGD to a minimum (a final global model) does not bootstrap.

In order to understand how to set parameters that balance accuracy and differential privacy, (ϵ, δ)-DP guarantees for DP-SGD have originally been analysed by the moment account method [1]. This method leads to a simple closed form bound relating various DP-SGD parameters and as a result gives a first intuitive insight of how to chose a concrete parameter setting. The disadvantage of the bound is its dependence on two constants c_1 and c_2 which are proven to exist but are themselves not further characterized. It turns out that many pairs (c_1, c_2) are possible and we show in this paper that these constants can be related through a function and this allows us to non-trivially improve the moment accountant method and show a bound without unknown constants.

We know, however, that the recent f-DP framework [6] already provides a *tight* differential private characterization of DP-SGD in terms of a so-called trade-off function. The f-DP framework shows how (ϵ, δ)-DP can be equivalently formulated as $f_{\epsilon,\delta}$-DP in the f-DP framework. We notice that a trade-off function can also be used to derive other divergence based DP guarantees, but not the other way around (as opposed to (ϵ, δ)-DP). The f-DP framework supersedes all existing other frameworks in that a trade-off function contains all the information needed to derive known DP metrics.

Given that f-DP produces tight DP characterizations, why are we concerned with improving the moment accountant method? The reason is that the *general* f-DP framework, cited from [6], has "the disadvantage is that the expressions it yields are more unwieldy: they are computer evaluable, so usable in implementations, but do not admit simple closed form." For the *specific* case of DP-SGD, we want to improve the moment account method in order to obtain a simple closed form (ϵ, δ)-DP guarantee which we want to show is tight "up to a constant" by using the f-DP framework. This closes the gap between the moment accountant method and f-DP and in essence marries the advantages of both: It allows us to gain immediate intuition and to a-priori set concrete parameters for DP-SGD without extensive simulations using the f-DP based differential privacy accountant of [38].

We want a simple, close to being tight, closed form expression of the differential privacy guarantee as a function/expression of parameters representing the local data set size N, the mini-batch or sample size s during local SGD iterations, the total number of rounds T, and the added Gaussian noise per communicated SGD update represented by standard deviation σ.

- We non-trivially improve the analysis of the moment accountant method in [1] and show for the first time that (ϵ, δ)-differential privacy can be achieved for
$$\sigma = \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon} \tag{1}$$
in parameter settings with (a) a reasonable DP guarantee by choosing $\delta \leq 1/N$ and ϵ smaller than 0.5, where (b) the total number K of gradient computations over all local rounds performed on the local data set is at least a constant ($\approx e/2$) times $\sqrt{\epsilon \ln(1/\delta)}$ epochs (of size N), and (c) T is at most another constant ($\approx 1/2$) times $\epsilon(N/s)^2$.
We notice that in a practical setting conditions (a) and (b) are typically satisfied (since we want small enough ϵ and δ for reasonable DP guarantees and the lower bound on K is generally satisfied in practice as usually K is 50 or 100s of epochs). Therefore, condition (c) is the limiting constraint.
- We equivalently reformulate condition (c) as a lower bound on T: T is at least $\approx 2k^2/\epsilon$ where $k = K/N$ measures K in number of epochs (of size N). Confirmed by simulations, we show that by setting T equal to the lower bound $T \approx 2k^2/\epsilon$, we optimize accuracy (as this minimizes the number of times/rounds when noise is aggregated into the global model at the server) and minimize round complexity. By using the f-DP framework, we prove that T's required lower bound $\approx 2k^2/\epsilon$ cannot be made smaller by more than a constant factor ≈ 8 (otherwise, this conflicts with an asymptotical result proved by the f-DP framework). We conclude that setting
$$T \approx 2k^2/\epsilon \tag{2}$$
leads to a close to tight (ϵ, δ)-DP guarantee which also optimizes accuracy and minimizes round complexity.
- Simulations based on (1) and (2) show a significantly smaller ϵ compared to current state of the art. For example, our theory applies to $\epsilon = 0.15$ for the non-convex problem of the simple neural network LeNet [18] with cross entropy loss function for image classification of MNIST [17] at a test accuracy of 93%, compared to 98% without differential privacy. Notice that [1] reports for a 60-dimensional PCA projection layer with a single 1,000-unit ReLU hidden layer for MNIST 90% test accuracy for $\epsilon = 0.5$ and 95% test accuracy for $\epsilon = 2$, both for $\delta = 10^{-5} = 0.6/N$.

Outline: We provide background in Section 2, where we define (ϵ, δ)-differential privacy and explain DP-SGD as introduced by [1]. In Section 3 we explain our main theory, where we start by discussing the theoretical result of the moment accountant method of [1] and its limitation, which we improve leading to our main contribution as given in (1) with a tightness result for (2) based on the f-DP framework. Experiments are in Section 4. The asynchronous SGD framework, detailed differential privacy proofs and analysis, and additional experiments with extra details are in the appendices.

2 Differential Private SGD (DP-SGD)

The optimization problem for training many Machine Learning (ML) models using a training set $\{\xi_i\}_{i=1}^m$ of m samples can be formulated as a finite-sum minimization problem:

$$\min_{w \in \mathbb{R}^d} \left\{ F(w) = \frac{1}{m} \sum_{i=1}^{m} f(w; \xi_i) \right\}. \tag{3}$$

The objective is to minimize a loss function with respect to model parameters w. This problem is known as empirical risk minimization and it covers a wide range of convex and non-convex problems from the ML domain, including, but not limited to, logistic regression, multi-kernel learning, conditional random fields and neural networks.

In this paper, we want to solve (3) in a distributed setting where many clients have their own local data sets and the finite-sum minimization problem is over the collection of all local data sets. A widely accepted approach is to repeatedly use the Stochastic Gradient Descent (SGD) recursion

$$w_{t+1} = w_t - \eta_t \nabla f(w_t; \xi), \tag{4}$$

where w_t represents the model after the t-th iteration; w_t is used in computing the gradient of $f(w_t; \xi)$, where ξ is a data sample randomly selected from the data set $\{\xi_i\}_{i=1}^m$ which comprises the union of all local data sets.

This approach allows each client to perform local SGD recursions for the ξ that belong to the client's local data set. The updates as a result of the SGD recursion (4) are sent to a centralized server who aggregates all received updates and maintains a global model. The server regularly broadcasts its most recent global model so that clients can use it in their local SGD computations. This allows each client to use what has been learned from the local data sets at the other clients. This leads to good accuracy of the final global model.

Each client is doing SGD recursions for a batch of local data. These recursions together represent a local round and at the end of the local round the sum of local model updates, i.e., the addition of computed gradients, is transmitted to the server. The server in turn adds the received sum of local updates to its global model – and once the server receives new sums from all clients, the global model is broadcast to each of the clients. When considering privacy, we are concerned about how much information these sums of local updates reveal about the used local data sets. Each client wants to keep its local data set as private as possible with respect to the outside world which observes round communication.

Rather than reducing the amount of round communication such that less sensitive information is leaked, differential privacy [10, 8, 11, 9] offers a solution in which each client-to-server communication is obfuscated by noise. If the magnitude of the added noise is not too much, then a good accuracy of the global model can still be achieved possibly at the price of more overall SGD iterations. On the other hand, only if the magnitude of the added noise is large enough, then good differential privacy guarantees can be given.

2.1 (ϵ, δ)-Differential Privacy

Differential privacy [10, 8, 11, 9] defines privacy guarantees for algorithms on databases, in our case a client's sequence of mini-batch gradient computations on his/her training data set. The guarantee quantifies into what extent the output of a client (the collection of updates communicated to the server) can be used to differentiate among two adjacent training data sets d and d' (i.e., where one set has one extra element compared to the other set).

Definition 2.1. A randomized mechanism $\mathcal{M} : D \to R$ is (ϵ, δ)-DP (Differentially Private) if for any adjacent d and d' in D and for any subset $S \subseteq R$ of outputs,

$$Pr[\mathcal{M}(d) \in S] \leq e^{\epsilon} Pr[\mathcal{M}(d') \in S] + \delta,$$

where the probabilities are taken over the coin flips of mechanism $\mathcal{M}$.

The privacy loss incurred by observing o is given by

$$L^{o}_{\mathcal{M}(d) \| \mathcal{M}(d')} = \ln \left(\frac{Pr[\mathcal{M}(d) = o]}{Pr[\mathcal{M}(d') = o]} \right).$$

As explained in [11] (ϵ, δ)-DP ensures that for all adjacent d and d' the absolute value of privacy loss will be bounded by ϵ with probability at least $1-\delta$. The larger ϵ the more certain we are about which of d or d' caused observation o. When using differential privacy in machine learning we typically use $\delta = 1/N$ (or $1/(10N)$) inversely proportional with the data set size N.

In order to prevent data leakage from inference attacks in machine learning [19] such as the deep leakage from gradients attack [37, 36, 12] or the membership inference attack [30, 23, 31] a range of privacy-preserving methods have been proposed. Privacy-preserving solutions for federated learning are Local Differential Privacy (LDP) solutions [1, 2, 22, 32, 14, 7] and Central Differential Privacy (CDP) solutions [22, 13, 20, 27, 34]. In LDP, the noise for achieving differential privacy is computed locally at each client and is added to the updates before sending to the server – in this paper we also consider LDP. In CDP, a trusted server (aka trusted third party) aggregates received client updates into a global model; in order to achieve differential privacy the server adds noise to the global model before communicating it to the clients.

2.2 DP-SGD

Algorithm 1 DP-SGD: Local Model Updates with Differential Privacy

procedure LOCALSGDWITHDP(d)
    for $i \in \{0, \ldots, T-1\}$ **do**
        Receive the current global model $\hat{w}$ from Server.
        Uniformly sample a random set $\{\xi_h\}_{h=1}^{s_i} \subseteq d$
        $h = 0, U = 0$
        while $h < s_i$ **do**
            $g = [\nabla f(\hat{w}, \xi_h)]_C$
            $U = U + g$
            h++
        end while
        $n \leftarrow \mathcal{N}(0, C^2\sigma^2\mathbf{I})$
        $U = U + n$
        Send (i, U) to the Server.
    end for
end procedure

We analyse the Gaussian based differential privacy method, called DP-SGD, of [1], depicted in Algorithm 1 in a distributed setting as described above. Rather than using the gradient $\nabla f(\hat{w}, \xi)$ itself, DP-SGD uses its clipped version $[\nabla f(\hat{w}, \xi)]_C$ where $[x]_C = x/\max\{1, \|x\|/C\}$. Clipping is needed because in general we cannot assume a bound C on the gradients (for example, the bounded gradient assumption is in conflict with strong convexity [24]), yet the added gradients need to be bounded by some constant C in order for the DP analysis to go through.

DP-SGD uses a mini-batch approach where before the start of the i-th local round a random min-batch of sample size s_i is selected out of a local data set d of size $|d| = N$. Here, we slighty generalize DP-SGD's original formulation which uses a constant $s_i = s$ sample size sequence, while our analysis will hold for a larger class of sample size sequences. The inner loop maintains the sum U of gradient updates where each of the gradients correspond to the same local model $\hat{w}$ until it is replaced by a newer global model at the start of the outer loop. At the end of each local round the sum of updates U is obfuscated with Gaussian noise $\mathcal{N}(0, C^2\sigma^2)$ added to each vector entry, and the result is transmitted to the server. The noised U is transmitted to the server who adds U times the round step size $\bar{\eta}_i$ to its global model $\hat{w}$ (we discount averaging the sum represented by U by scaling the step size inversely with s_i). As soon as all clients have submitted their updates, the resulting new global model $\hat{w}$ is broadcast to all clients, who in turn replace their local models with the newly received global model (at the start of the outer loop).

2.3 Tight f-DP Framework

Appendix C summarizes the recent work by [6] that introduces the f-DP framework based on hypothesis testing. f-DP has (ϵ, δ)-DP as a special case in that a mechanism is (ϵ, δ)-DP if and only if it is $f_{\epsilon,\delta}$-DP with $f_{\epsilon,\delta}(\alpha) = \min\{0, 1 - \delta e^{\epsilon}\alpha, (1-\delta-\alpha)e^{-\epsilon}\}$. They prove that DP-SGD is $C_{s/N}(G_{(\sigma/2)^{-1}})^{\otimes T}$-DP where $C_{s/N}$ is an operator representing the effect of subsampling, $G_{(\sigma/2)^{-1}}$ is a Gaussian function characterizing the differential privacy (called Gaussian DP) due to adding Gaussian noise, and operator $\otimes T$ describes composition over T rounds. $C_{s/N}(G_{(\sigma/2)^{-1}})^{\otimes T}$-DP can be translated into a tight (ϵ, δ)-DP formulation.

Towards understanding how to a-priori set parameters for best utility and minimal privacy leakage, the tight f-DP formulation for DP-SGD can be translated into sharp privacy guarantees. However, as stated in the introduction by a citation from [6], the expressions it yields are more unwieldy. This leads in [38] to a differential privacy accountant (using a complex characteristic function based on taking the Fourier transform) for a client to understand when to stop helping the server to learn a global model. A differential privacy accountant is a method for keeping track (account for) spent privacy budget.

3 Improved Moment Accountant Method

[1] proves the following main result (rephrased using our notation by substituting $q = s/N$ in their work): There exist constants c_1 and c_2 so that given a constant sample size sequence $s_i = s$ and number of rounds T, for any $\epsilon < c_1 T(s/N)^2$, Algorithm 1 is (ϵ, δ)-DP for any $\delta > 0$ if we choose

$$\sigma \geq c_2 \frac{(s/N) \cdot \sqrt{T \ln(1/\delta)}}{\epsilon}.$$

The interpretation of this result is subtle: The condition on ϵ is equivalent to

$$1/\sqrt{c_1} < z \text{ where } z = (s/N) \cdot \sqrt{T/\epsilon}.$$

Substituting this into the bound for σ yields

$$\sigma \geq (c_2 \cdot z) \cdot \sqrt{\frac{\ln(1/\delta)}{\epsilon}}. \tag{5}$$

This formulation only depends on T through the definition of z. Notice that z may be as small as $1/\sqrt{c_1}$. In fact, it is unclear how z depends on T since T is equal to the total number K of gradient computations over all local rounds performed on the local data set divided by the mini-batch size s, i.e., $T = K/s$, hence, $z = k \cdot \sqrt{1/(T\epsilon)}$ with $k = K/N$. This shows that for fixed K and N, we can increase T as long as $1/\sqrt{c_1} < z$, or equivalently,

$$T < c_1 k^2/\epsilon \tag{6}$$

(notice that the original constraint on ϵ directly translates into this upper bound on T by using $T = K/s$).

3.1 Main Contribution

Rather than applying the main result of [1], we can directly use the moment accountant method of their proof to analyse specific parameter settings. It turns out that T can be much larger than upper bound (6). In this paper we formalize this insight (by showing that 'constants' c_1 and c_2 can be chosen as functions of T and other parameters) and show a lower bound on σ which does not depend on T at all – in fact z in (5) can be characterized as a constant independent of any parameters. This will show that σ can remain small up to a lower bound that only depends on the privacy budget, see (1). Our improvement over the existing moment accountant method is that it can be generalized for analysing the complimentary range $T \geq c_1 k^2/\epsilon$, i.e., for T at least lower bound (2).

Theorem 3.1. *Let σ and (ϵ, δ) satisfy the relation*

$$\sigma = \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon} \text{ with } \delta \leq 1/N \text{ and } \epsilon < 0.5. \tag{7}$$

For sample size sequence $\{s_i\}_{i=0}^{T-1}$ the total number of local SGD iterations is equal to $K = \sum_{i=0}^{T-1} s_i$. We define $k = K/N$ as the total number of local SGD iterations measured in epochs (of size N). Related to the sample size sequence we define the mean $\bar{s}$ and maximum s_{max} and their quotient θ as

$$\bar{s} = \frac{1}{T}\sum_{i=0}^{T-1} s_i = \frac{K}{T}, \quad s_{max} = \max\{s_0, \ldots, s_{T-1}\},$$
$$\text{and} \quad \theta = \frac{s_{max}}{\bar{s}}.$$

Let γ be the smallest solution satisfying

$$\gamma \geq \frac{2}{1-\bar{\alpha}} + \frac{2^4 \cdot \bar{\alpha}}{1-\bar{\alpha}} \left(\frac{\sigma}{(1-\sqrt{\bar{\alpha}})^2} + \frac{1}{\sigma(1-\bar{\alpha}) - 2e\sqrt{\bar{\alpha}}} \frac{e^3}{\sigma} \right) e^{3/\sigma^2}$$
$$\text{with } \bar{\alpha} = \frac{\epsilon}{\gamma k}.$$

Parameter $\gamma = 2 + O(\bar{\alpha})$, which is close to 2 for small $\bar{\alpha}$. We assume data sets of size $N \geq 10000$ and sample size sequences with $\theta \leq 6.85$. If

$$k \geq \sqrt{2\epsilon \ln(1/\delta)} \cdot e/(\gamma\theta) \text{ and} \tag{8}$$

$$T \geq \frac{\gamma\theta^2}{\epsilon} \cdot k^2, \tag{9}$$

then Algorithm 1 is (ϵ, δ)-differentially private.

By substituting $k = K/N$ and by using $K = \bar{s}T$, lower bound (9) is equivalent to the upper bound $T \leq \frac{\epsilon}{\gamma\theta^2}(N/\bar{s})^2$ as is mentioned in the introduction for a constant sample size sequence $s = s_i = \bar{s}$.

All our theory, including the above theorem, holds in the asynchronous SGD framework of Appendix A. In Appendix A we provide a more general *asynchronous mini-batch SGD algorithm* (which follows Hogwild!'s philosophy [28, 4, 35, 24, 16, 25]) *with DP*. The asynchronous setting allows clients to adapt their sample sizes to their processing speed and communication latency.

We notice that polynomial increasing sample size sequences $s_i \sim qNi^p$ have $\bar{s} \approx [qNT^{p+1}/(p+1)]/T$ and $s_{max} = qNT^p$, hence, $\theta = 1 + p$. This shows that our theory covers e.g. linear increasing sample size sequences as discussed in [25], where is explained how this implies reduced round communication – another metric which one may trade-off against accuracy and total local number K of gradient computations.

The proof is in Appendix B and follows a sequence of steps: We discuss the analysis of [1] and explain where we will improve. This leads to an improved analysis with a first generally applicable Theorem B.2. As a consequence we derive a simplified characterization in the form of Theorem B.4. Finally, we introduce more coarse bounds in order to extract the more readable Theorem 3.1. We notice that the simulations in Section 4 are based on parameters that satisfy constraints (32, 33, 34, 62) of Theorem B.4 as this leads to slightly better results.

3.2 Reusing the Local Data Set

We stress that T cannot be chosen arbitrarily large in Theorem 3.1 as it is restricted by $K = kN$. Also k cannot grow arbitrarily large since $kN = K \geq T \geq \frac{\gamma\theta^2}{\epsilon} \cdot k^2$, hence, $k \leq \frac{\epsilon}{\gamma\theta^2} \cdot N$. This upper bound on k does impose a constraint after which (ϵ, δ)-DP cannot be guaranteed – so, K and, hence, T cannot increase indefinitely without violating the privacy budget. Here we notice that repeated use of the same data set over multiple learning problems (one after another) is allowed as long as the number of epochs of gradient computing satisfies the upper bound $k \leq \frac{\epsilon}{\gamma\theta^2} \cdot N$. Hence, the larger N the more collaborative learning tasks the client can participate in. For typical values $\epsilon = 0.2$, $\gamma\theta^2 \approx 2$, and a data set of size $N = 10000$ we have $k \leq 1000$, which may accommodate about 10-20 learning tasks.

3.3 Accuracy

Lower bound (9) shows that we may freely choose $T \geq \gamma\theta^2 k^2/\epsilon$ without affecting the (ϵ, δ)-DP guarantee.* Here, the corresponding standard deviation σ according to (7) only depends on ϵ and δ and does not depend on the actual choice of T as long as it is at least $\gamma\theta^2 k^2/\epsilon$. This allows one to optimize the number of rounds T with constant mini-batch size $s = K/T$ for best accuracy of the final global model.

If all local data sets are iid coming from the same source distribution†, then simulations in Section 4 show that the best accuracy of the final global model is achieved by choosing the largest possible mini-batch size s or, equivalently, since $K = sT$, choosing the smallest possible number of rounds $T = \gamma\theta^2 k^2/\epsilon$ according to lower bound (9). Optimizing accuracy by choosing the smallest T can be understood by observing that this implies that the least number of times noise is added and aggregated into the global model at the server (larger mini-batches imply less noise relative to the size of the mini-batches). As a secondary objective, a smaller number of rounds means less round communication.

3.4 Tightness

Appendix C.2 uses the f-DP framework‡ to prove the tightness of Theorem 3.1:

*Even though more rounds leak more updates, the sample size is smaller leading to less privacy amplification due to subsampling.

†This is the case in big data analysis where each local data set represents a too small sample of the source distribution for learning an accurate local model on its own. Hence, collaboration among multiple clients through a central server is needed to generate an accurate joint global model.

‡We correct for the factor 2 larger Gaussian noise in the f-DP framework because our analysis based on [1] assumes a probabilistic (rather than a deterministic) sampling strategy as implemented in the Opacus library [26].

Theorem 3.2. *For $T = (\gamma\theta^2 k^2/\epsilon)/a$ with constant $a > 4\gamma$, there exists a parameter setting that fits all conditions of Theorem 3.1 except for lower bound (9) such that (ϵ, δ)-DP is violated.*

Theorem 3.2 shows that in Theorem 3.1 T's required lower bound $\gamma\theta^2 k^2/\epsilon$ cannot be made smaller by more than a constant factor $4\gamma \approx 8$ (otherwise, this conflicts with an asymptotical result proved by the f-DP framework). This shows that choosing T equal to the lower bound $\gamma\theta^2 k^2/\epsilon$ is *close to tight* in order to achieve (ϵ, δ)-DP.

Of course, larger T also satisfy lower bound (8) implying the same (ϵ, δ)-DP guarantee. We notice that a larger T can meet $\gamma\theta^2 \cdot k^2/\epsilon'$ for a smaller ϵ' leading to a close to tight (ϵ', δ)-DP guarantee if we choose a larger σ, which can be done if this still leads to sufficient accuracy. Intuitively, a T larger than the lower bound $\gamma\theta^2 k^2/\epsilon$ invests in the potential of improved differential privacy (i.e., $\epsilon' \leq \epsilon$) which we do not need if we only require the (ϵ, δ)-DP guarantee. Better is to sacrifice this potential and meet the lower bound so that accuracy of the final global model is optimized (as discussed above). Since the lower bound can at most be a constant factor $4\gamma \approx 8$ smaller, we cannot improve the accuracy much more by reducing T further without violating the (ϵ, δ)-DP guarantee.

As a theoretical consequence, for fixed k, Appendix C.3 formulates (ϵ, δ)-DP guarantees for varying ϵ which can be used to show f-DP for a (non-trivial) trade-off function f that depends on a target ϵ and δ but does not depend on the choice of T assuming $\gamma\theta^2 k^2/\epsilon \leq T \leq K = kN$.

3.5 Choosing ϵ

We notice that, since $T \leq K = kN$ (this corresponds to the smallest possible mini-batch size $s = 1$), lower bound (9) implies $kN \geq k^2/\epsilon$, hence, $\epsilon \geq k/N$ and we must have $\epsilon = \Omega(1/N)$. Therefore, the smallest possible ϵ is $\Theta(1/N)$ and leads to $\sigma = \Theta(\sqrt{N \ln N})$ according to (7). We notice that the theory in [5] for a similar but not exactly the same setting of DP-SGD strongly suggests for DP-SGD that unless the added Gaussian perturbation is as large as $\sqrt{N}$ almost the whole database can be recovered by a polynomial (in N) adversary; $\sigma = \Theta(\sqrt{N \ln N})$ seems needed if one wants cryptographical strong security. However, in general, $\sigma = \Theta(\sqrt{N \ln N})$ is too large for sufficient accuracy. In practice we choose $\epsilon = \theta(1)$:

In order to attain an accuracy comparable to the non-DP setting where no noise is added, the papers cited in Section 2.1 generally require large ϵ (such that σ can be small enough) – which gives a weak privacy posture (a weak bound on the privacy loss). For example, when considering LDP (see Section 2.1), 10% deduction in accuracy yields only $\epsilon = 50$ in [2] and $\epsilon = 10.7$ in [22], while [32, 14] show solutions for a much lower $\epsilon = 0.5$. Similarly, when considering CDP (see Section 2.1), in order to remain close to the accuracy of the non-DP setting [22] requires $\epsilon = 8.1$, [13] requires $\epsilon = 8$, and [20] requires $\epsilon = 2.038$.

The theory presented in this paper allows relatively small Gaussian noise for small ϵ: We only need to satisfy the main equation (7). For example, in Section 4 simulations for the LIBSVM data set show $(\epsilon = 0.05, \delta = 1/N)$-DP is possible while achieving good accuracy with $\sigma \approx 20$. Such small ϵ is a significant improvement over existing literature.

3.6 Utility Graph for Selecting C, σ, and K

The most important mission in machine learning is achieving a good accuracy, therefore, the added Gaussian noise cannot be too large and is constrained. For this reason each client wants to choose (i) the smallest possible clipping constant C for the clipping operation used in DP-SGD such that SGD still bootstraps convergence, and given C, set (ii) the standard deviation σ of the added Gaussian noise for differential privacy to a maximum value beyond which we cannot expect to achieve sufficiently good (test) accuracy for the learning task at hand, and given C and σ, estimate (iii) the total number of gradient computations $K = kN$ needed to achieve the target test accuracy, and given C, σ, and K, select (iv) $T = \gamma\theta^2 k^2/\epsilon$ and $s = K/T$, where ϵ and $\delta \leq 1/N$ are chosen according to (7). Choosing $T = \gamma\theta^2 k^2/\epsilon$ is close to tight for achieving (ϵ, δ)-DP, and in Theorem 3.1, $T = \gamma\theta^2 k^2/\epsilon$ optimizes the accuracy of the final global model and minimizes the round communication.

Choosing the maximum value for σ in the above procedure leads to the best possible (ϵ, δ)-DP guarantees as this allows the smallest possible ϵ for a given $\delta = 1/N$ or $1/(10N)$. If the smallest possible ϵ is larger than the client's target epsilon ϵ_{tar}, then the client should not participate in the collaborative training of the global model and should abort. If the smallest ϵ is $\leq \epsilon_{tar}$, then the client can participate.

In order to select parameters C, σ and K, we introduce the concept of a utility graph where a "best-case" accuracy is depicted as a function of noise σ and clipping constant C in DP-SGD (see Section 4). In DP-SGD the last round of local updates is aggregated into an update of the global model, after which the global model is finalized. This means that the Gaussian noise added to a client local update of its last round is directly added as a perturbation to the final global model. We have a best-case scenario if we neglect the added noise of all previous rounds. That is, the "best-case" accuracy for DP-SGD is the accuracy of a global model which is trained using SGD without DP after which Gaussian noise is added. The utility graph depicts this "best-case" accuracy.

To generate the graph, we fix a diminishing learning rate $\bar{\eta}_t$ (step size) from round to round and we fix the total number K of local gradient computations that will be performed. Based on local training data and a-priori knowledge (possibly from transferring a public model of another similar learning task), a local client can run SGD locally without any added DP mechanism. This learns a local model w^* and we compare how much accuracy is sacrificed by adding Gaussian noise $n \sim \mathcal{N}(0, C^2\sigma^2\mathbf{I})$; that is, we compute and depict the ratio "$F(w^* + n)/F(w^*)$." This teaches us the range of σ and C combinations that may lead to sufficient accuracy (say at most a 10% drop).

4 Experiments

Our goal is to show that the more general asynchronous differential privacy framework (asynchronous DP-SGD which includes DP-SGD of Algorithm 1) of Appendix A ensures a strong privacy guarantee, i.e, can work with very small ϵ (and $\delta = 1/N$), while having a good convergence rate to good accuracy. We refer to Appendix D for simulation details and complete parameter settings.

Objective function. We summarize experimental results of our asynchronous DP-SGD framework for strongly convex, plain convex and non-convex objective functions with *constant sample size sequences*. As the plain convex objective function we use logistic regression: The weight vector w and bias value b of the logistic function can be learned by minimizing the log-likelihood function J:

$$J = -\sum_{i=1}^{N}[y_i \cdot \log(\bar{\sigma}_i) + (1 - y_i) \cdot \log(1 - \bar{\sigma}_i)],$$

where N is the number of training samples (x_i, y_i) with $y_i \in \{0, 1\}$, and $\bar{\sigma}_i = 1/(1 + e^{-(w^{\mathrm{T}} x_i + b)})$ is the sigmoid function. The goal is to learn a vector/model w^* which represents a pair $\bar{w} = (w, b)$ that minimizes J. Function J changes into a strongly convex problem by adding ridge regularization with a regularization parameter $\lambda > 0$, i.e., we minimize $\hat{J} = J + \frac{\lambda}{2}\|\bar{w}\|^2$ instead of J. For simulating non-convex problems, we choose a simple neural network (LeNet) [18] with cross entropy loss function for image classification.

Asynchronous DP-SGD setting. The experiments are conducted with 5 compute nodes and 1 central server. For simplicity, the compute nodes have iid data sets.

Data sets. All our experiments are conducted on LIBSVM [3][§] and MNIST [17][¶] data sets.

4.1 Utility Graph

Since we do not have a closed form to describe the relation between the utility of the model (i.e., prediction accuracy) and σ, we propose a heuristic approach to learn the range of σ from which we may select σ for finding the best (ϵ, δ)-DP. The utility graphs – Figures 1(a), 2(a) and 3(a) – show the fraction of test accuracy between the model "$F(w + n)$" over the original model $F(w)$ (without noise), where $n \sim \mathcal{N}(0, C^2\sigma^2\mathbf{I})$ (per round) for various values of the clipping constant C and noise standard deviation σ. Intuitively, the closer $F(w+n)/F(w)$ to 1, the better accuracy wrt to $F(w)$. Note that w can be any solution and in the utility graphs, we choose $w = w^*$ with w^* being near to an optimal solution.

The smaller C, the larger σ can be, hence, ϵ can be smaller which gives stronger privacy. However, the smaller C, the more iterations (larger K) are needed for convergence. And if selected too small, no fast enough convergence is possible. In next experiments we use clipping constant $C = 0.1$, which gives a drop of at most 10% in test accuracy for $\sigma \leq 20$ for both strongly convex and plain convex objective functions. To keep the test accuracy loss $\leq 10\%$ in the non-convex case, we choose $C = 0.025$ and $\sigma \leq 12$.

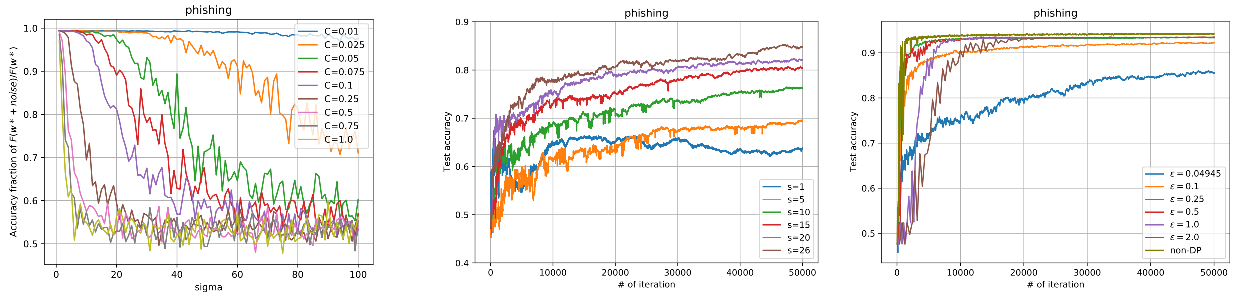
4.2 DP-SGD with Different Constant Sample Sizes

Figure 1(b) and Figure 2(b) illustrate the test accuracy of our asynchronous DP-SGD with various constant sample sizes for the strongly convex and plain convex cases. Here, we use privacy budget $\epsilon = 0.04945$ and noise $\sigma = 19.2$. It is clear that with $s = 1$. When we use a bigger constant sample size s, for example, $s = 26$, our algorithm can achieve the desired performance, when compared to other constant sample sizes.[‖] The experiment is extended to the

[§]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html

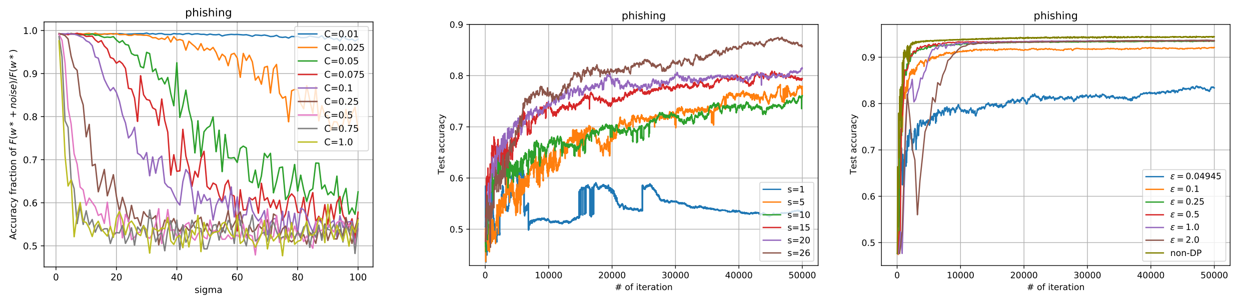
[¶]http://yann.lecun.com/exdb/mnist/

[‖]$s = 26$ meets the lower bound on T; a larger s violates this lower bound. The reason for having a somewhat small maximum possible s is because of the relatively small data set size.



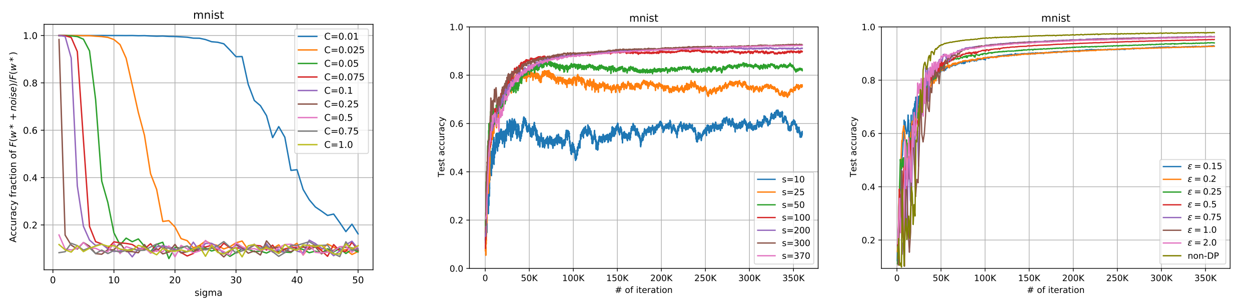
(a) (b) (c)

Figure 1: Strongly convex. (a) Utility graph, (b) Different s, (c) Different ϵ



(a) (b) (c)

Figure 2: Plain convex. (a) Utility graph, (b) Different s, (c) Different ϵ



(a) (b) (c)

Figure 3: Non-convex. (a) Utility graph, (b) Different s, (c) Different ϵ

non-convex case as shown in Figure 3(b), where we can see a similar pattern. Experimental results for other data sets are in Supplemental Material D. This confirms that our DP-SGD framework can converge to a decent accuracy while achieving a very small privacy budget ϵ.

4.3 DP-SGD with Different Levels of Privacy Budget

Figure 1(c) and Figure 2(c) show that our DP-SGD framework converges to better accuracy if ϵ is slightly larger. E.g., in the strongly convex case, privacy budget $\epsilon = 0.04945$ achieves test accuracy 86% compared to 93% without differential privacy (hence, no added noise); $\epsilon = 0.1$, still significantly smaller than what is reported in literature, achieves test accuracy 91%. Figure 3(c) shows the test accuracy of our asynchronous DP-SGD for different privacy budgets ϵ in the non-convex case. For $\epsilon = 0.15$, our framework can achieve the test accuracy about 93%, compared to 98% without differential privacy. These figures again confirm the effectiveness of our DP-SGD framework, which can obtain a strong differential privacy guarantee.

5 Conclusion

We improved the moment account method and provided a simple characterization of parameters that achieves (ϵ, δ)-DP, which we prove is close to tight in that T cannot be more than a factor ≈ 8 smaller. Choosing T as small as possible optimizes accuracy and minimizes round complexity.

References

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

[2] Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. Protection against reconstruction and its applications in private federated learning, 2019.

[3] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[4] Christopher M De Sa, Ce Zhang, Kunle Olukotun, and Christopher Ré. Taming the wild: A unified analysis of hogwild-style algorithms. In *NIPS*, pages 2674–2682, 2015.

[5] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.

[6] Jinshuo Dong, Aaron Roth, and Weijie Su. Gaussian differential privacy. *Journal of the Royal Statistical Society*, 2021.

[7] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy, data processing inequalities, and statistical minimax rates, 2014.

[8] Cynthia Dwork. A firm foundation for private data analysis. *Communications of the ACM*, 54(1):86–95, 2011.

[9] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503. Springer, 2006.

[10] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[11] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[12] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients – how easy is it to break privacy in federated learning? In *NIPS*, 2020.

[13] Robin C. Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective, 2018.

[14] Meng Hao, Hongwei Li, Xizhao Luo, Guowen Xu, Haomiao Yang, and Sen Liu. Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Transactions on Industrial Informatics*, 16(10): 6532–6542, 2020. doi: 10.1109/TII.2019.2945367.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[16] Rémi Leblond, Fabian Pedregosa, and Simon Lacoste-Julien. Improved asynchronous parallel optimization analysis for stochastic incremental methods. *JMLR*, 19(1):3140–3207, 2018.

[17] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

[20] Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*, 2018. URL `https://openreview.net/pdf?id=BJ0hF1Z0b`.

[21] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *ICLR Workshop Track*, 2016.

[22] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy, 2021.

[23] M. Nasr, R. Shokri, and A. Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 739–753, 2019. doi: 10.1109/SP.2019.00065.

[24] Lam Nguyen, Phuong Ha Nguyen, Marten Dijk, Peter Richtárik, Katya Scheinberg, and Martin Takác. Sgd and hogwild! convergence without the bounded gradients assumption. In *International Conference on Machine Learning*, pages 3750–3758. PMLR, 2018.

[25] Nhuong Nguyen, Toan Nguyen, Phuong Ha Nguyen, Quoc Tran-Dinh, Lam Nguyen, and Marten Dijk. Hogwild! over distributed local data sets with linearly increasing mini-batch sizes. In *International Conference on Artificial Intelligence and Statistics*, pages 1207–1215. PMLR, 2021.

[26] Opacus. Opacus PyTorch library. URL `https://opacus.ai`.

[27] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. In *International conference on learning representations*, 2018.

[28] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in neural information processing systems*, pages 693–701, 2011.

[29] Nicolas Roux, Mark Schmidt, and Francis Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[30] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

[31] L. Song, R. Shokri, and P. Mittal. Membership inference attacks against adversarially robust deep learning models. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 50–56, 2019. doi: 10.1109/SPW.2019.00021.

[32] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, pages 1–11, 2019.

[33] Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010.

[34] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. *2019 IEEE Symposium on Security and Privacy (SP)*, May 2019. doi: 10.1109/sp.2019.00019. URL `http://dx.doi.org/10.1109/SP.2019.00019`.

[35] Huan Zhang, Cho-Jui Hsieh, and Venkatesh Akella. Hogwild++: A new mechanism for decentralized asynchronous stochastic gradient descent. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 629–638. IEEE, 2016.

[36] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients, 2020.

[37] Ligeng Zhu, Zhijian Liu, and Song Han. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[38] Yuqing Zhu, Jinshuo Dong, and Yu-Xiang Wang. Optimal accounting of differential privacy via characteristic function. *arXiv preprint arXiv:2106.08567*, 2021.

A Asynchronous Mini-Batch DP-SGD

Algorithms** 2, 3, and 4 explain in pseudo code our asynchronous LDP approach. It is based on the Hogwild! [28] recursion

$$w_{t+1} = w_t - \eta_t \nabla f(\hat{w}_t; \xi_t), \tag{10}$$

where $\hat{w}_t$ represents the vector used in computing the gradient $\nabla f(\hat{w}_t; \xi_t)$ and whose vector entries have been read (one by one) from an aggregate of a mix of previous updates that led to w_j, $j \leq t$. In a single-thread setting where updates are done in a fully consistent way, i.e. $\hat{w}_t = w_t$, yields SGD with diminishing step sizes $\{\eta_t\}$.

Recursion (10) models asynchronous SGD. The amount of asynchronous behavior that can be tolerated is given by some function $\tau(t)$, see [24] where this is analysed for strongly convex objective functions: We say that the sequence $\{w_t\}$ is consistent with delay function τ if, for all t, vector $\hat{w}_t$ includes the aggregate of the updates up to and including those made during the $(t - \tau(t))$-th iteration, i.e.,

$$\hat{w}_t = w_0 - \sum_{j \in \mathcal{U}} \eta_j \nabla f(\hat{w}_j; \xi_j)$$

for some $\mathcal{U}$ with $\{0, 1, \ldots, t - \tau(t) - 1\} \subseteq \mathcal{U}$.

In Algorithm 4 the local SGD iterations all compute gradients based on the same local model $\hat{w}$, which gets substituted by a newer global model $\hat{v}_k$ as soon as it is received by the interrupt service routine ISRRECEIVE. As explained in ISRRECEIVE $\hat{v}_k$ includes all the updates from all the clients up to and including their local rounds $\leq k$. This shows that locally the delay τ can be estimated based on the current local round i together with k. Depending on how much delay can be tolerated SETUP defines $\Upsilon(k, i)$ to indicate whether the combination (k, i) is permissible (i.e., the corresponding delay aka asynchronous behavior can be tolerated). It has been shown that for strongly convex objective functions (without DP enhancement) the convergence rate remains optimal even if the delay $\tau(t)$ is as large as $\approx \sqrt{t/\ln t}$ [24]. Similar behavior has been reported for plain convex and non-convex objective functions in [25].

In Algorithm 4 we assume that messages/packets never drop; they will be resent but can arrive out of order. This guarantees that we get out of the "while $\Upsilon(k, i)$ is false loop" because at some moment the server receives all the updates in order to broadcast a new global model $\hat{v}_{k+1}$ and once received by ISRRECEIVE this will increment k and make $\Upsilon(k, i)$ true which allows LOCALSGDWITHDP to exit the wait loop. As soon as the wait loop is exited we know that all local gradient computations occur when $\Upsilon(k, i)$ is true which reflect that these gradient computations correspond to delays that are permissible (in that we still expect convergence of the global model to good accuracy).

Algorithm 2 Client – Local model with Differential Privacy

1: **procedure** SETUP(n):
  Initialize sample size sequence $\{s_i\}_{i=0}^T$, (diminishing) round step sizes $\{\bar{\eta}_i\}_{i=0}^T$, and a default global model $\hat{v}_0$ to start with.
  Define a permissible delay function $\Upsilon(k, i) \in \{\textbf{True}, \textbf{False}\}$ which takes the current local round number i and the round number k of the last received global model into account to find out whether local SGD should wait till a more recent global model is received. $\Upsilon(\cdot, \cdot)$ can also make use of knowledge of the sample size sequences used by each of the clients.
2: **end procedure**

In this paper we analyse the Gaussian based differential privacy method of [1]. We use their clipping method; rather than using the gradient $\nabla f(\hat{w}, \xi)$ itself, we use its clipped version $[\nabla f(\hat{w}, \xi)]_C$ where $[x]_C = x / \max\{1, \|x\|/C\}$. Also, we use the same mini-batch approach where before the start of the i-th local round a random min-batch of sample size s_i is selected. During the inner loop the sum of gradient updates is maintained where each of the gradients correspond to the same local model $\hat{w}$ until it is replaced by a newer global model. In supplementary material B we show that this is needed for proving DP guarantees and that generalizing the algorithm by locally implementing the Hogwild! recursion itself (which updates the local model each iteration) does not work together with the DP analysis. So, our approach only uses the Hogwild! concept at a global round by round interaction level.

At the end of each local round the sum of updates U is obfuscated with Gaussian noise; Gaussian noise $\mathcal{N}(0, C^2\sigma_i^2)$ is added to each vector entry. In this general description σ_i is round dependent, but our DP analysis in Supplementary Material B must from some point onward assume a constant $\sigma = \sigma_i$ over all rounds. The noised U times the round step size $\bar{\eta}_i$ is added to the local model after which a new local round starts again.

**Our pseudocode uses the format from [25].

Algorithm 3 Client – Local model with Differential Privacy

procedure ISRRECEIVE($\hat{v}_k$):

This Interrupt Service Routine is called whenever a new broadcast global model $\hat{v}_k$ is received from the server. Once received, *the client's local model* $\hat{w}$ *is replaced with* $\hat{v}_k$ (if no more recent global model $\hat{v}_{>k}$ was received out of order before receiving this $\hat{v}_k$)

The server broadcasts global model $\hat{v}_k$ for global round number k once the updates corresponding to local round numbers $\leq k-1$ from *all* clients have been received and have been aggregated into the global model. The server aggregates updates from clients into the current global model as soon as they come in. This means that $\hat{v}_k$ includes all the updates from all the clients up to and including their local round numbers $\leq k-1$ and potentially includes updates corresponding to later round numbers from subsets of clients. The server broadcasts the global round number k together with $\hat{v}_k$.

end procedure

Algorithm 4 Client – Local model with Differential Privacy

procedure LOCALSGDWITHDP(d)
  $i = 0$, $\hat{w} = \hat{v}_0$
  while True do
    while $\Upsilon(k, i) =$ **False do** nothing **end** ▷ k is the global round at the server.
    Uniformly sample a random set $\{\xi_h\}_{h=1}^{s_i} \subseteq d$
    $h = 0, U = 0$
    while $h < s_i$ **do**
      $g = [\nabla f(\hat{w}, \xi_h)]_C$
      $U = U + g$
      h++
    end while
    $n \leftarrow \mathcal{N}(0, C^2\sigma_i^2\mathbf{I})$
    $U = U + n$
    $\hat{w} = \hat{w} + \bar{\eta}_i \cdot U$
    Send (i, U) to the Server.
    i++
  end while
end procedure

The noised U is also transmitted to the server who adds U times the round step size $\bar{\eta}_i$ to its global model $\hat{v}$. As soon as all clients have submitted their updates up to and including their local rounds $\leq k-1$, the global model $\hat{v}$, denoted as $\hat{v}_k$, is broadcast to all clients, who in turn replace their local models with the newly received global model. Notice that $\hat{v}_k$ may include updates from a subset of client that correspond to local rounds $\geq k$.

The presented algorithm adapts to asynchronous behavior in the following two ways: We explained above that the broadcast global models $\hat{v}_k$ themselves include a mix of received updates that correspond to local rounds $\geq k$ – this is due to asynchronous behavior. Second, the sample size sequence $\{s_i\}$ does not necessarily need to be fixed a-priori during SETUP (the round step size sequence $\{\bar{\eta}_i\}$ does need to be fixed a-priori). In fact, the client can adapt its sample sizes s_i on the fly to match its speed of computation and communication latency. This allows the client to adapt its local mini-batch SGD to its asynchronous behavior due to the scheduling of its own resources. Our DP analysis holds for a wide range of varying sample size sequences.

We notice that adapting sample size sequences on a per client basis still fits the same overall objective function as long as all local data sets are iid: This is because iid implies that the execution of the presented algorithm can be cast in a single Hogwild! recursion where the ξ_h are uniformly chosen from a common data source distribution $\mathcal{D}$. This corresponds to the stochastic optimization problem

$$\min_{w \in \mathbb{R}^d} \{F(w) = \mathbb{E}_{\xi \sim \mathcal{D}}[f(w; \xi)]\},$$

which defines objective function F (independent of the locally used sample size sequences). Local data sets being iid in the sense that they are all, for example, drawn from car, train, boat, etc images benefit from DP in that car details (such as an identifying number plate), boat details, etc. need to remain private.

B Differential privacy proofs

This appendix proves a key observation improving the DP moment accountant from [1]. As shown in [1], for *any given* $T \geq 0$ in one specific setting, there are many choices for $(\epsilon, \delta, \sigma)$ depending on two constants (c_1, c_2) (see Theorem B.1). We re-frame the problem as for a given $(\epsilon, \delta, \sigma)$ there are many choices T depending on K and sample size sequence s, where $T \geq k^2/(const \cdot \epsilon)$ (see Theorem 3.1).

This appendix provides the proof of Theorem 3.1. It follows a sequence of steps: In Section B.1 we discuss the analysis of [1] and explain where we will improve. This leads in Section B.2 to an improved analysis yielding a first generally applicable Theorem B.2; DP definitions/tools with a key lemma (generalized from [1]) are discussed in Section B.2.1 and the proof of Theorem B.2 is in Section B.2.2. As a consequence we derive in Section B.3 a simplified characterization in the form of Theorem B.4. Finally, we introduce more coarse bounds in order to extract the more readable Theorem 3.1 in Section B.4.

B.1 DP-SGD Analysis by Abadi et al.

[1] proves the following theorem (rephrased using our notation substituting $q = s/N$):

Theorem B.1. *There exist constants c_1 and c_2 so that given a sample size sequence $s_i = s$ and number of steps T, for any $\epsilon < c_1 T (s/N)^2$, Algorithm 1 is (ϵ, δ)-differentially private for any $\delta > 0$ if we choose*

$$\sigma \geq c_2 \frac{(s/N) \cdot \sqrt{T \ln(1/\delta)}}{\epsilon}.$$

The interpretation of Theorem B.1, however, subtle: The condition on ϵ in Theorem B.1 is equivalent to

$$1/\sqrt{c_1} < z \text{ where } z = (s/N) \cdot \sqrt{T/\epsilon}.$$

Substituting this into the bound for σ yields

$$\sigma \geq (c_2 \cdot z) \cdot \sqrt{\frac{\ln(1/\delta)}{\epsilon}}. \tag{11}$$

This formulation only depends on T through the definition of z. Notice that z may be as small as $1/\sqrt{c_1}$. In fact, it is unclear how z depends on T since T is equal to the total number K of gradient computations over all local rounds performed on the local data set divided by the mini-batch size s, i.e., $T = K/s$, hence, $z = (K/N) \cdot \sqrt{1/(T\epsilon)}$. This shows that for fixed K and N, we can increase T as long as $1/\sqrt{c_1} < z$, or equivalently,

$$T < c_1 (K/N)^2/\epsilon \tag{12}$$

(notice that the original constraint on ϵ in Theorem B.1 directly translates into this upper bound on T by using $T = K/s$). Since σ cannot be chosen too large (otherwise the final global model has too much noise), ϵ, see (11), cannot be very small. Therefore, 912) puts an upper bound on T which is in general much less than K for practically sized large data sets (K equals the maximum possible number of rounds for mini-batch size $s = 1$).

Rather than applying Theorem B.1, we can directly use the moment accountant method of its proof to analyse specific parameter settings. It turns out that T can be much larger than upper bound (12). In this paper we formalize this insight (by showing that 'constants' c_1 and c_2 can be chosen as functions of T and other parameters) and show a lower bound on σ which does not depend on T at all – in fact z in (11) can be characterized as a constant independent of any parameters. This will show that σ can remain small to at least a lower bound that only depends on the privacy budget. We can freely increase T (up to K if needed) in order to improve accuracy.

B.2 A General Improved DP-SGD Analysis

We generalize Theorem B.1 [1]:

Theorem B.2. *We assume that $\sigma = \sigma_i$ with $\sigma \geq 216/215$ for all rounds i. Let*

$$r = r_0 \cdot 2^3 \cdot \left(\frac{1}{1-u_0} + \frac{1}{1-u_1} \frac{e^3}{\sigma^3} \right) e^{3/\sigma^2} \text{ with } u_0 = \frac{2\sqrt{r_0 \sigma}}{\sigma - r_0} \text{ and } u_1 = \frac{2e\sqrt{r_0 \sigma}}{(\sigma - r_0)\sigma},$$

where r_0 is such that it satisfies

$$r_0 \leq 1/e, \; u_0 < 1, \; \textit{and } u_1 < 1.$$

Let the sample size sequence satisfy $s_i/N \leq r_0/\sigma$. For $j = 1, 2, 3$ we define $\hat{S}_j$ (resembling an average over the sum of j-th powers of s_i/N) with related constants ρ and $\hat{\rho}$:

$$\hat{S}_j = \frac{1}{T}\sum_{i=0}^{T-1}\frac{s_i^j}{N(N-s_i)^{j-1}}, \quad \frac{\hat{S}_1\hat{S}_3}{\hat{S}_2^2} \leq \rho \text{ and } \frac{\hat{S}_1^2}{\hat{S}_2} \leq \hat{\rho}.$$

Let $\epsilon = c_1 T\hat{S}_1^2$. Then, Algorithm 4 is (ϵ, δ)-differentially private if

$$\sigma \geq \frac{2}{\sqrt{c_0}}\frac{\sqrt{\hat{S}_2 T(\epsilon + \ln(1/\delta))}}{\epsilon} \text{ where } c_0 = c(c_1) \text{ with } c(x) = \min\left\{\frac{\sqrt{2r\rho x + 1} - 1}{r\rho x}, \frac{2}{\hat{\rho}x}\right\}.$$

This generalizes Theorem B.1 where all $s_i = s$ are constant. First, Theorem B.2 covers a much broader class of sample size sequences that satisfy bounds on their 'moments' $\hat{S}_j$ (this is more clear as a consequence of Theorem B.2). Second, our detailed analysis provides a tighter bound in that it makes the relation between "constants" c_0 and c_1 explicit, contrary to [1]. Exactly due to this relation $c_0 = c(c_1)$ we are able to prove in Appendix B.3 Theorem B.4 as a consequence of Theorem B.2 by considering the case $c(c_1) = 2/(\hat{\rho}c_1)$.

In order to prove Theorem B.2, we first set up the differential privacy framework of [1] in Appendix B.2.1. Here we enhance a core lemma by proving a concrete bound rather than an asymptotic bound on the so-called λ-th moment which plays a crucial role in the differential privacy analysis. The concrete bound makes explicit the higher order error term in [1].

In Appendix B.2.2 we generalize Theorem B.1 of [1] by proving Theorem B.4 using the core lemma of Appendix B.2.1.

B.2.1 Definitions and Main Lemma

We base our proofs on the framework and theory presented in [1]. In order to be on the same page we repeat and cite word for word their definitions:

For neighboring databases d and d', a mechanism $\mathcal{M}$, auxiliary input $\texttt{aux}$, and an outcome o, define the privacy loss at o as

$$c(o; \mathcal{M}, \texttt{aux}, d, d') = \ln\frac{Pr[\mathcal{M}(\texttt{aux}, d) = o]}{Pr[\mathcal{M}(\texttt{aux}, d') = o]}.$$

For a given mechanism $\mathcal{M}$, we define the λ-th moment $\alpha_{\mathcal{M}}(\lambda; \texttt{aux}, d, d')$ as the log of the moment generating function evaluated at the value λ:

$$\alpha_{\mathcal{M}}(\lambda; \texttt{aux}, d, d') = \ln \mathbf{E}_{o\sim\mathcal{M}(\texttt{aux},d)}[\exp(\lambda \cdot c(o; \mathcal{M}, \texttt{aux}, d, d'))].$$

We define

$$\alpha_{\mathcal{M}}(\lambda) = \max_{\texttt{aux},d,d'} \alpha_{\mathcal{M}}(\lambda; \texttt{aux}, d, d')$$

where the maximum is taken over all possible $\texttt{aux}$ and all the neighboring databases d and d'.

We first take Lemma 3 from [1] and make explicit their order term $O(q^3\lambda^3/\sigma^3)$ with $q = s_{i,c}$ and $\sigma = \sigma_i$ in our notation. The lemma considers as mechanism $\mathcal{M}$ the i-th round of gradient updates and we abbreviate $\alpha_{\mathcal{M}}(\lambda)$ by $\alpha_i(\lambda)$. The auxiliary input of the mechanism at round i includes all the output of the mechanisms of previous rounds (as in [1]).

For the local mini-batch SGD the mechanism $\mathcal{M}$ of the i-th round is given by

$$\mathcal{M}(\texttt{aux}, d) = \sum_{h=0}^{s_i-1}[\nabla f(\hat{w}, \xi_h)]_C + \mathcal{N}(0, C^2\sigma_i^2\mathbf{I}),$$

where $\hat{w}$ is the local model at the start of round i which is replaced by a new global model $\hat{v}$ as soon as a new $\hat{v}$ is received from the server (see ISRReceive), and where ξ_h are drawn from training data d, and $[.]_C$ denotes clipping (that is $[x]_C = x/\max\{1, \|x\|_2/C\}$). In order for $\mathcal{M}$ to be able to compute its output, it needs to know the global models received in round i and it needs to know the starting local model $\hat{w}$. To make sure $\mathcal{M}$ has all this information, $\texttt{aux}$ represents the collection of all outputs generated by the mechanisms of previous rounds $< i$ together with the global models received in round i itself.

In the next subsection we will use the framework of [1] and apply its composition theory to derive bounds on the privacy budget (ϵ, δ) for the whole computation consisting of T rounds that reveal the outputs of the mechanisms for these T rounds as described above.

We remind the reader that s_i/N is the probability of selecting a sample from a sample set (batch) of size s_i out of a training data set d' of size $N = |d'|$; σ_i corresponds to the $\mathcal{N}(0, C^2\sigma_i^2\mathbf{I})$ noise added to the mini-batch gradient computation in round i (see the mechanism described above).

Lemma B.3. *Assume a constant $r_0 < 1$ and deviation $\sigma_i \geq 216/215$ such that $s_i/N \leq r_0/\sigma_i$. Suppose that λ is a positive integer with*

$$\lambda \leq \sigma_i^2 \ln \frac{N}{s_i\sigma_i}$$

and define

$$U_0(\lambda) = \frac{2\sqrt{\lambda r_0/\sigma_i}}{\sigma_i - r_0} \text{ and } U_1(\lambda) = \frac{2e\sqrt{\lambda r_0/\sigma_i}}{(\sigma_i - r_0)\sigma_i}.$$

Suppose $U_0(\lambda) \leq u_0 < 1$ and $U_1(\lambda) \leq u_1 < 1$ for some constants u_0 and u_1. Define

$$r \;=\; r_0 \cdot 2^3 \left(\frac{1}{1-u_0} + \frac{1}{1-u_1} \frac{e^3}{\sigma_i^3} \right) \exp(3/\sigma_i^2).$$

Then,

$$\alpha_i(\lambda) \leq \frac{s_i^2\lambda(\lambda+1)}{N(N-s_i)\sigma_i^2} + \frac{r}{r_0} \cdot \frac{s_i^3\lambda^2(\lambda+1)}{N(N-s_i)^2\sigma_i^3}.$$

Proof. The start of the proof of Lemma 3 in [1] implicitly uses the proof of Theorem A.1 in [11], which up to formula (A.2) shows how the 1-dimensional case translates into a privacy loss that corresponds to the 1-dimensional problem defined by μ_0 and μ_1 in the proof of Lemma 3 in [1], and which shows at the end of the proof of Theorem A.1 (p. 268 [11]) how the multi-dimensional problem transforms into the 1-dimensional problem. In the notation of Theorem A.1, $f(D) + \mathcal{N}(0, \sigma^2\mathbf{I})$ represents the general (random) mechanism $\mathcal{M}(D)$, which for Lemma 3 in [1]'s notation should be interpreted as the batch computation

$$\mathcal{M}(d) = \sum_{h \in J} f(d_h) + \mathcal{N}(0, \sigma^2\mathbf{I})$$

for a random sample/batch $\{d_h\}_{h\in J}$. Here, $f(d_h)$ (by abuse of notation – in this context f does not represent the objective function) represent clipped gradient computations $\nabla f(\hat{w}; d_h)$ where $\hat{w}$ is the last received global model with which round i starts (Lemma 3 in [1] uses clipping constant $C = 1$, hence $\mathcal{N}(0, C^2\sigma^2\mathbf{I}) = \mathcal{N}(0, \sigma^2\mathbf{I})$).

Let us detail the argument of the proof of Lemma 3 in [1] in order to understand what flexibility is possible: We consider two data sets $d = \{d_1, \ldots, d_{N-1}\}$ and $d' = d + \{d_N\}$, where $d_N \notin d$ represents a new data base element so that d and d' differ in exactly one element. The size of d' is equal to N. We define vector x as the sum

$$x = \sum_{J\setminus\{N\}} f(d_i).$$

Let

$$z = f(d_N).$$

If we consider data set d, then sample set $J \subseteq \{1, \cdots, N-1\}$ and mechanism $\mathcal{M}(d)$ returns

$$\mathcal{M}(d) = \sum_{h\in J} f(d_h) + \mathcal{N}(0, \sigma^2\mathbf{I}) = \sum_{h\in J\setminus\{N\}} f(d_h) + \mathcal{N}(0, \sigma^2\mathbf{I}) = x + \mathcal{N}(0, \sigma^2\mathbf{I}).$$

If we consider data set d', then $J \subseteq \{1, \cdots, N\}$ contains d_N with probability $q = |J|/N$ ($|J| = s_i$ is the sample size used in round i). In this case mechanism $\mathcal{M}(d')$ returns[††]

$$\mathcal{M}(d') = \sum_{h \in J} f(d_h) + \mathcal{N}(0, \sigma^2 \mathbf{I}) = f(d_N) + \sum_{h \in J \setminus \{N\}} f(d_h) + \mathcal{N}(0, \sigma^2 \mathbf{I}) = z + x + \mathcal{N}(0, \sigma^2 \mathbf{I})$$

with probability q. It returns

$$\mathcal{M}(d') = \sum_{h \in J} f(d_h) + \mathcal{N}(0, \sigma^2 \mathbf{I}) = \sum_{h \in J \setminus \{N\}} f(d_h) + \mathcal{N}(0, \sigma^2 \mathbf{I}) = x + \mathcal{N}(0, \sigma^2 \mathbf{I})$$

with probability $1 - q$. Combining both cases shows that $\mathcal{M}(d')$ represents a mixture of two Gaussian distributions (shifted over a vector x):

$$\mathcal{M}(d') = x + (1 - q) \cdot \mathcal{N}(0, \sigma^2 \mathbf{I}) + q \cdot \mathcal{N}(z, \sigma^2 \mathbf{I}).$$

This high dimensional problem is transformed into a single dimensional problem at the end of the proof of Theorem A.1 (p. 268 [11]) by considering the one dimensional line from point x into the direction of z, i.e., the line through points x and $x + z$; the one dimensional line maps x to the origin 0 and $x + z$ to $\|z\|_2$. $\mathcal{M}(d)$ as wells as $\mathcal{M}(d')$ projected on this line are distributed as

$$\mathcal{M}(d) \sim \mu_0 \text{ and } \mathcal{M}(d') \sim (1 - q)\mu_0 + q\mu_1,$$

where

$$\mu_0 \sim \mathcal{N}(0, \sigma^2) \text{ and } \mu_1 \sim \mathcal{N}(\|z\|_2, \sigma^2).$$

In [1] as well as in this paper the gradients are clipped (their Lemma 3 uses clipping constant $C = 1$) and this implies

$$\|z\|_2 = \|f(d_N)\|_2 \leq C = 1.$$

Their analysis continues by assuming the worst-case in differential privacy, that is,

$$\mu_1 \sim \mathcal{N}(1, \sigma^2).$$

Notice that the above argument analyses a local mini-batch SGD computation. Rather than using a local mini-batch SGD computation, can we use clipped SGD iterations which continuously update the local model:

$$\hat{w}_{h+1} = \hat{w}_h - \eta_h \nabla [f(\hat{w}_h, \xi_h)]_C.$$

This should lead to faster convergence to good accuracy compared to a local minibatch computation. However, the above arguments cannot proceed[‡‡] because (in the notation used above where the d_h, $h \in J$, are the ξ_h, $h \in \{0, \ldots, s_i - 1 = |J| - 1\}$) selecting sample d_N in iteration h does not only influence the update computed in iteration h but also influences all iterations after h till the end of the round (because $f(d_N)$ updates the local model in iteration h which is used in the iterations that come after). Hence, the dependency on d_N is directly felt by $f(d_N)$ in iteration h and indirectly felt in the $f(d_j)$ that are computed after iteration h. This means that we cannot represent distribution $\mathcal{M}(d')$ as a clean mix of Gaussian distributions with a mean z, whose norm is bounded by the clipping constant.

The freedom which we do have is replacing the local model by a newly received global model. This is because the updates $f(d_h)$, $h \in J$, computed locally in round i have not yet been transmitted to the server and, hence, have not been aggregated into the global model that was received. In a way the mechanism $\mathcal{M}(d)$ is composed of two (or multiple if more newer and newer global models are received during the round) sums

$$\mathcal{M}(d) = \sum_{h \in J_0} f_0(d_h) + \sum_{h \in J_1} f_1(d_h) + \mathcal{N}(0, \sigma^2 \mathbf{I}),$$

[††]This is actually a subtle argument: We do not have fixed constant sample sizes, instead we have probabilistic sample sizes with a predetermined expectation. The idea is to add each data element to the sample with probability s_i/N. This means that the sample size is equal to s_i in expectation. This allows one to compare two samples that differ in exactly one element d_N (as is done in this argument). If one uses fixed constant sample sizes, then $\mathcal{M}(d') = x + (1 - q) \cdot \mathcal{N}(0, \sigma^2 \mathbf{I}) + q \cdot \mathcal{N}(f(d_N) - f(d_h), \sigma^2 \mathbf{I})$ for $z = f(d_N)$ and some $h \in J$. Now $\|f(d_N) - f(d_h)\|_2 \leq \|f(d_N)\|_2 + \|f(d_h)\|_2 \leq 2C = 2$ (for $C = 1$) and we pay a factor 2 penalty. In the f-DP framework we actually consider the latter and work with fixed (non-probabilistic) constant sample sizes. In this paper and, what should have been assumed in [1] and is actually implemented in the Opacus library [26], we assume a probabilistic sample size and safe the factor 2. We notice that even if we aim at a constant sample size sequence with sample sizes s, we can reinterpret the s_i as the actual chosen probabilistic sample size with $\mathbb{E}[s_i] = s$ and apply our theory that holds for varying sample size sequences.

[‡‡]Unless we assume a general upper bound on the norm of the Hessian of the objective function which should be large enough to cover a wide class of objective functions and small enough in order to be able to derive practical differential privacy guarantees.

where $J = J_0 \cup J_1$ and J_0 represent local gradient computations, shown by $f_0(.)$, based on the initial local model $\hat{w}$ and J_1 represent the local gradient computations, shown by $f_1(.)$, based on the newly received global model $\hat{v}$ which replaces $\hat{w}$. As one can verify, the above arguments are still valid for this slight adaptation. As in Lemma 3 in [1] we can now translate our privacy loss to the 1-dimensional problem defined by $\mu_0 \sim \mathcal{N}(0, C^2\sigma^2)$ and $\mu_1 \sim \mathcal{N}(C, C^2\sigma^2)$ for $\|\nabla f(.,.)\|_2 \leq C$ as in the proof of Lemma 3 (which after normalization with respect to C gives the formulation of Lemma 3 in [1] for $C = 1$).

The remainder of the proof of Lemma 3 analyses μ_0 and the mix $\mu = (1-q)\mu_0 + q\mu_1$ leading to bounds for the expectations (3) and (4) in [1] which only depend on μ_0 and μ_1. Here, q is the probability of having a special data sample ξ (written as d_N in the arguments above) in the batch. In our algorithm $q = s_i/N$. So, we may adopt the statement of Lemma 3 and conclude for the i-th batch computation

$$\alpha_i(\lambda) \leq \frac{s_i^2 \lambda(\lambda+1)}{N(N-s_i)\sigma_i^2} + O\left(\frac{s_i^3\lambda^3}{N^3\sigma_i^3}\right).$$

In order to find an exact expression for the higher order term we look into the details of Lemma 3 of [1]. It computes an upper bound for the binomial tail

$$\sum_{t=3}^{\lambda+1} \binom{\lambda+1}{t} \mathbb{E}_{z\sim\nu_1}[((\nu_0(z)-\nu_1(z))/\nu_1(z))^t], \tag{13}$$

where

$$\begin{aligned} & \mathbb{E}_{z\sim\nu_1}[((\nu_0(z)-\nu_1(z))/\nu_1(z))^t] \\ \leq\ & \frac{(2q)^t(t-1)!!}{2(1-q)^{t-1}\sigma^t} + \frac{q^t}{(1-q)^t\sigma^{2t}} + \frac{(2q)^t\exp((t^2-t)/(2\sigma^2))(\sigma^t(t-1)!!+t^t)}{2(1-q)^{t-1}\sigma^{2t}} \\ =\ & \frac{(2q)^t(t-1)!!(1+\exp((t^2-t)/(2\sigma^2)))}{2(1-q)^{t-1}\sigma^t} + \frac{q^t(1+(1-q)2^t\exp((t^2-t)/(2\sigma^2))t^t)}{2(1-q)^t\sigma^{2t}} \end{aligned} \tag{14}$$

Since $t \geq 3$, we have the coarse upper bounds

$$1 \leq \frac{\exp((t^2-t)/(2\sigma^2))}{\exp((3^2-3)/(2\sigma^2))} \text{ and } 1 \leq \frac{(1-q)2^t\exp((t^2-t)/(2\sigma^2))t^t}{(1-q)2^3\exp((3^2-3)/(2\sigma^2))3^3}.$$

By defining c as 1 plus the maximum of these two bounds,

$$c = 1 + \frac{\max\{1, 1/((1-q)\cdot 216)\}}{\exp(3/\sigma^2)},$$

we have (14) at most

$$\leq \frac{(2q)^t(t-1)!!c\exp((t^2-t)/(2\sigma^2))}{2(1-q)^{t-1}\sigma^t} + \frac{q^t c(1-q)2^t\exp((t^2-t)/(2\sigma^2))t^t}{2(1-q)^t\sigma^{2t}}. \tag{15}$$

Generally (for practical parameter settings as we will find out), $q \leq 1 - 1/216$ which makes $c \leq 2$. In the remainder of this proof, we use $c = 2$ and assume $q \leq 215/216$. In fact, assume in the statement of the lemma that $\sigma = \sigma_i \geq 216/215$ which together with $q = s_i/N \leq r_0/\sigma_i$ and $r_0 < 1$ implies $q \leq 215/216$.

After multiplying (15) with the upper bound for

$$\binom{\lambda+1}{t} \leq \frac{\lambda+1}{\lambda}\frac{\lambda^t}{t!}$$

and noticing that $(t-1)!!/t! \leq 1$ and $t^t/t! \leq e^t$ we get the addition of the following two terms

$$\frac{\lambda+1}{\lambda}\frac{\lambda^t(2q)^t\exp((t^2-t)/(2\sigma^2))}{(1-q)^{t-1}\sigma^t} + \frac{\lambda+1}{\lambda}\frac{\lambda^t q^t(1-q)2^t\exp((t^2-t)/(2\sigma^2))e^t}{(1-q)^t\sigma^{2t}}.$$

This is equal to

$$\begin{aligned} & (1-q)\frac{\lambda+1}{\lambda}\left(\frac{\lambda 2q\exp((t-1)/(2\sigma^2))}{(1-q)\sigma}\right)^t \\ & \quad +(1-q)\frac{\lambda+1}{\lambda}\left(\frac{\lambda q 2\exp(1+(t-1)/(2\sigma^2))}{(1-q)\sigma^2}\right)^t. \end{aligned} \tag{16}$$

We notice that by using $t \leq \lambda + 1$, $\lambda/\sigma^2 \leq \ln(1/(q\sigma))$ (assumption), and $q = s_{i,c}/N_c \leq r_0/\sigma$ we obtain

$$\frac{\lambda 2q \exp((t-1)/(2\sigma^2))}{(1-q)\sigma} \quad \leq \quad \frac{\lambda 2q \exp(\lambda/(2\sigma^2))}{(1-q)\sigma} \leq \frac{2\sqrt{\lambda q}}{(1-q)\sigma} = \frac{2\sqrt{\lambda r_0/\sigma}}{\sigma - r_0} = U_0(\lambda)$$

and

$$\frac{\lambda q 2 \exp(1+(t-1)/(2\sigma^2))}{(1-q)\sigma^2} \quad \leq \quad \frac{\lambda q 2e \exp(\lambda/(2\sigma^2))}{(1-q)\sigma^2} \leq \frac{2e\sqrt{\lambda q}}{(1-q)\sigma^2} = \frac{2e\sqrt{\lambda r_0/\sigma}}{(\sigma - r_0)\sigma} = U_1(\lambda).$$

Together with our assumption on $U_0(\lambda)$ and $U_1(\lambda)$, this means that the binomial tail (13) is upper bounded by the two terms in (16) after substituting $t = 3$, with the two terms multiplied by

$$\sum_{j=0}^{\infty} U_0(\lambda)^j = \frac{1}{1-U_0(\lambda)} \leq \frac{1}{1-u_0} \text{ and } \sum_{j=0}^{\infty} U_1(\lambda)^j = \frac{1}{1-U_1(\lambda)} \leq \frac{1}{1-u_1}$$

respectively. For (13) this yields the upper bound

$$\begin{aligned} & \frac{1}{1-u_0}(1-q)\frac{\lambda+1}{\lambda}\left(\frac{\lambda 2q\exp(1/\sigma^2)}{(1-q)\sigma}\right)^3 + \frac{1}{1-u_1}(1-q)\frac{\lambda+1}{\lambda}\left(\frac{\lambda q 2\exp(1+1/\sigma^2)}{(1-q)\sigma^2}\right)^3 \\ \leq \quad & \left(\frac{1}{1-u_0}2^3\exp(3/\sigma^2) + \frac{1}{1-u_1}\frac{2^3\exp(3+3/\sigma^2)}{\sigma^3}\right) \cdot \frac{\lambda^2(\lambda+1)q^3}{(1-q)^2\sigma^3}. \end{aligned}$$

By the definition of r, we obtain the bound

$$\leq \quad \frac{r}{r_0} \cdot \frac{\lambda^2(1+\lambda)q^3}{(1-q)^2\sigma^3},$$

which finalizes the proof.

B.2.2 Proof of Theorem B.2

The proof Theorem B.2 follows the line of thinking in the proof of Theorem 1 in [1]. Our theorem applies to varying sample/batch sizes and for this reason introduces moments $\hat{S}_j$. Our theorem explicitly defines the constant used in the lower bound of σ – this is important for proving our second (main) theorem in the next subsection.

Theorem B.2 assumes $\sigma = \sigma_i$ for all rounds i with $\sigma \geq 216/215$; constant $r_0 \leq 1/e$ such that $s_i/N \leq r_0/\sigma$; constant

$$r \quad = \quad r_0 \cdot 2^3 \left(\frac{1}{1-u_0} + \frac{1}{1-u_1}\frac{e^3}{\sigma^3}\right)\exp(3/\sigma^2), \tag{17}$$

where

$$u_0 = \frac{2\sqrt{r_0\sigma}}{\sigma - r_0} \text{ and } u_1 = \frac{2e\sqrt{r_0\sigma}}{(\sigma - r_0)\sigma}$$

are both assumed < 1.

For $j = 1, 2, 3$ we define[§§]

$$\hat{S}_j = \frac{1}{T}\sum_{i=0}^{T-1} \frac{s_i^j}{N(N-s_i)^{j-1}} \text{ with } \frac{\hat{S}_1\hat{S}_3}{\hat{S}_2^2} \leq \rho,\ \frac{\hat{S}_1^2}{\hat{S}_2} \leq \hat{\rho}.$$

Based on these constants we define

$$c(x) = \min\left\{\frac{\sqrt{2r\rho x + 1} - 1}{r\rho x}, \frac{2}{\hat{\rho}x}\right\}.$$

Let $\epsilon = c_1 T\hat{S}_1^2$. We want to prove Algorithm 4 is (ϵ, δ)-differentially private if

$$\sigma \geq \frac{2}{\sqrt{c_0}} \frac{\sqrt{\hat{S}_2 T(\epsilon + \ln(1/\delta))}}{\epsilon} \text{ where } c_0 = c(c_1).$$

§§ s_i^j denotes the j-th power $(s_i)^j$.

Proof. For $j = 1, 2, 3$, we define

$$S_j = \sum_{i=0}^{T-1} \frac{s_i^j}{N(N-s_i)^{j-1}\sigma_i^j} \text{ and } S'_j = \frac{1}{T}\sum_{i=0}^{T-1} \frac{s_i^j \sigma_i^j}{N(N-s_i)^{j-1}}.$$

(Notice that $S'_1 \leq r_0$.) Translating Lemma B.3 in this notation yields (we will verify the requirement/assumptions of Lemma B.3 on the fly below)

$$\sum_{i=0}^{T-1} \alpha_i(\lambda) \leq S_2\lambda(\lambda+1) + \frac{r}{r_0} S_3 \lambda^2(\lambda+1).$$

The composition Theorem 2 in [1] shows that our algorithm for client c is (ϵ, δ)-differentially private for

$$\delta \quad \geq \quad \min_\lambda \exp\left(\sum_{i=0}^{T-1} \alpha_i(\lambda) - \lambda\epsilon\right),$$

where T indicates the total number of batch computations and the minimum is over positive integers λ. Similar to their proof we choose λ such that

$$S_2\lambda(\lambda+1) + \frac{r}{r_0} S_3\lambda^2(\lambda+1) - \lambda\epsilon \leq -\lambda\epsilon/2. \tag{18}$$

This implies that we can choose δ as small as $\exp(-\lambda\epsilon/2)$, i.e., if

$$\delta \geq \exp(-\lambda\epsilon/2), \tag{19}$$

then we have (ϵ, δ)-differential privacy. After dividing by the positive integer λ, inequality (18) is equivalent to the inequality

$$S_2(\lambda+1) + \frac{r}{r_0} S_3 \lambda(1+\lambda) \leq \epsilon/2,$$

which is equivalent to

$$(\lambda+1)\left(1 + \frac{r}{r_0}\frac{S_3}{S_2}\lambda\right) \leq \frac{\epsilon}{2S_2}.$$

This is in turn implied by

$$\lambda + 1 \leq c_0 \frac{\epsilon}{2S_2} \tag{20}$$

together with

$$c_0\frac{\epsilon}{2S_2}\left(1 + \frac{r}{r_0}\frac{S_3}{S_2} c_0 \frac{\epsilon}{2S_2}\right) \leq \frac{\epsilon}{2S_2},$$

or equivalently,

$$c_0\left(1 + \frac{r}{2r_0} \cdot c_0 \cdot \frac{S_3}{S_2^2}\epsilon\right) \leq 1. \tag{21}$$

We use

$$\epsilon = c_1 \cdot T\hat{S}_1^2 = c_1 \cdot S_1 S'_1 \tag{22}$$

(for constant $\sigma_i = \sigma$). This translates our requirements (20) and (21) into

$$\lambda + 1 \leq \frac{c_0 c_1}{2}\frac{S_1 S'_1}{S_2} \text{ and} \tag{23}$$

$$c_0\left(1 + \frac{r}{2r_0} \cdot c_0 c_1 \frac{S_1 S_3}{S_2^2} S'_1\right) \leq 1. \tag{24}$$

Since we assume

$$\frac{S_1 S_3}{S_2^2} = \frac{\hat{S}_1 \hat{S}_3}{\hat{S}_2^2} \leq \rho$$

and since we know that $S'_1 \leq r_0$, requirement (24) is implied by

$$c_0(1 + \frac{r\rho}{2} \cdot c_0 c_1) \leq 1,$$

or equivalently

$$c_1 \leq \frac{1-c_0}{\frac{r\rho}{2}c_0^2}. \tag{25}$$

Also notice that for constant $\sigma_i = \sigma$ we have $S'_1 = S_1\sigma^2/T$. Together with

$$\frac{S_1^2}{S_2} = \frac{\hat{S}_1^2}{\hat{S}_2}T \leq \hat{\rho}T$$

we obtain from (23)

$$\lambda + 1 \leq \frac{c_0 c_1}{2}\frac{S_1 S'_1}{S_2} \leq \frac{c_0 c_1}{2}\hat{\rho}\sigma^2. \tag{26}$$

Generally, if

$$c_1 \leq \frac{2}{\hat{\rho}c_0}, \tag{27}$$

then (26) implies $\lambda \leq \sigma^2$: Hence, (a) for our choice of u_0 and u_1 in this theorem, $U_0(\lambda) \leq u_0$ and $U_1(\lambda) \leq u_1$ as defined in Lemma B.3, and (b) the condition $\lambda \leq \sigma_i^2 \ln \frac{N_c}{s_{i,c}\sigma_i}$ is satisfied (by assumption, $\frac{N_c}{s_{i,c}\sigma_i} \geq 1/r_0 \geq e$). This implies that Lemma B.3 is indeed applicable.

For the above reasons we strengthen the requirement on ϵ (conditions (25) and (27) with (22)) to

$$\epsilon \leq \min\left\{\frac{1-c_0}{\frac{r\rho}{2}c_0^2}, \frac{2}{\hat{\rho}c_0}\right\} \cdot S_1 S'_1$$

For constant $\sigma_i = \sigma$, we have

$$S_1 S'_1 = T\hat{S}_1^2,$$

hence, we need

$$\epsilon \leq \min\left\{\frac{1-c_0}{\frac{r\rho}{2}c_0^2}, \frac{2}{\hat{\rho}c_0}\right\} \cdot T\hat{S}_1^2 \tag{28}$$

Summarizing (28), (20), and (19) for some positive integer λ proves (ϵ, δ)-differential privacy.

Condition (19) (i.e., $\exp(-\lambda\epsilon/2) \leq \delta$) is equivalent to

$$\ln(1/\delta) \leq \frac{\lambda\epsilon}{2} \tag{29}$$

If

$$\lambda = \lfloor c_0 \frac{\epsilon}{2S_2} \rfloor - 1 \tag{30}$$

is positive, then it satisfies (20) and we may use this λ in (29). This yields the condition

$$\ln(1/\delta) \leq \left(\lfloor c_0 \frac{\epsilon}{2S_2} \rfloor - 1\right)\frac{\epsilon}{2},$$

which is implied by

$$\ln(1/\delta) \leq \left(c_0 \frac{\epsilon}{2S_2} - 2\right)\frac{\epsilon}{2} = \frac{c_0}{4S_2}\epsilon^2 - \epsilon.$$

For constant $\sigma_i = \sigma$ we have $S_2 = \hat{S}_2 T/\sigma^2$ and the latter inequality is equivalent to

$$\sigma \geq \frac{2}{\sqrt{c_0}} \frac{\sqrt{\hat{S}_2}\sqrt{T(\epsilon + \ln(1/\delta))}}{\epsilon}. \tag{31}$$

Summarizing, if (28), (31), and the lambda value (30) is positive, then this shows (ϵ, δ)-differential privacy.

The condition (30) being positive follows from

$$\frac{4S_2}{c_0} \leq \epsilon.$$

Substituting $S_2 = \hat{S}_2 T/\sigma^2$ yields the equivalent condition

$$\frac{4T\hat{S}_2}{\sigma^2 c_0} \leq \epsilon$$

or

$$\sigma \geq \frac{2}{\sqrt{c_0}} \sqrt{\hat{S}_2} \frac{\sqrt{T\epsilon}}{\epsilon},$$

which is implied by (31). Summarizing, if (28) and (31), then this shows (ϵ, δ)-differential privacy. Notice that (31) corresponds to Theorem 1 in [1] where all s_i are constant implying $\sqrt{\hat{S}_2} = q/\sqrt{1-q}$ in their notation.

We are interested in a slightly different formulation: Given

$$c_1 = \min\left\{\frac{1-c_0}{\frac{r\rho}{2}c_0^2}, \frac{2}{\hat{\rho}c_0}\right\}$$

what is the maximum possible c_0 (which minimizes σ implying more fast convergence to an accurate solution). We need to satisfy $c_0 \leq 2/(\hat{\rho}c_1)$ and

$$\frac{r\rho}{2}c_1 c_0^2 + c_0 - 1 \leq 0,$$

that is,

$$(c_0 + 1/(r\rho c_1))^2 \leq 1/\left(\frac{r\rho}{2}c_1\right) + 1/(r\rho c_1)^2,$$

or

$$c_0 \leq \sqrt{1/\left(\frac{r\rho}{2}c_1\right) + 1/\left(r\rho c_1\right)^2} - 1/(r\rho c_1) = \frac{\sqrt{2r\rho c_1 + 1} - 1}{r\rho c_1}.$$

We have

$$c_0 = \min\left\{\frac{\sqrt{2r\rho c_1 + 1} - 1}{r\rho c_1}, 2/(\hat{\rho}c_1)\right\} = c(c_1).$$

This finishes the proof.

B.3 A Simplified Characterization

So far, we have generalized Theorem B.1 in Appendix B in a non-trivial way by analysing increasing sample size sequences, by making explicit the higher order error term in [1], and by providing a precise functional relationship among the constants c_1 and c_2 in Theorem B.1. The resulting Theorem B.2 is still hard to interpret. The next theorem is a consequence of Theorem B.2 and brings us the interpretation we look for.

Theorem B.4. *For sample size sequence $\{s_i\}_{i=0}^{T-1}$ the total number of local SGD iterations is equal to $K = \sum_{i=0}^{T-1} s_i$. We define the mean $\bar{s}$ and maximum s_{max} and their quotient θ as*

$$\bar{s} = \frac{1}{T}\sum_{i=0}^{T-1} s_i = \frac{K}{T}, \;\; s_{max} = \max\{s_0, \ldots, s_{T-1}\}, \;\; \textit{and} \;\; \theta = \frac{s_{max}}{\bar{s}}.$$

We define

$$h(x) = \left(\sqrt{1 + (e/x)^2} - e/x\right)^2, \;\; g(x) = \min\left\{\frac{1}{ex}, h(x)\right\},$$

and denote by γ the smallest solution satisfying

$$\gamma \geq \frac{2}{1-\bar{\alpha}} + \frac{2^4 \cdot \bar{\alpha}}{1-\bar{\alpha}}\left(\frac{\sigma}{(1-\sqrt{\bar{\alpha}})^2} + \frac{1}{\sigma(1-\bar{\alpha}) - 2e\sqrt{\bar{\alpha}}}\frac{e^3}{\sigma}\right)e^{3/\sigma^2} \textit{ with } \bar{\alpha} = \frac{\epsilon N}{\gamma K}.$$

If the following requirements are satisfied:

$$\bar{s} \leq \frac{g\left(\sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}\right)}{\theta} \cdot N, \tag{32}$$

$$\epsilon \leq \gamma h(\sigma) \cdot \frac{K}{N}, \tag{33}$$

$$\epsilon \geq \gamma\theta^2 \cdot \frac{K}{N} \cdot \frac{\bar{s}}{N}, \textit{ and} \tag{34}$$

$$\sigma \geq \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}, \tag{35}$$

then Algorithm 4 is (ϵ, δ)-differentially private.

Its proof follows from analysing the requirements stated in Theorem B.2. We will focus on the case where $c(x) = \frac{2}{\hat{\rho}x}$, which turns out to lead to practical parameter settings as discussed in the main body of the paper.

Requirement on r – (38): In Theorem B.2 we use

$$r = r_0 \cdot 2^3 \cdot \left(\frac{1}{1-u_0} + \frac{1}{1-u_1} \frac{e^3}{\sigma^3} \right) e^{3/\sigma^2}$$

with

$$u_0 = \frac{2\sqrt{r_0 \sigma}}{\sigma - r_0} \text{ and } u_1 = \frac{2e\sqrt{r_0\sigma}}{(\sigma - r_0)\sigma},$$

where r_0 is such that it satisfies

$$r_0 \leq 1/e, \; u_0 < 1, \text{ and } u_1 < 1. \tag{36}$$

In our application of Theorem B.2 we substitute $r_0 = \alpha\sigma$. This translates the requirements of (36) into

$$\alpha \leq \frac{1}{e\sigma}, \; \alpha < 1, \text{ and } \sigma > \frac{2e\sqrt{\alpha}}{1-\alpha}. \tag{37}$$

As we will see in our derivation, we will require another lower bound (42) on σ. We will use (42) together with

$$\alpha \leq \frac{1}{e\sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}}, \alpha < 1, \text{ and } \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon} > \frac{2e\sqrt{\alpha}}{1-\alpha}$$

to imply the needed requirement (37). These new bounds on α are in turn equivalent to

$$\alpha \leq g(\epsilon, \delta) \text{ where} \tag{38}$$

$$g(\epsilon,\delta) = \min\left\{ \frac{\sqrt{\epsilon}}{e\sqrt{2(\epsilon+\ln(1/\delta))}}, \left(\sqrt{1 + \frac{e^2\epsilon}{2(\epsilon+\ln(1/\delta))}} - \frac{e\sqrt{\epsilon}}{\sqrt{2(\epsilon+\ln(1/\delta))/\epsilon}} \right)^2 \right\}$$

(notice that this implies $\alpha < 1$).

Substituting $r_0 = \alpha\sigma$ in the formula for r yields the expression

$$r \;=\; 2^3 \cdot \left(\frac{\sigma}{(1-\sqrt{\alpha})^2} + \frac{1}{\sigma(1-\alpha) - 2e\sqrt{\alpha}} \frac{e^3}{\sigma} \right) \cdot e^{3/\sigma^2}(1-\alpha)\alpha. \tag{39}$$

Requirement on s_i/N – (40): In Theorem B.2 we also require $s_i/N \leq r_0/\sigma$ which translates into

$$s_i/N \leq \alpha. \tag{40}$$

Requirement on σ – (42) and (43): In Theorem B.2 we restrict ourselves to the case where function $c(x)$ attains the minimum $c(x) = 2/(\hat{\rho}x)$. This happens when

$$\frac{\sqrt{2r\rho x + 1} - 1}{r\rho x} \geq \frac{2}{\hat{\rho}x}.$$

This is equivalent to

$$x \geq 2r\frac{\rho}{\hat{\rho}^2} + \frac{2}{\hat{\rho}}. \tag{41}$$

Notice that in the lower bound for σ in Theorem B.2 we use $c_0 = c(x)$ for $x = c_1$, where c_1 is implicitly defined by

$$\epsilon = c_1 T \hat{S}_1^2$$

or equivalently

$$c_1 = \frac{\epsilon}{T\hat{S}_1^2}.$$

To minimize ϵ, we want to minimize $c_1 = x$. That is, we want $c_1 = x$ to match the lower bound (41). This lower bound is smallest if we choose the smallest possible ρ (due to the linear dependency of the lower bound on ρ). Given the constraint on ρ this means we choose

$$\rho = \frac{\hat{S}_1\hat{S}_3}{\hat{S}_2^2}.$$

For $c_1 = x$ satisfying (41) we have

$$c_0 = c(c_1) = \frac{2}{\hat{\rho}x}.$$

Substituting this in the lower bound for σ attains

$$\sigma \geq \frac{2}{\sqrt{c(c_1)}} \frac{\sqrt{\hat{S}_2 T(\epsilon + \ln(1/\delta))}}{\epsilon} = \sqrt{\frac{\hat{\rho}\hat{S}_2}{\hat{S}_1^2}} \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}.$$

In order to yield the best test accuracy we want to choose the smallest possible σ. Hence, we want to minimize the lower bound for σ and therefore choose the smallest $\hat{\rho}$ given its constraints, i.e.,

$$\hat{\rho} = \frac{\hat{S}_1^2}{\hat{S}_2}.$$

This gives

$$\sigma \geq \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}. \tag{42}$$

Notice that this lower bound implies $\sigma \geq 216/215$ and for this reason we do not state this as an extra requirement.

Our expressions for ρ, $\hat{\rho}$, and c_1 with $x = c_1$ shows that lower bound (41) holds if and only if

$$\epsilon \geq \left(2r\frac{\hat{S}_3}{\hat{S}_1} + 2\hat{S}_2\right) T. \tag{43}$$

Requirement implying (43): The definition of moments $\hat{S}_j$ imply

$$\hat{S}_1 = \frac{K}{TN}$$

and, since $s_i/N \leq \alpha < 1$,

$$\hat{S}_j \leq \alpha^j/(1-\alpha)^{j-1}.$$

Lower bound (43) on ϵ is therefore implied by

$$\epsilon \geq 2r\frac{\alpha^3}{(1-\alpha)^2}\frac{T^2 N}{K} + 2\frac{\alpha^2}{1-\alpha}T. \tag{44}$$

We substitute

$$T = \beta\frac{K}{N} \tag{45}$$

in (44) which yields the requirement

$$\epsilon\frac{N}{K} \geq \frac{2r}{\alpha(1-\alpha)^2}(\alpha^2\beta)^2 + \frac{2}{1-\alpha}(\alpha^2\beta). \tag{46}$$

This inequality is implied by the combination of the following two inequalities:

$$\alpha^2\beta \leq \frac{\epsilon N}{\gamma K} \tag{47}$$

and

$$1 \geq \frac{2r}{\alpha(1-\alpha)^2}\frac{\epsilon N}{K}\frac{1}{\gamma^2} + \frac{2}{1-\alpha}\frac{1}{\gamma}. \tag{48}$$

Inequality (48) is equivalent to

$$\gamma \geq \frac{2r}{\alpha(1-\alpha)^2}\frac{\epsilon N}{\gamma K} + \frac{2}{1-\alpha}. \tag{49}$$

This implies

$$\gamma \geq \frac{2}{1-\alpha} \geq 2.$$

Also notice that

$$\frac{1}{\beta} = \frac{K}{TN} = \hat{S}_1 \leq \alpha$$

from which we obtain

$$1 \leq \alpha\beta.$$

Let us define

$$\bar{\alpha} = \frac{\epsilon N}{\gamma K}. \tag{50}$$

Inequalities $\gamma \geq 2$ and $1 \leq \alpha\beta$ together with (47) and the definition of $\bar{\alpha}$ imply

$$\alpha \leq \alpha^2\beta \leq \frac{\epsilon N}{\gamma K} = \bar{\alpha} \leq \frac{\epsilon N}{2K}. \tag{51}$$

We will require

$$\bar{\alpha} < 1 \tag{52}$$

and also $\sigma(1-\bar{\alpha}) - 2e\sqrt{\bar{\alpha}} > 0$ i.e,

$$\sigma > \frac{2e\sqrt{\bar{\alpha}}}{1-\bar{\alpha}}. \tag{53}$$

Bounds (52) and (53) are equivalent to

$$\bar{\alpha} \leq h(\sigma) \text{ where } h(\sigma) = \left(\sqrt{1+(e/\sigma)^2} - e/\sigma\right)^2. \tag{54}$$

With condition (54) in place we may derive the upper bound

$$\begin{aligned} & \frac{2r}{\alpha(1-\alpha)^2} \\ = \quad & \frac{2^4}{1-\alpha}\left(\frac{\sigma}{(1-\sqrt{\alpha})^2} + \frac{1}{\sigma(1-\alpha)-2e\sqrt{\alpha}}\frac{e^3}{\sigma}\right)e^{3/\sigma^2} \\ \leq \quad & \frac{2^4}{1-\bar{\alpha}}\left(\frac{\sigma}{(1-\sqrt{\bar{\alpha}})^2} + \frac{1}{\sigma(1-\bar{\alpha})-2e\sqrt{\bar{\alpha}}}\frac{e^3}{\sigma}\right)e^{3/\sigma^2} \end{aligned}$$

because all denominators are decreasing functions in α and remain positive for $\alpha \leq \bar{\alpha}$. Similarly,

$$\frac{2}{1-\alpha} \leq \frac{2}{1-\bar{\alpha}}.$$

These two upper bounds combined with (50) show that (49) is implied by choosing

$$\gamma = \gamma(\sigma, \epsilon N/K),$$

where $\gamma(\sigma, \epsilon N/K)$ is defined as the smallest solution of γ satisfying

$$\begin{aligned} \gamma \quad & \geq \frac{2}{1-\bar{\alpha}} + \\ & \frac{2^4 \cdot \bar{\alpha}}{1-\bar{\alpha}}\left(\frac{\sigma}{(1-\sqrt{\bar{\alpha}})^2} + \frac{1}{\sigma(1-\bar{\alpha})-2e\sqrt{\bar{\alpha}}}\frac{e^3}{\sigma}\right)e^{3/\sigma^2}, \end{aligned} \tag{55}$$

where $\bar{\alpha} = (\epsilon N/K)/\gamma$. The smallest solution γ will meet (55) with equality. For this reason the minimal solution γ will be at most the right hand side of (55) where γ is replaced by its lower bound 2; this is allowed because this increases $\bar{\alpha}$ to the upper bound in (51) and we know that the right hand side of (55) increases in $\bar{\alpha}$ up to the upper bound in (51) if the upper bound satisfies

$$\frac{\epsilon N}{2K} \leq h(\sigma).$$

This makes requirement (54) slightly stronger – but in practice this stronger requirement is already satisfied because K is several epochs of N iterations making $\frac{\epsilon N}{2K} \ll 1$ while $\sigma \gg 1$ for small ϵ implying that $h(\sigma)$ is close to 1.

Notice that $\gamma = 2 + O(\bar{\alpha})$, hence, for small $\bar{\alpha}$ we have $\gamma \approx 2$. A more precise asymptotic analysis reveals

$$\gamma = 2 + (2 + 2^4 \cdot \left(\sigma + \frac{e^3}{\sigma^2}\right)e^{3/\sigma^2})\bar{\alpha} + O(\bar{\alpha}^{3/2}).$$

Relatively large $\bar{\alpha}$ closer to 1 will yield $\gamma \gg 2$.

Summarizing

$$\{(45), (47), (50), (54), (55)\} \Rightarrow (43).$$

Combining all requirements – resulting in (57), (58), and (42), or equivalently (60), (61), and (42): The combination of requirements (45) and (47) is equivalent to

$$\alpha \leq \sqrt{\frac{\epsilon}{\gamma T}} \tag{56}$$

(notice that T and β are not involved in any of the other requirements including those discussed earlier in this discussion, hence, we can discard (45) and substitute this in (47)). The combination of (50), (54), and (55) is equivalent to

$$\frac{\epsilon N}{\gamma K} \leq h(\sigma) \text{ with } \gamma = \gamma\left(\sigma, \frac{\epsilon N}{K}\right) \tag{57}$$

(for the definition of $h(.)$ see (54) and for $\gamma(.,.)$ see (55)).

We may now combine (56), (38), and (40) into a single requirement

$$s_i/N \leq \min\left\{g(\epsilon, \delta), \sqrt{\frac{\epsilon}{\gamma T}}\right\} \tag{58}$$

(for the definition of $g(.,.)$ see (38)). This shows that (57), (58), and (42) (we remind the reader that the last condition is the lower bound on $\sigma \geq \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}$) implies (ϵ, δ)-DP by Theorem B.2.

Let us rewrite these conditions. We introduce the mean $\bar{s}$ of all s_i defined by

$$\bar{s} = \frac{1}{T}\sum_{i=0}^{T-1} s_i = \frac{K}{T}$$

and we introduce the maximum s_{max} of all s_i defined by

$$s_{max} = \max\{s_0, \ldots, s_{T-1}\}.$$

We define θ as the fraction

$$\theta = \frac{s_{max}}{\bar{s}}. \tag{59}$$

This notation allows us to rewrite

$$s_i/N \leq \sqrt{\frac{\epsilon}{\gamma T}}$$

from (58) as

$$\gamma \frac{K}{N}\frac{\bar{s}}{N}\theta^2 \leq \epsilon.$$

From this we obtain that the requirements (57) and (58) are equivalent to

$$\gamma\left(\sigma, \frac{\epsilon N}{K}\right) \cdot \frac{K}{N}\frac{\bar{s}}{N}\theta^2 \leq \epsilon \leq \gamma\left(\sigma, \frac{\epsilon N}{K}\right) \cdot h(\sigma)\frac{K}{N} \tag{60}$$

and

$$\theta \bar{s} \leq g(\epsilon, \delta) N. \tag{61}$$

This alternative description shows that (60), (61), and (42) with definitions for $h(.)$, $\gamma(.,.)$, $g(.,.)$, and θ in (54), (55), (38), and (59) implies (ϵ, δ)-DP. This proves Theorem B.4 (after a slight rewrite of the definitions of functions $h(.)$ and $g(.,.)$).

B.4 Proof of the Main Theorem

Theorem B.4 can already be used to a-priori set hyperparameters given DP and accuracy targets. Still, as discussed below, by making slight approximations (leading to slightly stronger constraints) we obtain the easy to interpret Theorem 3.1 discussed in Section 3.

We set σ as large as possible with respect to the accuracy we wish to have. Given this σ we want to max out on our privacy budget. That is, we satisfy (35) with equality,

$$\sigma = \sqrt{\frac{2(\epsilon + \ln(1/\delta))}{\epsilon}}. \tag{62}$$

We discuss (62) with constraints (32), (33), and (34) below:

Replacing (32) and (33): In practice, we need a sufficiently strong DP guarantee, hence, $\delta \leq 1/N$ and ϵ is small enough, typically ≤ 0.5. This means that we will stretch σ to at least $\sqrt{2+4\ln N}$. A local data set of size $N = 10000$ requires $\sigma \geq 6.23$; a local data set of size $N = 100000$ requires $\sigma \geq 6.93$. For such $\sigma \geq 6$ we have $h(\sigma) \geq h(6) = 0.42$ (since $h(\sigma)$ is increasing in σ). (For reference, $h(10) = 0.58$, and for $\sigma \gg 1$ we have $h(\sigma) \approx 1$.) From (62) we infer that $g(\sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}) = g(\sigma) = \min\{1/(e\sigma), h(\sigma)\}$. One can verify that $h(\sigma) - 1/(e\sigma)$ is positive and increasing for $\sigma \geq 2.5$, hence, $g(\sigma) = 1/(e\sigma)$ for $\sigma \geq 6$. This reduces requirement (32) to $\bar{s} \leq N/(e\sigma\theta)$ and requirement (33) to $\epsilon \leq 0.42 \cdot \gamma K/N$. Notice that (34) in combination with $\epsilon \leq \frac{\gamma\theta}{e\sigma} K/N$ implies condition $\bar{s} \leq N/(e\sigma\theta)$. This implies that (32) and (33) are satisfied for $\epsilon \leq \min\{0.42 \cdot \gamma, \gamma\theta/(e\sigma)\} \cdot \frac{K}{N}$ or, equivalently, $K \geq \epsilon \cdot \max\{2.4/\gamma, e\sigma/(\gamma\theta)\}$ epochs of size N. If $\theta \leq 6.85$, then $\sigma \geq 6 \geq 0.88 \cdot \theta = 2.4 \cdot \theta/e$, hence, $\max\{2.4/\gamma, e\sigma/(\gamma\theta)\} = e\sigma/(\gamma\theta)$ and this reduces the condition on K to

$$K \geq \epsilon\sigma \cdot \frac{e}{\gamma\theta} = \sqrt{2\epsilon(\epsilon + \ln(1/\delta))} \cdot e/(\gamma\theta) \text{ epochs of size } N,$$

where the equality follows from (62). In practical settings, K consists of multiple (think 50 or 100s of) epochs (of size N) computation and this is generally true. We conclude that (32) and (33) are automatically satisfied by (34) for general practical settings with $\delta \leq 1/N$, ϵ typically smaller than 0.5, $N \geq 10000$, $\theta \leq 6.85$, and $K \geq \sqrt{2\epsilon(\epsilon + \ln(1/\delta))} \cdot e/(\gamma\theta)$ epochs.

Remaining constraint (34): By using (62), (34) can be equivalently recast as an upper bound on σ,

$$\sigma \leq \sqrt{\frac{2(\epsilon + \ln(1/\delta))}{\gamma\theta^2 \cdot (K/N) \cdot (\bar{s}/N)}}.$$

Here, γ is a function of σ because γ depends on ϵ in $\bar{\alpha}$ which is a function of σ through (62). However, the definition of γ shows that for small ϵ, γ is close to 2 and this gives $\sqrt{\ln(1/\delta)/(\theta^2 \cdot (K/N) \cdot (\bar{s}/N))}$ as a good approximation of the upper bound. Substituting $\bar{s} = K/T$ yields

$$\sigma \leq \frac{N\sqrt{T}}{K} \sqrt{\frac{2(\epsilon + \ln(1/\delta))}{\gamma\theta^2}}. \quad (63)$$

For $\gamma \approx 2$ and $\theta = 1$ (constant sample size), this upper bound compares to taking $c_2 z \approx \sqrt{2}$ in (11); we go beyond the analysis presented in [1] in a non-trivial way.

If $N\sqrt{T}/K$ is large enough, larger than the relatively small $\sigma\sqrt{\theta^2(\gamma/2)/(\epsilon + \ln(1/\delta))}$, then upper bound (63) is satisfied. That is, for given K and N, we need T to be large enough, or equivalently the mean sample/mini-batch size $\bar{s} = K/T$ small enough. Squaring both sides of (63) and moving terms yields the equivalent lower bound

$$T \geq \frac{\gamma}{2} \frac{\sigma^2\theta^2}{\epsilon + \ln(1/\delta)} \cdot (K/N)^2,$$

which after substituting (62) gives

$$T \geq \frac{\gamma\theta^2}{\epsilon} \cdot (K/N)^2.$$

In other words T is at least a factor $\gamma\theta^2/\epsilon$ larger than the square of the overall amount of local SGD computations measured in epochs (of size N). Notice that we have a lower bound on T rather than an upper bound as in (12) from the theorem presented in [1].

C Tight Analysis using Gaussian DP

[6] explain an elegant alternative DP formulation based on hypothesis testing. From the attacker's perspective, it is natural to formulate the problem of distinguishing two neighboring data sets d and d' based on the output of a DP mechanism $\mathcal{M}$ as a hypothesis testing problem:

$$H_0: \text{ the underlying data set is } d \quad \text{versus} \quad H_1: \text{ the underlying data set is } d'.$$

We define the Type I and Type II errors by

$$\alpha_\phi = \mathbf{E}_{o\sim\mathcal{M}(d)}[\phi(o)] \text{ and } \beta_\phi = 1 - \mathbf{E}_{o\sim\mathcal{M}(d')}[\phi(o)],$$

where ϕ in $[0,1]$ denotes the rejection rule which takes the output of the DP mechanism as input. We flip a coin and reject the null hypothesis with probability ϕ. The optimal trade-off between Type I and Type II errors is given by the trade-off function

$$T(\mathcal{M}(d), \mathcal{M}(d'))(\alpha) = \inf_{\phi}\{\beta_\phi \,:\, \alpha_\phi \leq \alpha\},$$

for $\alpha \in [0,1]$, where the infimum is taken over all measurable rejection rules ϕ. If the two hypothesis are fully indistinguishable, then this leads to the trade-off function $1-\alpha$. We say a function $f \in [0,1] \to [0,1]$ is a trade-off function if and only if it is convex, continuous, non-increasing, and $f(x) \leq 1-x$ for $x \in [0,1]$. We define a mechanism $\mathcal{M}$ to be f-DP if

$$T(\mathcal{M}(d), \mathcal{M}(d')) \geq f$$

for all neighboring d and d'. Proposition 2.5 in [6] is an adaptation of a result in [33] and states that a mechanism is (ϵ, δ)-DP if and only if the mechanism is $f_{\epsilon,\delta}$-DP, where

$$f_{\epsilon,\delta}(\alpha) = \min\{0, 1-\delta-e^{\epsilon}\alpha, (1-\delta-\alpha)e^{-\epsilon}\}.$$

We see that f-DP has the (ϵ, δ)-DP formulation as a special case. It turns out that the DP-SGD algorithm can be tightly analysed by using f-DP.

Gaussian DP: In order to proceed [6] first defines Gaussian DP as another special case of f-DP as follows: We define the trade-off function

$$G_\mu(\alpha) = T(\mathcal{N}(0,1), \mathcal{N}(\mu,1))(\alpha) = \Phi(\Phi^{-1}(1-\alpha)-\mu),$$

where Φ is the standard normal cumulative distribution of $\mathcal{N}(0,1)$. We define a mechanism to be μ-Gaussian DP if it is G_μ-DP. Corollary 2.13 in [6] shows that a mechanism is μ-Gaussian DP if and only if it is $(\epsilon, \delta(\epsilon))$-DP for all $\epsilon \geq 0$, where

$$\delta(\epsilon) = \Phi(-\frac{\epsilon}{\mu} + \frac{\mu}{2}) - e^{\epsilon}\Phi(-\frac{\epsilon}{\mu} - \frac{\mu}{2}). \tag{64}$$

Subsampling: Besides implementing Gaussian noise, DP-SGD also uses sub-sampling: For a data set d of N samples, $\texttt{Sample}_s(d)$ selects a subset of size s from d uniformly at random. We define convex combinations

$$f_p(\alpha) = pf(\alpha) + (1-p)(1-\alpha)$$

with corresponding p-sampling operator

$$C_p(f) = \min\{f_p, f_p^{-1}\}^{**},$$

where the conjugate g^* of a function g is defined as

$$g^*(y) = \sup_x\{yx - g(x)\}.$$

Theorem 4.2 in [6] shows that if a mechanism $\mathcal{M}$ on data sets of size N is f-DP, then the subsampled mechanism $\mathcal{M} \circ \texttt{Sample}_s$ is $C_{s/N}(f)$-DP.

Composition: The tensor product $f \otimes g$ for trade-off functions $f = T(P,Q)$ and $g = T(P',Q')$ is well-defined by

$$f \otimes g = T(P \times P', Q \times Q').$$

Let $y_i \leftarrow \mathcal{M}_i(\texttt{aux}, d)$ with $\texttt{aux} = (y_1, \ldots, y_{i-1})$. Theorem 3.2 in [6] shows that if $\mathcal{M}_i(\texttt{aux}, .)$ is f-DP for all $\texttt{aux}$, then the composed mechanism $\mathcal{M}$, which applies $\mathcal{M}_i$ in sequential order from $i=1$ to $i=T$, is $f^{\otimes T}$-DP.

Tight Analysis DP-SGD: We are now able to formulate the differential privacy guarantee of DP-SGD since it is a composition of subsampled Gaussian DP mechanisms. Theorem 5.1 in [6] states that DP-SGD in Algorithm 1 is[¶¶]

$$C_{s/N}(G_{(\sigma/2)^{-1}})^{\otimes T}\text{-DP}.$$

Since each of the theorems and results from [6] enumerated above are exact, we have a tight analysis.

Our Goal: We want to understand the behavior of the DP guarantee in terms of s, N, T, and σ. Our goal is to have an easy interpretation of the DP guarantee so that we can select "good" parameters s, N, T, and σ a-priori; good in

[¶¶]Their DP-SGD algorithm uses noise $\mathcal{N}(0, C^2(2\sigma)^2\mathbf{I})$ compared to $\mathcal{N}(0, C^2\sigma^2\mathbf{I})$ in our version of the DP-SGD algorithm. This is related to the earlier footnote on probabilistic versus constant sample sizes. Here, we use $\sigma/2$ in the f-DP framework in order to compare with our DP-SGD parameter setting.

terms of achieving at least our target accuracy without depleting our privacy budget. If we know how the differential privacy budget is being depleted over DP-SGD iterations, then we can optimize parameter settings in order to attain best performance, that is, best accuracy of the final global model (the most important target when we work with machine learning modelling). According to our best knowledge, all the current-state-of-the art privacy accountants do not allow us to achieve this goal because they are only privacy loss accountants and do not offer ahead-planning. It is not sufficient to only rely on a differential privacy accountant (see e.g., [38] as follow-up work of [6]) for a client to understand when to stop helping the server to learn a global model.

When talking about accuracy, we mean how much loss in prediction/test accuracy is sacrificed by fixing a σ (and clipping constant C). Our theory maps σ directly to an (ϵ, δ)-DP guarantee independent of the number of rounds T. This allows use to characterize the trade-off between accuracy and privacy budget. All the current-state-of-the art privacy loss frameworks do not offer this property.

We notice that [6] makes an effort to interpret the $C_{s/N}(G_{(\sigma/2)^{-1}})^{\otimes T}$-DP guarantee. Their Corollary 5.6 provides a precise expression based on integrals, themselves again depending on $p = s/N$ and $\mu = (\sigma/2)^{-1}$ in our notation. This still does not lead to the intuition we seek as we cannot extract how to select parameters σ, s and T given a data set of size N, given a privacy budget, and given a utility that we wish to achieve. We further explain this point in next paragraphs.

In what follows, we seek a relationship between σ, s, T, ϵ, δ, and N for Gaussian DP based on Corollary 5.4 in [6]. Corollary 5.4 in [6] provides an **asymptotic analysis** which is a step forward to the kind of easy to understand interpretation we seek for: It states that if both $T \to \infty$ and $N \to \infty$ such that $s\sqrt{T}/N \to c$ for some constant $c > 0$ (and where s is a function of N that may tend to ∞ as well), then the DP-SGD algorithm (in this paper with the factor 2 differing σ) is μ-Gaussian DP for

$$\mu = (c/2) \cdot \tau^{-1} \text{ with } \tau^{-1} = \sqrt{8} \cdot \sqrt{e^{(\sigma/2)^{-2}} \Phi(3(\sigma/2)^{-1}/2) + 3\Phi(-(\sigma/2)^{-1}/2) - 2}. \tag{65}$$

In Section C.1 we show that $\tau^{-1} = \sigma^{-1} + O(\sigma^{-2})$ and we show that for $\mu \le \epsilon \le 1$, μ-Gaussian DP translates to the DP-SGD algorithm being (ϵ, δ)-DP for $\delta \ll \epsilon \ll 1$ if

$$\tau \quad \approx \quad \frac{(c/2)\sqrt{2(\ln(1/\delta) + \ln(\epsilon) - O(\ln\ln(1/\delta)))}}{\epsilon} \text{ with } s\sqrt{T}/N \to c.$$

We see a similar $s\sqrt{T}/N$ dependency in Theorem B.1 by [1]. The difference is that Theorem B.1 holds in a **non-asymptotic** setting. That is, T and N do not need to tend to ∞ whereas the expression above does require taking such a limit. Of course, one can analyse the convergence rate of achieving the limit μ given T and N tending to infinity. When doing such an analysis one may find expressions of Gaussian DP guarantees as a function of T and N that hold for all concrete values of T and N. This may lead to results that strengthen our Theorem B.1 (we leave this as an open problem). It is clear that the above asymptotic result is still insufficient for our purpose: How do we a-priori select concrete parameters σ, s, and T given concrete parameters for N, a given privacy budget and utility that we wish to achieve?

In this paper we decided to generalize the proof method of Theorem B.1 rather than working with the complex integrals that provide the exact characterization of f-DP for the DP-SGD algorithm as stated above. This approach allows us to obtain the non-asymptotic result of Theorem 3.1 which shows into large extent the independence of T, which is not immediately understood from Theorem B.1 and the corollary discussed above. Section C.2 shows a first result on the tightness of our Theorem 3.1. The advantage of our result is that it is easy to interpret and we do not need to fully rely on an accountant method to keep track of spent privacy budget while participating in learning a global model based on local data.

C.1 Translation to (ϵ, δ)-DP

We first observe that by using $e^x = 1 + x + O(x^2)$ and $\Phi(x) = \frac{1}{2} + \frac{e^{-x^2/2}}{\sqrt{2\pi}}(x + O(x^3)) = \frac{1}{2} + \frac{x}{\sqrt{2\pi}} + O(x^3)$, a first order approximation of τ^{-1} is equal to $\sigma^{-1} + O(\sigma^{-2})$ (hence, $\tau^{-1} \approx \sigma^{-1}$ for large σ).

For $x \ge 0$, we have the approximation

$$\Phi(-x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}} \left(\frac{1}{x} - \frac{1}{x^3} + O(\frac{1}{x^5}) \right).$$

Let $y \leq x$. Together with $-(x-y)^2/2 = 2xy - (x+y)^2/2$ we derive

$$\begin{aligned}
\Phi(-x+y) - e^{2xy}\Phi(-x-y) &= \frac{e^{-(x-y)^2/2}}{\sqrt{2\pi}}\left(\frac{1}{x-y} - \frac{1}{(x-y)^3} + O(\frac{1}{(x-y)^5}\right) \\
&\quad - \frac{e^{-(x-y)^2/2}}{\sqrt{2\pi}}\left(\frac{1}{x+y} - \frac{1}{(x+y)^3} + O(\frac{1}{(x+y)^5})\right) \\
&= \frac{e^{-(x-y)^2/2}}{\sqrt{2\pi}}\left(\frac{2y}{x^2-y^2} - \frac{6yx^2+2y^3}{(x^2-y^2)^3} + O(\frac{1}{x^5})\right) \\
&= \frac{e^{-x^2(1-y/x)^2/2}}{\sqrt{2\pi}}\left(\frac{2y}{x^2(1-(y/x)^2)} + O(\frac{y}{x^4} + \frac{1}{x^5})\right)
\end{aligned}$$

If we assume

$$\mu \leq \epsilon \leq 1,$$

then $(\mu/2)/(\epsilon/\mu) = \mu^2/(2\epsilon) \leq \epsilon/2 \leq 1$, $\epsilon/\mu \geq 1$, and $\epsilon \leq 1$. We can use the above formulas and approximate (64) as follows:

$$\begin{aligned}
\delta &= \Phi(-\frac{\epsilon}{\mu} + \frac{\mu}{2}) - e^{\epsilon}\Phi(-\frac{\epsilon}{\mu} - \frac{\mu}{2}) \\
&= \frac{e^{-(\epsilon/\mu)^2(1-\mu^2/(2\epsilon))^2/2}}{\sqrt{2\pi}}\left(\frac{\mu^3}{\epsilon^2(1-(\mu^2/(2\epsilon))^2)} + O(\frac{\mu^5}{\epsilon^5})\right).
\end{aligned}$$

This gives

$$1/\mu = \frac{\sqrt{2(\ln(1/\delta) + \ln(\frac{\mu^3}{\sqrt{2\pi}(\epsilon^2-\mu^4/4)} + O((\frac{\mu}{\epsilon})^5)))}}{\epsilon - \mu^2/2},$$

hence,

$$\tau = \frac{(c/2)\sqrt{2(\ln(1/\delta) + \ln(\frac{\mu^3}{\sqrt{2\pi}(\epsilon^2-\mu^4/4)} + O((\frac{\mu}{\epsilon})^5)))}}{\epsilon - \mu^2/2} \text{ with } s\sqrt{T}/N \to c \text{ and } \tau^{-1} = \sigma^{-1} + O(\sigma^{-2}).$$

For small ϵ, we can approximate τ as

$$\tau \approx \frac{(c/2)\sqrt{2(\ln(1/\delta) + \ln(\frac{\epsilon}{\sqrt{2\pi}}(\frac{\mu}{\epsilon})^3 + O((\frac{\mu}{\epsilon})^5)))}}{\epsilon}.$$

Since $(c/2)\tau^{-1} = \mu \leq \epsilon$, we may write $\mu = \epsilon/b$ for some $b \geq 1$. Notice that $(c/2)/\epsilon = \tau/b$. This leads to the approximation

$$b \approx \sqrt{2(\ln(1/\delta) + \ln(\frac{\epsilon}{\sqrt{2\pi}}\frac{1}{b^3} + O(\frac{1}{b^5})))}.$$

By default $\delta = 1/N$ and, see Section 3.5, $N = \Omega(1/\epsilon)$, that is, $\epsilon = \Omega(1/N)$ (also notice that good accuracy can only be achieved for σ small enough, that is, ϵ is generally orders of magnitude larger than $1/N$). So, we may assume $\epsilon \gg \delta$. Then substituting $\sqrt{2\ln(1/\delta)}$ for b at the right hand side yields

$$b \approx \sqrt{2(\ln(1/\delta) + \ln(\epsilon) - O(\ln\ln(1/\delta)))}.$$

Substituting back in the expression for τ proves for $\delta \ll \epsilon \ll 1$,

$$\tau \approx \frac{(c/2)\sqrt{2(\ln(1/\delta) + \ln(\epsilon) - O(\ln\ln(1/\delta)))}}{\epsilon} \text{ with } s\sqrt{T}/N \to c \text{ and } \tau^{-1} = \sigma^{-1} + O(\sigma^{-2}).$$

C.2 Asymptotic Tightness of Theorem 3.1

We consider the special case where T meets its lower bound: $T = \frac{\gamma\theta^2}{\epsilon} \cdot k^2$ (notice that our experiments and understanding show that this is a good setting for best accuracy). Let

$$c = \sqrt{\frac{\epsilon}{\gamma\theta^2}} = \frac{k}{\sqrt{T}} = \frac{K}{\sqrt{T}N} = \frac{s\sqrt{T}}{N},$$

where we consider a constant step size s (hence, $\theta = 1$). Substituting this in the final formula for τ of Section C.1 yields for $\epsilon \gg \delta$

$$\tau \approx \sqrt{\frac{\ln(1/\delta) + \ln(\epsilon) - O(\ln\ln(1/\delta))}{2\gamma\epsilon}}.$$

Here, $\tau \approx \sigma$ for $\sigma \gg 1$ and we see that compared to Theorem 3.1 this formula attains a factor $2\sqrt{\gamma} \approx \sqrt{8}$ smaller σ for the same (ϵ, δ). This shows that in this asymptotic setting for $N \to \infty$, $T \to \infty$, $\epsilon \gg \delta$, and $\sigma \gg 1$, Theorem 3.1 is up to a factor $2\sqrt{\gamma}$ tight. In other words, the formula in Theorem 3.1 pays a factor in tightness in order to hold for general concrete parameter settings (not just the asymptotic setting).

The factor $2\sqrt{\gamma}$ seems large. However, if the lower bound on T can be tightened to $T \geq \frac{\theta^2}{4\epsilon} \cdot k^2$, then the constant c above can be increased to $c = \sqrt{4\epsilon/\theta^2}$ leading to a formula for τ that matches the formula for σ in Theorem 3.1 implying that it is tight in the asymptotic setting. In other words, rather than expecting to be able to lower the constant 2 in $\sigma = \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}$, we can focus on how much the lower bound of T can be reduced by a small factor (notice that the derivations in Theorem B.4 that lead to this lower bound may be tightened up). We expect the bound $\sigma \geq \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon}$ to be quite tight, while the lower bound $T \geq T_{min} = \frac{\gamma\theta^2}{\epsilon} \cdot k^2$ can at most be reduced by a factor 4γ to $T \geq T_{min\_asym} = \frac{\theta^2}{4\epsilon} \cdot k^2$. For smaller $T < T_{min\_asym}$, the asymptotic setting for $N \to \infty$, $T \to \infty$, $\epsilon \gg \delta$, and $\sigma \gg 1$ contradicts the tightness of the asymptotic result (65) of the f-DP framework. This proves Theorem 3.2. We notice that this is also (unsurprisingly) confirmed by experiments in Appendix D.4 where we implement the f-DP accountant in order to compute which ϵ is actually being achieved.

C.3 Interpreting Theorem 3.1 in the f-DP Framework

In the f-DP framework, if a mechanism is f-DP, then it is (ϵ, δ)-DP for all (ϵ, δ) for which $f \geq f_{\epsilon,\delta}$. When considering DP-SGD for parameters σ and T and the other hyper parameters fixed, there exists some function $h_{\sigma,T}$ such that DP-SGD is f-DP if and only if $h_{\sigma,T} \geq f$. This function implies (ϵ, δ)-DP for all $h_{\sigma,T} \geq f_{\epsilon,\delta}$. Conversely, if we want to realize (ϵ, δ)-DP for some target privacy budget defined by (ϵ, δ), then we need to choose σ and T such that $h_{\sigma,T} \geq f_{\epsilon,\delta}$.

We notice that Theorem 3.1 can be cast in the f-DP framework as it allows us to formulate an appropriate (non-tight) $\hat{h}_{\sigma,T}$ for which DP-SGD is $\hat{h}_{\sigma,T}$-DP as follows: We first notice that by the definition of $h_{\sigma,T}$ we have

$$h_{\sigma,T} \geq \hat{h}_{\sigma,T}.$$

By fixing σ and T, we may freely choose (ϵ, δ) as long as the conditions of Theorem 3.1 are satisfied: $(\epsilon, \delta) \in \mathcal{H}(\sigma, T)$ with

$$\mathcal{H}(\sigma, T) = \left\{ (\epsilon, \delta) \ : \ \begin{array}{c} \sigma \geq \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon} \\ \delta \leq 1/N \\ \epsilon < 0.5 \\ k \geq \sqrt{2\epsilon\ln(1/\delta)} \cdot e/(\gamma(\sigma, \epsilon, k) \cdot \theta) \\ T \geq \gamma(\sigma, \epsilon, k) \cdot \theta^2 k^2/\epsilon \end{array} \right\},$$

where γ is a function of σ, ϵ and $k = K/N$. Hyper parameters K and N (and, therefore also k) are fixed. For a given T, we consider a constant sample size $s = K/T$ from round to round, hence, we use $\theta = 1$ in the definition of $\mathcal{H}(\sigma, T)$. By defining

$$\hat{h}_{\sigma,T}(\alpha) = \sup_{(\epsilon,\delta) \in \mathcal{H}(\sigma,T)} f_{\epsilon,\delta}(\alpha),$$

we have that DP-SGD is $\hat{h}_{\sigma,T}$-DP.

Now consider a target epsilon $\epsilon = \epsilon_{target}$ and define a new set

$$\mathcal{G}(\sigma, \epsilon_{target}) = \left\{ (\epsilon, \delta) \ : \ \begin{array}{c} \sigma \geq \sqrt{2(\epsilon + \ln(1/\delta))/\epsilon} \\ \delta \leq 1/N \\ \epsilon < 0.5 \\ k \geq \sqrt{2\epsilon\ln(1/\delta)} \cdot e/(\gamma(\sigma, \epsilon, k) \cdot \theta) \\ \epsilon \geq \epsilon_{target} \end{array} \right\}.$$

Notice that $\mathcal{G}(\sigma, \epsilon_{target})$ is independent of T, but does satisfy the property

$$\mathcal{G}(\sigma, \epsilon_{target}) \subseteq \mathcal{H}(\sigma, T) \text{ for } T \geq \gamma\theta^2 k^2/\epsilon_{target}.$$

Therefore,

$$g_{\sigma,\epsilon_{low}}(\alpha) = \sup_{(\epsilon,\delta)\in\mathcal{G}(\sigma,\epsilon_{target})} f_{\epsilon,\delta}(\alpha),$$

is also independent of T while it satisfies

$$\hat{h}_{\sigma,T} \geq g_{\sigma,\epsilon_{target}} \text{ for } T \geq \gamma\theta^2 k^2/\epsilon_{target}.$$

Together with $h_{\sigma,T} \geq \hat{h}_{\sigma,T}$ we conclude

$$h_{\sigma,T} \geq g_{\sigma,\epsilon_{target}} \text{ for } T \geq \gamma\theta^2 k^2/\epsilon_{target}.$$

This shows how Theorem 3.1 translates to the f-DP framework in that as long as the number of rounds is large enough, that is, $K \geq T \geq \gamma\theta^2 k^2/\epsilon_{target}$, we have that there always remains some f-DP privacy guarantee given by $g_{\sigma,\epsilon_{target}}$. Notice that $g_{\sigma,\epsilon_{target}}$ is independent of T. So, even in the limit for larger T, not all of the privacy budget is being depleted. For larger T, $f_{\sigma,T}$ remains at least $g_{\sigma,\epsilon_{target}}$. Since the f-DP framework and its resulting DP accountant provide a tight analysis, this shows that an increasing number T of rounds revealing more and more local updates does not linearly increase the privacy leakage! Instead, a larger T gives rise to more updates that each leak privacy and gives rise to subsampling of smaller mini-batches which amplifies the differential privacy guarantee more, hence, less leakage per round. The total resulting leakage remains bounded in that we can always guarantee $g_{\sigma,\epsilon_{target}}$-DP (even for larger T). This implies that the tight f-DP based privacy accountant will have a limit (upper bound) on its reported privacy leakage for increasing T ($\leq K$).

D Experiments

We provide experiments to support our theoretical findings, i.e., convergence of our proposed asynchronous distributed learning framework with differential privacy (DP) to a sufficiently accurate solution. We cover strongly convex, plain convex and non-convex objective functions over iid local data sets.

We introduce our experimental set up in Section D.1. Section D.2 provides utility graphs for different data sets and objective functions. A utility graph helps choosing the maximum possible noise σ, in relation to the value of the clipping constant C, for which decent accuracy can be achieved. Section D.3 provides detailed experiments for our asynchronous differential privacy SGD framework (asynchronous DP-SGD) with different types of objective functions (i.e., strongly convex, plain convex and non-convex objective functions), different types of constant sample size sequences and different levels of privacy guarantees (i.e., different privacy budgets ϵ).

All our experiments are conducted on LIBSVM [3][***] , MNIST [17] [†††], and CIFAR10[‡‡‡] data sets.

D.1 Experiment settings

Simulation environment. For simulating the asynchronous DP-SGD framework, we use multiple threads where each thread represents one compute node joining the training process. The experiments are conducted on Linux-64bit OS, with 16 cpu processors, and 32Gb RAM.

Objective functions. Equation (66) defines the plain convex logistic regression problem. The weight vector w and the bias value b of the logistic function can be learned by minimizing the log-likelihood function J:

$$J = -\sum_{i=1}^{N}[y_i \cdot \log(\bar{\sigma}_i) + (1-y_i)\cdot\log(1-\bar{\sigma}_i)], \text{ (plain convex)} \tag{66}$$

where N is the number of training samples (x_i, y_i) with $y_i \in \{0,1\}$ and $\bar{\sigma}_i$ is defined by

$$\bar{\sigma}_i = \frac{1}{1+e^{-(w^{\mathrm{T}}x_i+b)}},$$

which is the sigmoid function with parameters w and b. Our goal is to learn a vector w^* which represents a pair $\bar{w} = (w, b)$ that minimizes J.

[***]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/binary.html
[†††]http://yann.lecun.com/exdb/mnist/
[‡‡‡]https://www.cs.toronto.edu/ kriz/cifar.html

Function J can be changed into a strongly convex problem $\hat{J}$ by adding a regularization parameter $\lambda > 0$:

$$\hat{J} = -\sum_{i=1}^{N} [y_i \cdot \log(\sigma_i) + (1 - y_i) \cdot \log(1 - \sigma_i)] + \frac{\lambda}{2} \|w\|^2 \text{ , (strongly convex).}$$

where $\bar{w} = (w, b)$ is vector w concatenated with bias value b. In practice, the regularization parameter λ is set to $1/N$ [29].

For simulating non-convex problems, we choose a simple neural network (Letnet) [18] for MNIST data set and AlexNet [15] for CIFAR10 data set with cross entropy loss function for image classification.

The loss functions for the strong, plain, and non-convex problems represent the objective function $F(.)$.

Parameter selection. The parameters used for our distributed algorithm with Gaussian based differential privacy for strongly convex, plain convex and non-convex objective functions are described in Table 1. The clipping constant C is set to 0.1 for strongly convex and plain convex problems and 0.025 for non-convex problem (this turns out to provide good utility).

Table 1: Common parameters of asynchronous DP-SGD framework with differential privacy

	# of clients n	Diminishing step size $\bar{\eta}_t$	Regular λ	Clipping constant C
Strongly convex	5	$\frac{\eta_0}{1+\beta t}$ ‡	$\frac{1}{N}$	0.1
Plain convex	5	$\frac{\eta_0}{1+\beta t}$ or $\frac{\eta_0}{1+\beta\sqrt{t}}$	N/A	0.1
Non-convex	5	$\frac{\eta_0}{1+\beta\sqrt{t}}$	N/A	0.025

‡ The i-th round step size $\bar{\eta}_i$ is computed by substituting $t = \sum_{j=0}^{i-1} s_j$ into the diminishing step size formula.

For the plain convex case, we can use diminishing step size schemes $\frac{\eta_0}{1+\beta \cdot t}$ or $\frac{\eta_0}{1+\beta \cdot \sqrt{t}}$. In this paper, we focus our experiments for the plain convex case on $\frac{\eta_0}{1+\beta \cdot \sqrt{t}}$. Here, η_0 is the initial step size and we perform a systematic grid search on parameter $\beta = 0.001$ for strongly convex case and $\beta = 0.01$ for both plain convex and non-convex cases. Moreover, most of the experiments are conducted with 5 compute nodes and 1 central server. When we talk about accuracy (from Figure 7 and onward), we mean test accuracy defined as the fraction of samples from a test data set that get accurately labeled by the classifier (as a result of training on a training data set by minimizing a corresponding objective function).

D.2 Utility graph

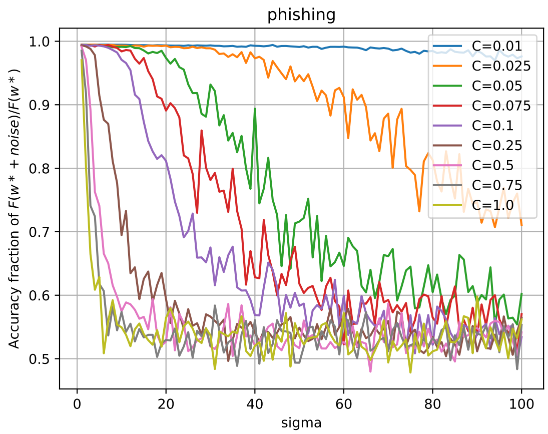
The purpose of a utility graph is to help us choose, given the value of the clipping constant C, the maximum possible noise σ for which decent accuracy can be achieved. A utility graph depicts the test accuracy of model $F(w^* + n)$ over $F(w^*)$, denotes as accuracy fraction, where w^* is a near optimal global model and $n \sim \mathcal{N}(0, C^2\sigma^2\mathbf{I})$ is Gaussian noise. This shows which maximum σ can be chosen with respect to allowed loss in expected test accuracy, clipping constant C and standard deviation σ.

As can be seen from Figure 4 and Figure 5, for clipping constant $C = 0.1$, we can choose the maximum σ somewhere in the range $\sigma \in [18, 22]$ if we want to guarantee there is at most about 10% accuracy loss compared to the (near)-optimal solution without noise. Another option is $C = 0.075$, where we can tolerate $\sigma \in [18, 30]$ yielding the same accuracy loss guarantee. When the gradient bound C gets smaller, our DP-SGD can tolerate bigger noise, i.e, bigger values of σ. However, we need to increase the number K of iterations during the training process when C is smaller in order to converge and gain a specific test accuracy – this is the trade-off. For simplicity, we intentionally choose $C = 0.1$, $\sigma \leq 20$ and expected test accuracy loss about 10% for our experiments with strongly convex and plain convex objective functions.

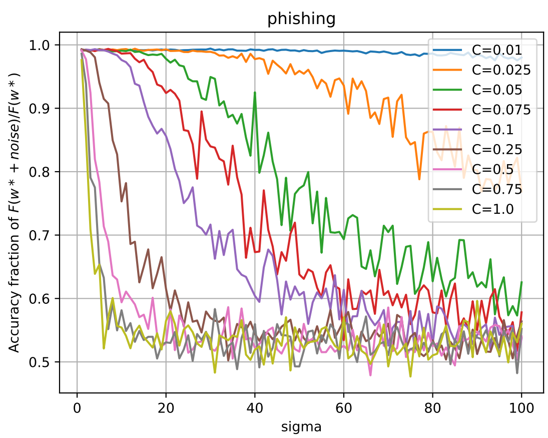
The utility graph is extended to the non-convex objective function in Figure 6. To keep the test accuracy loss less or equal to 10% (of the final test accuracy of the original model w^*), we choose $C = 0.025$ and noise level $\sigma \leq 12$ for MNIST data set (as shown in Figure 6(a)) and $C = 0.025$ and noise level $\sigma \leq 6.572$ for CIFAR10 data set (as shown in Figure 6(b)). For simplicity, we use this parameter setting for our experiments with the non-convex problem.

D.3 Asynchronous distributed learning with differential privacy

We consider the asynchronous DP-SGD framework with strongly convex, plain convex and non-convex objective functions for different settings, i.e., different levels of privacy budget ϵ and different constant sample size sequences.



(a) Strong convex.



(b) Plain convex.

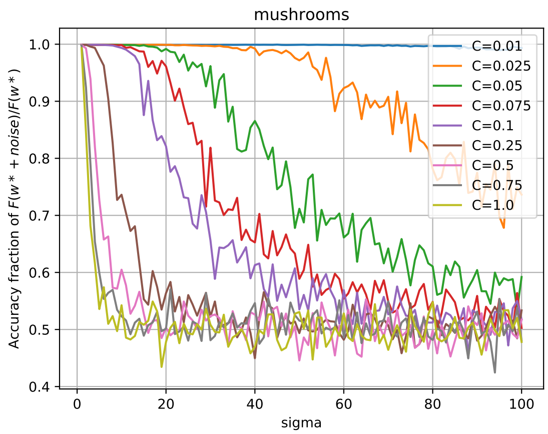
Figure 4: Utility graph with various gradient norm C and noise level σ

D.3.1 Asynchronous DP-SGD with different constant sample size sequences

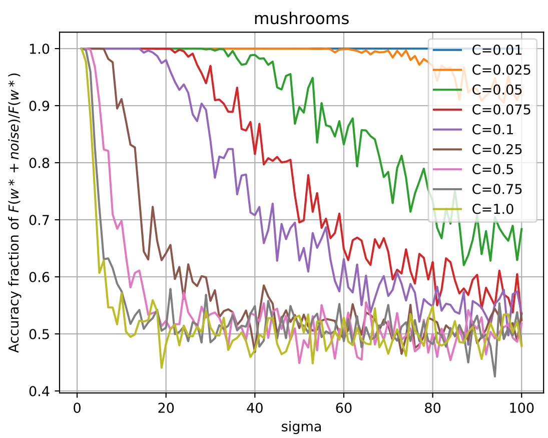
The purpose of this experiment is to investigate which is the best constant sample size sequence $s_i = s$. This experiment allows us to choose a decent sample size sequence that will be used in our subsequent experiments. To make the analysis simple, we consider our asynchronous DP-SGD framework with $\Upsilon(k, i)$ defined as false if and only if $k < i - 1$, i.e., compute nodes are allowed to run fast and/or have small communication latency such that broadcast global models are at most 1 local round in time behind (so different clients can be asynchronous with respect to one another for 1 local round). We also use iid data sets. The detailed parameters are in Table 2.

The results from Figure 7 to Figure 8 confirm that our asynchronous DP-SGD framework can converge under a very small privacy budget. When the constant sample size $s = 1$, it is clear that the DP-SGD algorithm does not achieve good accuracy compared to other constant sample sizes even though this setting has the maximum number of communication rounds. When we choose constant sample size $s = 26$ (this meets the upper bound for constant sample sizes for our small $N = 10,000$ and small $\epsilon \approx 0.05$, see Theorem B.4), our DP-SGD framework converges to a decent test accuracy, i.e, the test accuracy loss is expected less than or equal to 10% when compared to the original mini-batch SGD without noise. In conclusion, this experiment demonstrates that our asynchronous DP-SGD with diminishing step size scheme and constant sample size sequence works well under DP setting, i.e, our asynchronous DP-SGD framework can gain differential privacy guarantees while maintaining an acceptable accuracy.

We also conduct the experiment for the non-convex objective function with MNIST and CIFAR10 data sets. The detailed parameter settings can be found in Table 3 and Table 4. Here, we again consider our asynchronous setting



(a) Strong convex.



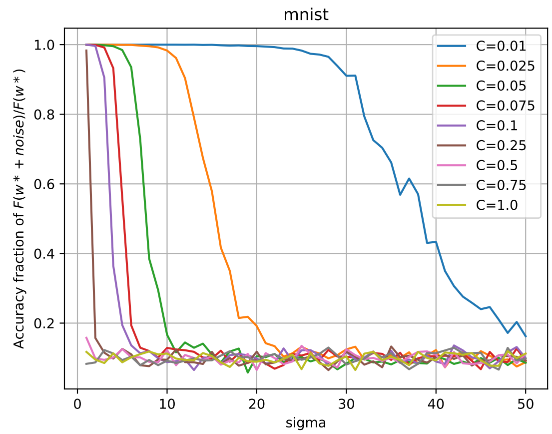
(b) Plain convex.

Figure 5: Utility graph with various gradient norm C and noise level σ

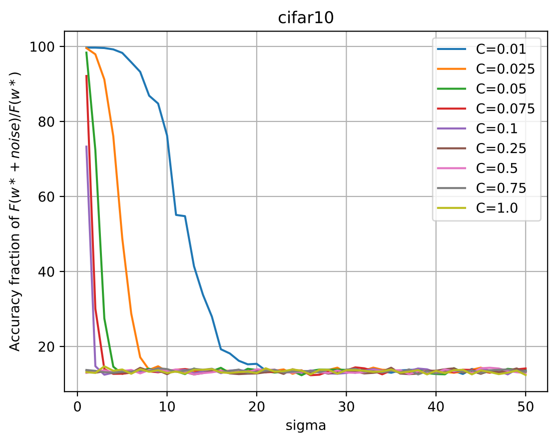
Table 2: Basic parameter setting for strongly and plain convex problems

Parameter	Value	Note
$\bar{\eta}_0$	0.1	initial stepsize
N_c	$10,000$	# of data points
K	$50,000$	# of iterations
ϵ	0.04945	
σ	19.29962	
δ	0.0001	
C	0.1	clipping constant
s	$\{1, 5, 10, 15, 20, 26\}$	constant sample size sequence
dataset	LIBSVM	iid dataset
n	5	# of nodes
Υ	$k \geq i - 1$	$1-$asynchronous round

where each compute node is allowed to run fast and/or has small communication latency such that broadcast global models are at most 1 local round in time behind. As can be seen from Figure 9 (with MNIST data set), our proposed asynchronous DP-SGD still converges under small privacy budget. Moreover, when we use the constant sample size $s = 370$ (this meets the upper bound for constant sample sizes for our small $N = 60,000$ and small $\epsilon \approx 0.15$, see Theorem B.4), we can significantly reduce the communication cost compared to other constant sample sizes while



(a)



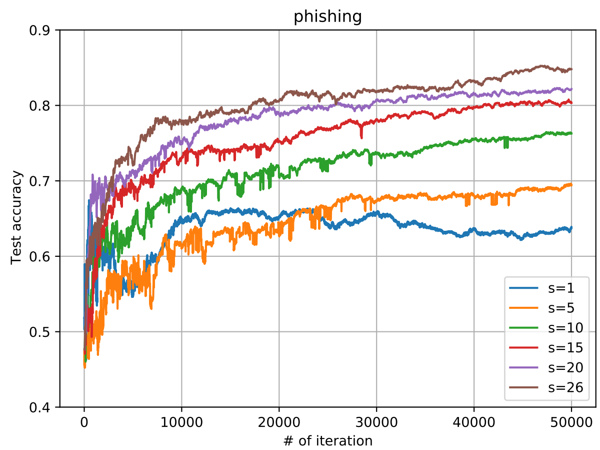
(b)

Figure 6: Utility graph with various gradient norm C and noise level σ for MNIST and CIFAR10 data sets.

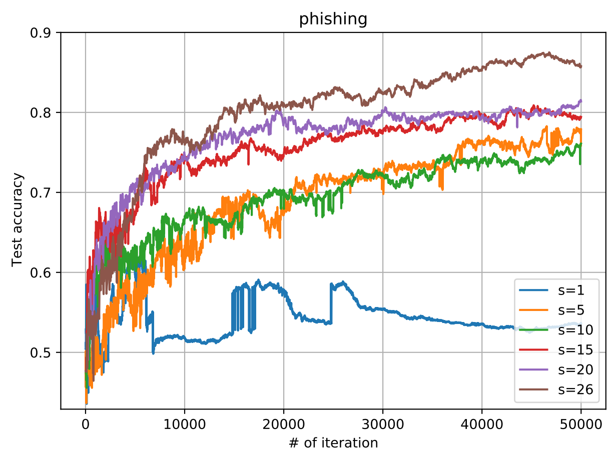
Table 3: Basic parameter setting for non-convex problem with MNIST data set

Parameter	Value	Note
$\bar{\eta}_0$	0.1	initial stepsize
N_c	$60,000$	# of data points
K	$360,000$	# of iterations
ϵ	0.15007	
σ	12.10881	
δ	$1.667 \cdot 10^{-5}$	
C	0.025	clipping constant
s	$\{10, 25, 50, 100, 200, 300, 370\}$	constant sample size sequence
dataset	MNIST	iid dataset
n	5	# of nodes
Υ	$k \geq i - 1$	$1-$asynchronous round

keeping the test accuracy loss within 10%. The constant sample size $s = 10$ (as well as $s \leq 10$) shows a worse performance while this setting requires more communication rounds, compared to other constant sample sizes. We can observe the same pattern for CIFAR10 data set as shown in Figure 10, where we can choose the constant sample size $s \leq 689$ with $N = 50,000$ data points and $\epsilon \approx 0.5$. While the constant sample size s satisfying $300 \leq s \leq 689$, the test accuracy gets the highest level while the constant sample size $s \leq 50$ deteriorates the performance of accuracy



(a) Strong convex.



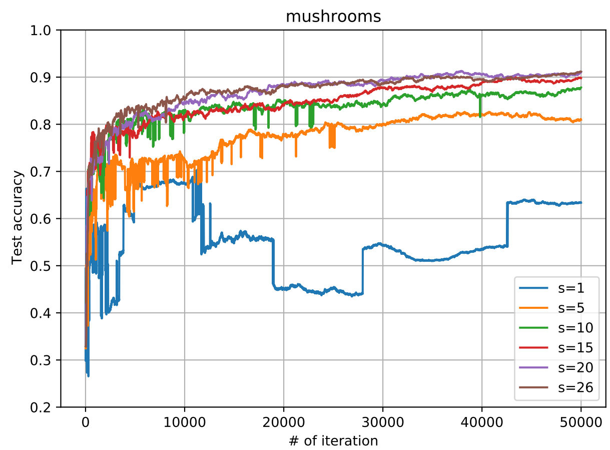
(b) Plain convex.

Figure 7: Effect of different constant sample size sequences

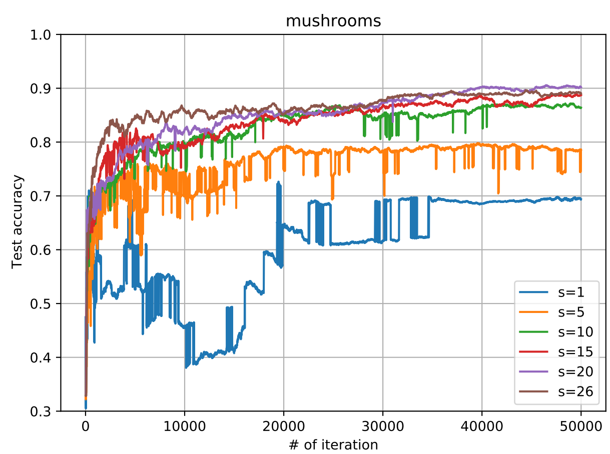
Table 4: Basic parameter setting for non-convex problem for CIFAR10 data set

Parameter	Value	Note
$\bar{\eta}_0$	0.1	initial stepsize
N_c	$50,000$	# of data points
K	$350,000$	# of iterations
ϵ	0.50102	
σ	6.572	
δ	$2 \cdot 10^{-5}$	
C	0.025	clipping constant
s	$\{10, 25, 50, 100, 300, 500, 689\}$	constant sample size sequence
dataset	CIFAR10	iid dataset
n	5	# of nodes
Υ	$k \geq i - 1$	$1-$asynchronous round

significantly. This figure again confirms the effectiveness of our asynchronous DP-SGD framework towards a strong privacy guarantee for all types of objective function.



(a) Strong convex.



(b) Plain convex.

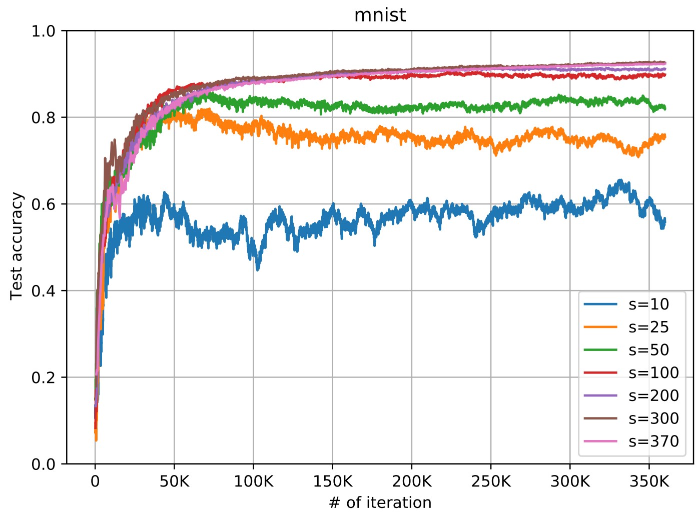
Figure 8: Effect of different constant sample size sequences

Table 5: Different privacy budget settings for strongly and plain convex problems

Privacy budget (ϵ, δ)	σ	Sample size s
$(0.04945, 0.0001)$	19.29962	26
$(0.1, 0.0001)$	13.06742	55
$(0.25, 0.0001)$	8.59143	103
$(0.5, 0.0001)$	6.05868	168
$(1.0, 0.0001)$	4.27273	265
$(2.0, 0.0001)$	3.03241	400

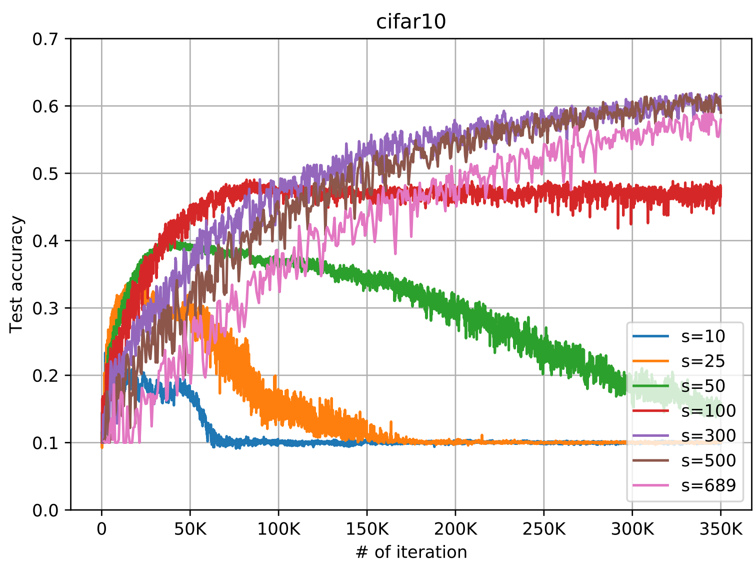
D.3.2 Asynchronous DP-SGD with different levels of privacy budget

We conduct the following experiments to compare the effect of our DP-SGD framework for different levels of privacy budget ϵ including the non-DP setting (i.e., no privacy at all, hence, no noise). The purpose of this experiment is to show that the test accuracy degradation is at most 10% even if we use very small ϵ. The detailed constant sample sequence s and noise level σ based on Theorem B.4 are illustrated in Table 5. Other parameter settings, such as initial stepsize η_0, are kept the same as in Table 2.



(a) Non-convex.

Figure 9: Effect of different constant sample size sequences

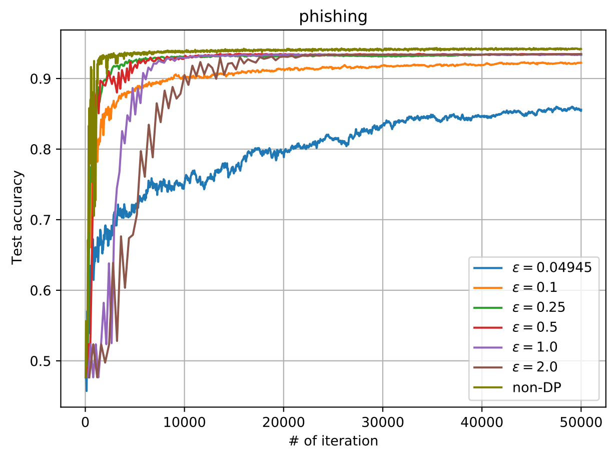


(a) Non-convex.

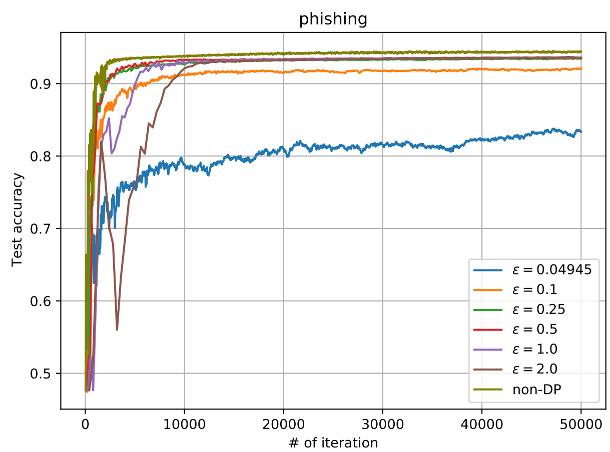
Figure 10: Effect of different constant sample size sequences.

As can be seen from Figures 11 and Figure 12, the test accuracy degradation is about 10% for $\epsilon = 0.04945$ compared to the other graphed privacy settings and non-DP setting. Privacy budget $\epsilon = 0.1$, still significant smaller than what is reported in literature, comes very close to the maximum attainable test accuracy of the non-DP setting.

We ran the same experiment for the non-convex objective function. The detailed setting of different privacy budgets is shown in Table 6. Note that we also set the asynchronous behavior to be 1 asynchronous round, and the total of iterations on each compute node is $K = 360,000$. Other parameter settings for the non-convex case, such as initial stepsize η_0, are kept the same as in Table 3. As can be seen from Figure 13 with MNIST data set, the test accuracy loss with $\epsilon \approx 0.15$ is less than 10% (the expected test accuracy degradation from utility graph at Figure 6). Another pattern can be found in Figure 14. By selecting $\epsilon \approx 0.5$ for CIFAR10 data set, the test accuracy reduces less than 10%,



(a) Strong convex.



(b) Plain convex.

Figure 11: Effect of different levels of privacy budgets ϵ and non-DP settings

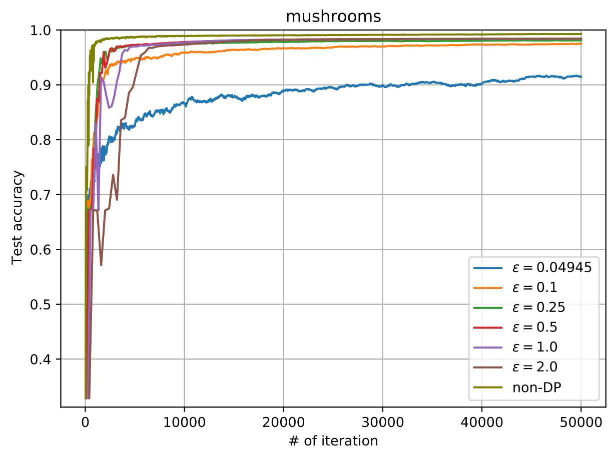
compared to the non-DP setting. Note that we use AlexNet for CIFAR10, which shows ≈ 0.74 maximum test accuracy in practice[§§§].

Table 6: Different privacy budget settings for non-convex problem for MNIST data set

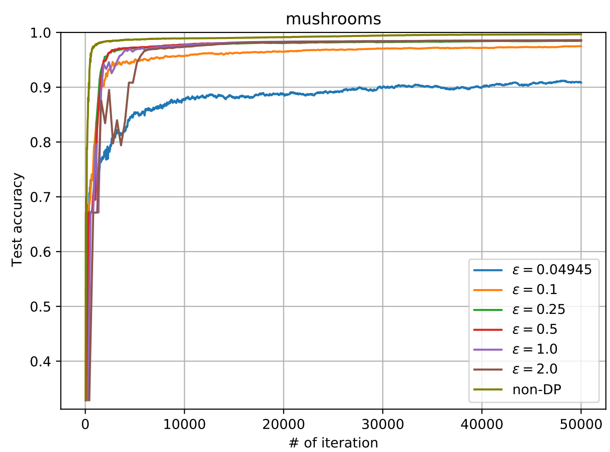
Privacy budget (ϵ, δ)	σ	Sample size s
$(0.15007, 1.667 \cdot 10^{-5})$	12.10881	370
$(0.2, 1.667 \cdot 10^{-5})$	10.48452	460
$(0.25, 1.667 \cdot 10^{-5})$	9.37379	543
$(0.5, 1.667 \cdot 10^{-5})$	6.63120	889
$(0.75, 1.667 \cdot 10^{-5})$	5.41887	1168
$(1.0, 1.667 \cdot 10^{-5})$	4.69244	1409
$(2.0, 1.667 \cdot 10^{-5})$	3.31648	2159

These figures again confirm the effective performance of our DP-SGD framework, which not only conserves strong privacy, but also keeps a decent convergence rate to good accuracy, even for a very small privacy budget.

[§§§] https://github.com/icpm/pytorch-cifar10



(a) Strong convex.



(b) Plain convex.

Figure 12: Effect of different levels of privacy budgets ϵ and non-DP settings

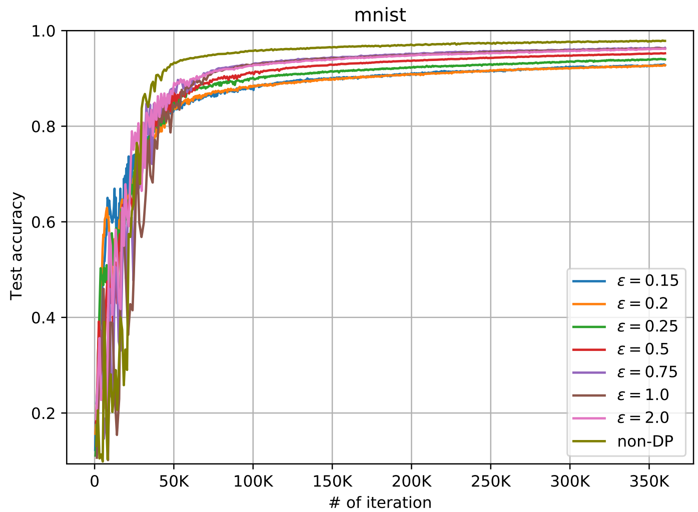
Table 7: Different privacy budget settings for non-convex problem for CIFAR10 data set

Privacy budget (ϵ, δ)	θ	Sample size S
$(0.25, 2.0 \cdot 10^{-5})$	9.29838	417
$(0.5, 2.0 \cdot 10^{-5})$	6.57192	689
$(0.75, 2.0 \cdot 10^{-5})$	5.36937	909
$(1.0, 2.0 \cdot 10^{-5})$	4.65014	1099
$(1.5, 2.0 \cdot 10^{-5})$	4.16111	1267
$(2.0, 2.0 \cdot 10^{-5})$	3.28831	1690
$(3.0, 2.0 \cdot 10^{-5})$	2.68273	1994

D.4 Comparison to the f-DP Accountant

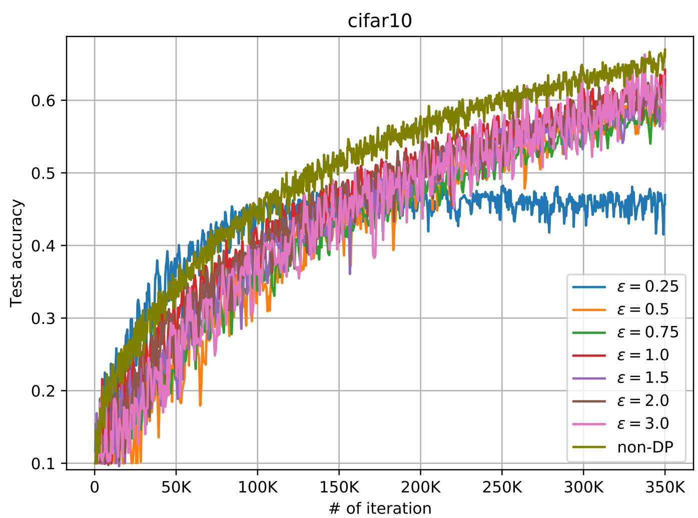
We have implemented a simplified differential privacy calculator based on Theorem 3.1 for computing the optimal privacy budget (ϵ, δ) given the training hyper-parameters $(\sigma, \theta, N, k, C)$. This calculator has the follow steps:

1. Set $\delta = 1/N$, $\epsilon = \frac{2 \ln 1/\delta}{\sigma^2 - 2}$.
2. Set $\gamma = 2$ (because $\gamma = 2 + O(\bar{\alpha})$) as the initial value.



(a) Non-convex.

Figure 13: Effect of different levels of privacy budgets ϵ and non-DP settings



(a) Non-convex.

Figure 14: Effect of different levels of privacy budgets ϵ and non-DP settings

3. According to Theorem 3.1, we compute T_{min} as a result of next steps (3, 4, and 5) as a lower bound on T as follows:
   - Compute $\bar{\alpha} = \frac{\epsilon N}{\gamma K}$.
   - Recompute the new $\gamma_{new} = \frac{2}{1-\bar{\alpha}} + \frac{2^4 \cdot \bar{\alpha}}{1-\bar{\alpha}} \left(\frac{\sigma}{(1-\sqrt{\bar{\alpha}})^2} + \frac{1}{\sigma(1-\bar{\alpha})-2e\sqrt{\bar{\alpha}}} \frac{e^3}{\sigma} \right) e^{3/\sigma^2}$.
4. Repeat steps 3 until $\gamma_{new} - \gamma \leq 0.0001\gamma$ or inequality (8) is violated. In the latter case the calculator cannot find a solution of a set of hyperparameters that satisfies the privacy constraint (ϵ, δ) and we lower σ and increase ϵ accordingly (in step 1) and repeat steps 2, 3, and 4.
5. Based on inequality (9) we compute minimal value $T_{min} = \frac{\gamma\theta^2}{\epsilon} k^2$. From the asymptotic tightness analysis in section C.2 we learn that T_{min} can at most be lowered to $T_{min\_asym} = \frac{\theta^2 k^2}{4\epsilon}$.
6. Corresponding to T_{min} and T_{min_asym}, we set $s_{max} = \frac{kN}{T_{min}}$ and $s_{max\_asym} = \frac{kN}{T_{min\_asym}}$

7. We obtain the resulting set of parameters $(\epsilon, \delta, \sigma, \gamma, \theta, k, N, s, T)$ for (s, T) equal to (s_{max}, T_{min}) and $(s_{max\_asym}, T_{min\_asym})$ respectively.

This calculation helps us planning ahead the number of rounds T and the sampling rate s and we can choose ϵ_{target} by defining an initial σ.

Given the hyper-parameters defined in Tables 2, 3 and 4, we use this calculator to compute ϵ_{target} and compared this with the value ϵ_{f-DP} as a result from the exact/tight f-DP accountant from [6], see github.com/tensorflow/privacy. This leads to Tables 8, 9, and 10. Since ϵ_{f-DP} is tight, we conclude that ϵ_{target} for T_{min_asymp} cannot be achieved, hence, our provided formula for T_{min} is indeed tight up to a factor 4γ as mentioned in the main body and studied in Appendix C.2.

T_{min} and T_{min_asym}	$\frac{\gamma\theta^2 k^2}{\epsilon}$	$\frac{\theta^2 k^2}{4\epsilon}$
$s_{max} = \frac{kN}{T_{min}}$ and $s_{max\_asym} = \frac{kN}{T_{min}}$	26	397
ϵ_{target}	0.0497	0.0497
ϵ_{f-DP}	0.0105	0.0533
Multiplication factor $(\epsilon_{target}/\epsilon_{f-DP})$	4.7234	0.9328

Table 8: Comparison of ϵ_{target} with ϵ_{f-DP} from the f-DP accountant for Table 2 where $\theta = 1, \delta = 1/10000$ and $\sigma = 19.29962$ for LIBSVM dataset ($N = 10000$), $k = 5$.

T_{min} and T_{min_asym}	$\frac{\gamma\theta^2 k^2}{\epsilon}$	$\frac{\theta^2 k^2}{4\epsilon}$
$s_{max} = \frac{kN}{T_{min}}$ and $s_{max\_asym} = \frac{kN}{T_{min}}$	288	6085
ϵ_{target}	0.1521	0.1521
ϵ_{f-DP}	0.0389	0.2097
Multiplication factor $(\epsilon_{target}/\epsilon_{f-DP})$	3.9147	0.7257

Table 9: Comparison of ϵ_{target} with ϵ_{f-DP} from the f-DP accountant for Table 2 where $\theta = 1, \delta = 1/60000$ and $\sigma = 12.10881$ for MNIST dataset ($N = 60000$), $k = 6$.

T_{min} and T_{min_asym}	$\frac{\gamma\theta^2 k^2}{\epsilon}$	$\frac{\theta^2 k^2}{4\epsilon}$
$s_{max} = \frac{kN}{T_{min}}$ and $s_{max\_asym} = \frac{kN}{T_{min}}$	406	15009
ϵ_{target}	0.5253	0.5253
ϵ_{f-DP}	0.1133	0.8239
Multiplication factor $(\epsilon_{target}/\epsilon_{f-DP})$	4.6372	0.6376

Table 10: Comparison of ϵ_{target} with ϵ_{f-DP} from the f-DP accountant for Table 2 where $\theta = 1, \delta = 1/50000$ and $\sigma = 6.572$ for CIFAR10 dataset ($N = 50000$), $k = 7$.