# Learning Rate Adaptation for Differentially Private Learning

**Antti Koskela**
University of Helsinki

**Antti Honkela**
University of Helsinki

## Abstract

Differentially private learning has recently emerged as the leading approach for privacy-preserving machine learning. Differential privacy can complicate learning procedures because each access to the data needs to be carefully designed and carries a privacy cost. For example, standard parameter tuning with a validation set cannot be easily applied. In this paper, we propose a differentially private algorithm for the adaptation of the learning rate for differentially private stochastic gradient descent (SGD) that avoids the need for validation set use. The idea for the adaptiveness comes from the technique of extrapolation in numerical analysis: to get an estimate for the error against the gradient flow we compare the result obtained by one full step and two half-steps. We prove the privacy of the method using the moments accountant mechanism. This allows us to compute tight privacy bounds. Empirically we show that our method is competitive with manually tuned commonly used optimisation methods for training deep neural networks and differentially private variational inference.

## 1 Introduction

Stochastic gradient descent (SGD) and its variants, including AdaGrad (Duchi et al., 2011), RMSProp (Tieleman and Hinton, 2012) and Adam (Kingma and Ba, 2015), are the main workhorses of modern machine learning and deep learning. These methods are quite sensitive to tuning the learning rate, which usually requires testing different alternatives and evaluating them on a validation dataset. When possible, this adds significantly to the computational cost of using them, especially in the deep learning setting (Strubell et al., 2019). However, there are also situations where proper validation is difficult, such as differentially private learning. In this setting, effective adaptive algorithms can be extremely important for efficient learning. While adaptive SGD alternatives such as AdaGrad, RMSProp and Adam are not as sensitive to tuning as plain SGD, they nevertheless require tuning for good performance. Furthermore, Wilson et al. (2017) argue that commonly used adaptive methods such as AdaGrad, RMSProp and Adam can lead to very poor generalisation performance in deep learning and that properly tuned basic SGD is a very competitive approach.

Differential privacy (DP) (Dwork et al., 2006) has recently risen as the dominant paradigm for privacy-preserving machine learning. A number of differentially private algorithms have been proposed addressing both important specific models (Abadi et al., 2016; Chaudhuri and Monteleoni, 2008; Dwork et al., 2014) as well as more general approaches to learning (Chaudhuri et al., 2011; Dimitrakakis et al., 2014; Jälkö et al., 2017; Park et al., 2016; Zhang et al., 2016). Differentially private stochastic gradient descent (DP-SGD) (Abadi et al., 2016; Rajkumar and Agarwal, 2012; Song et al., 2013) has emerged as an important tool for implementing differential privacy for a number of applications. The introduction of very tight bounds on the privacy loss occurring during the iterative algorithm computed via the moments accountant (Abadi et al., 2016) has made these algorithms particularly attractive. Furthermore, DP's invariance to post-processing means that the same privacy guarantees apply to any algorithm that uses the same gradient information, including adaptive and accelerated methods. In addition to deep learning, stochastic gradients and more recently the moments accountant have been used in algorithms for other paradigms, such as Bayesian inference (Jälkö et al., 2017; Li et al., 2019; Wang et al., 2015).

It is clear that standard hyperparameter tuning methods typically used for tuning the learning rate are not

Proceedings of the 23rdInternational Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. 

directly applicable to DP learning because of the need to account for the additional privacy loss for multiple runs of the learning and validation set use. Most previous work glosses this over, with two notable exceptions. Kusner et al. (2015) presented DP Bayesian optimisation that accounts for the privacy loss for the validation set, but they completely ignore training set privacy. Liu and Talwar (2019) recently introduced DP meta selection for DP hyperparameter tuning, but their approach only supports random hyperparameter search which may carry a significant computational cost.

In this paper, we propose a rigorous adaptive method for finding a good learning rate for SGD, and apply it in DP learning setting. The adaptation is performed during learning, which implies that the learning process only has to be executed once, leading to savings in compute time and efficient use of the privacy budget. We prove the privacy of our method based on the moments accountant mechanism.

**Main contributions**

We propose the first learning rate adaptive DP SGD method. We give rigorous moment bounds for the method, and using these bounds, we can compute $(\varepsilon, \delta)$-bounds using the so called moments accountant technique. By simple derivations, we show how to determine the additional tolerance hyperparameter in the algorithm. In experiments we show that it is competitive with optimally tuned standard optimisation methods without any tuning. We also illustrate the benefits when compared to the DP meta selection algorithms introduced by Liu and Talwar (2019).

## 2 Motivation for the learning rate adaptation: extrapolation of differential equations

The main ingredient of the learning rate adaptation comes from numerical extrapolation of ordinary differential equations (ODEs), see e.g. (Hairer et al., 1987). We next describe this idea. Let $g$ be a differentiable function $g : \mathbb{R}^d \to \mathbb{R}$. Gradient descent (GD)

$$\theta_{\ell+1} = \theta_\ell - \eta_\ell \nabla g(\theta_\ell) \tag{2.1}$$

is a first-order method for finding a (local) minimum of the function $g$. It can be seen as an explicit Euler method with the step size $\eta_\ell$ applied to the system of ODEs

$$\frac{d}{dt}\theta(t) = -\nabla g(\theta), \quad \theta(0) = \theta_0 \in \mathbb{R}^d,$$

which is also called the gradient flow of $g$.

To get an estimate of the error made in the numerical approximation (2.1), we extrapolate it as follows. Consider one step of size $\eta$,

$$\theta_1 = \theta_0 - \eta \nabla g(\theta_0), \tag{2.2}$$

and $\widehat{\theta}_1$ which is a result of two steps of size $\frac{\eta}{2}$:

$$\theta_{1/2} = \theta_0 - \frac{\eta}{2}\nabla g(\theta_0), \quad \widehat{\theta}_1 = \theta_{1/2} - \frac{\eta}{2}\nabla g(\theta_{1/2}).$$

As the following result shows, $2(\widehat{\theta}_1 - \theta_1)$ gives an $\mathcal{O}(\eta^3)$-accurate estimate of the local error.

**Lemma 1.** *Assume the function $g$ is twice differentiable. Then it holds*

$$\theta(\eta) - \theta_1 = 2(\widehat{\theta}_1 - \theta_1) + \mathcal{O}(\eta^3),$$

*i.e., the quantity $2\|\widehat{\theta}_1 - \theta_1\|$ gives an $\mathcal{O}(\eta^3)$-estimate of the local error generated by the GD step* (2.1).

*Proof.* From the Taylor expansion it follows that

$$\theta(\eta) - \theta_1 = \frac{\eta^2}{2} J_g(\theta_0)\nabla g(\theta_0) + \mathcal{O}(\eta^3). \tag{2.3}$$

Then, again by the Taylor expansion,

$$\begin{aligned}\widehat{\theta}_1 &= \theta_0 - \frac{\eta}{2}\nabla g(\theta_0) - \frac{\eta}{2}\nabla g\left(\theta_0 - \frac{\eta}{2}\nabla g(\theta_0)\right) \\ &= \theta_0 - \eta\nabla g(\theta_0) + \frac{\eta^2}{4} J_g(\theta_0)\nabla g(\theta_0) + \mathcal{O}(\eta^3).\end{aligned}$$

and therefore

$$\widehat{\theta}_1 - \theta_1 = \frac{\eta^2}{4} J_g(\theta_0)\nabla g(\theta_0) + \mathcal{O}(\eta^3). \tag{2.4}$$

The claim follows from (2.3) and (2.4). □

If at iteration $\ell$ we have the estimate

$$\mathrm{err}_\ell = \|\widehat{\theta}_{\ell+1} - \theta_{\ell+1}\|, \tag{2.5}$$

and if a local error of magnitude $\tau$ is desired, a simple mechanism for updating the step size is given by

$$\eta_{\ell+1} = \min\left(\max\left(\frac{\tau}{\mathrm{err}_\ell}, \alpha_{\min}\right), \alpha_{\max}\right) \cdot \eta_\ell,$$

where $\alpha_{\min} < 1$ and $\alpha_{\max} > 1$. In all our experiments we have used $\alpha_{\min} = 0.9$ and $\alpha_{\max} = 1.1$. In case $\nabla g$ has a large Lipschitz constant, a condition $\mathrm{err}_i < \tau$ for the update $\theta_{\ell+1} \leftarrow \theta_\ell$ gives a more stable algorithm. This procedure is described in Algorithm 1.

We apply Algorithm 1 to the differentially private SGD method. The challenge in the DP setting is that the gradients are blurred by the SGD approximation and the additive DP-noise.

**Algorithm 1** The update mechanism defined by the parameters $\alpha_{\min}$, $\alpha_{\max}$, $\tau$

Evaluate: $\mathrm{err}_\ell \leftarrow \|\widehat{\theta}_{\ell+1} - \theta_{\ell+1}\|$
**if** $\mathrm{err}_\ell > \tau$: **then**
  $\theta_{\ell+1} \leftarrow \theta_\ell$  (Discard)
**end if**
Update: $\eta_{\ell+1} = \min(\max(\frac{\tau}{\mathrm{err}_\ell}, \alpha_{\min}), \alpha_{\max}) \cdot \eta_\ell$.

## 3 Differential Privacy

We first recall some basic definitions of differential privacy (Dwork and Roth, 2014). We use the following notation. An input set containing $N$ data points is denoted as $X = (x_1, \ldots, x_N) \in \mathcal{X}^N$, where $x_i \in \mathcal{X}$, $1 \le i \le N$. To give the definition of the actual differential privacy we need the following definition.

**Definition 2.** *We say two datasets $X$ and $X'$ are neighbours if you get one by removing/adding an element from/to the other.*

We remark that the moments accountant method (Abadi et al., 2016) computes the privacy parameters in this neighbouring relation. The following definition formalises the $(\varepsilon, \delta)$-differential privacy of a randomised mechanism $\mathcal{M}$.

**Definition 3.** *Let $\varepsilon > 0$ and $\delta \in [0, 1]$. Mechanism $\mathcal{M} : \mathcal{X}^N \to \mathcal{R}$ is $(\varepsilon, \delta)$-DP if for every pair of neighbouring datasets $X$, $X'$ and every measurable set $E \subset \mathcal{R}$ we have*

$$\Pr(\mathcal{M}(X) \in E) \le \mathrm{e}^{\varepsilon} \Pr(\mathcal{M}(X') \in E) + \delta.$$

This definition is closed under post-processing which means that if a mechanism $\mathcal{A}$ is $(\varepsilon, \delta)$-differential private, then so is the mechanism $\mathcal{B} \circ \mathcal{A}$ for all functions $\mathcal{B}$ that do not depend on the data.

Assuming $X$ and $X'$ differ only by one record $x_i$, then by observing the outputs, the ability of an attacker to tell whether the output has resulted from $X$ or $X'$ remains bounded. Thus, the record $x_i$ is protected. As the record in which the two datasets differ is arbitrary, by definition, the protection applies for the whole dataset.

### 3.1 Moments accountant

We next recall some basic definitions and results concerning the moments accountant technique which is an important ingredient of our proposed method and crucial for obtaining $(\varepsilon, \delta)$-privacy bounds for the differentially private stochastic gradient descent. We refer to (Abadi et al., 2016) for more details.

**Definition 4.** *Let $\mathcal{M} : \mathcal{X}^N \to \mathcal{Y}$ be a randomised mechanism, and let $X$ and $X'$ be a pair of neighbouring datasets. Let* aux *denote any auxiliary input that does not depend on $X$ or $X'$. For an outcome $o \in \mathcal{Y}$, the privacy loss at $o$ is defined as*

$$c(o; \mathcal{M}, \mathrm{aux}, X, X') = \log \frac{\Pr(\mathcal{M}(\mathrm{aux}, X) = o)}{\Pr(\mathcal{M}(\mathrm{aux}, X') = o)}.$$

**Definition 5.** *$\lambda$th moment generating function $\alpha_{\mathcal{M}}(\lambda; \mathrm{aux}, X, X')$ is defined as*

$$\alpha_{\mathcal{M}}(\lambda; \mathrm{aux}, X, X') = \log \mathbb{E}_{o \sim \mathcal{M}(\mathrm{aux}, X)} \left(\exp(\lambda c(o; \mathcal{M}, \mathrm{aux}, X, X'))\right).$$

**Definition 6.** *Let $\mathcal{M} : \mathcal{X}^N \to \mathcal{Y}$ be a randomised mechanism, and let $X$ and $X'$ be a pair of neighbouring datasets. Let* aux *denote any auxiliary input that does not depend on $X$ or $X'$. The moments accountant with an integer parameter $\lambda$ is defined as*

$$\alpha_{\mathcal{M}}(\lambda) = \max_{\mathrm{aux}, X, X'} \alpha_{\mathcal{M}}(\lambda; \mathrm{aux}, X, X').$$

The privacy accounting of our proposed method is based on the composability theorem (Abadi et al., 2016, Thm. 2):

**Theorem 7.** *Suppose that $\mathcal{M}$ consists of a sequence of adaptive mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_k$, where $\mathcal{M}_i : \prod_{j=1}^{i-1} \mathcal{Y}_j \times \mathcal{X} \to \mathcal{Y}_i$, and $\mathcal{Y}_i$ is in the range of the $i$th mechanism, i.e., $\mathcal{M} = \mathcal{M}_k \circ \ldots \circ \mathcal{M}_1$. Then, for any $\lambda$*

$$\alpha_{\mathcal{M}}(\lambda) \le \sum_{i=1}^{k} \alpha_{\mathcal{M}_i}(\lambda), \tag{3.1}$$

*where the auxiliary input for $\alpha_{\mathcal{M}_i}(\lambda)$ is defined as all $\alpha_{\mathcal{M}_j}(\lambda)$'s outputs for $j < i$, and $\alpha_{\mathcal{M}}(\lambda)$ takes $\mathcal{M}_i$'s output, for $i < k$, as the auxiliary input.*

*Moreover, for any $\varepsilon > 0$, the mechanism $\mathcal{M}$ is $(\varepsilon, \delta)$-differentially private for*

$$\delta = \min_{\lambda} \exp(\alpha_{\mathcal{M}}(\lambda) - \lambda\varepsilon). \tag{3.2}$$

The inequality (3.1) gives an upper bound for the total moment $\alpha_{\mathcal{M}}(\lambda)$ of an iterative algorithm $\mathcal{M}$ if the moments $\alpha_{\mathcal{M}_i}(\lambda)$ of each iteration $i$ are known. Using (3.2), the privacy parameters $\varepsilon$ and $\delta$ can be approximated from numerically computed $\alpha_{\mathcal{M}}(\lambda)$-values.

### 3.2 Differentially private stochastic gradient descent

The objective is to find a minimum (with respect to $\theta$) of a loss function of the form

$$\mathcal{L}(\theta, X) = \frac{1}{N} \sum_{i=1}^{N} f(\theta, x_i).$$

At each step of the differentially private SGD, we compute the gradient $\nabla_\theta f(\theta, x_i)$ for a random minibatch $B$, clip the 2-norm of each gradient belonging to the minibatch, compute the average, add noise in order to protect privacy, and take a GD step using this noisy gradient. For a dataset $X$, the random mechanism to be analysed is then

$$\mathcal{M}(X) = \sum_{i \in B} \widetilde{\nabla} f(\theta, x_i) + \mathcal{N}(0, C^2\sigma^2 I), \qquad (3.3)$$

where $\widetilde{\nabla} f(\theta, x_i)$'s denote the gradients clipped with a constant $C > 0$, i.e., $\|\widetilde{\nabla} f(\theta, x_i)\| \leq C$ for all $i \in B$. The moments accountant method computes the privacy parameters for mechanisms of the form (3.3). We recall a result given by Abadi et al. (2016), which indicates how the privacy parameters $\varepsilon$ and $\delta$ scale for the mechanism (3.3) with respect to $q$ and $\sigma$.

**Theorem 8.** *There exists constants $c_1$ and $c_2$ so that given the sampling probability $q$ and the number of steps $T$, for any $\varepsilon < c_1 q^2 T$, DP-SGD is $(\varepsilon, \delta)$-differentially private for any $\delta > 0$ if we choose*

$$\sigma \geq c_2 \frac{q\sqrt{T \log(1/\delta)}}{\varepsilon}.$$

In experiments we compute the privacy parameters $\varepsilon$ and $\delta$ using the moments accountant method (Abadi et al., 2016) which gives the numerical $\alpha_{\mathcal{M}}(\lambda)$-values for the Poisson subsampling of minibatches, i.e., for the case where each data element is in the minibatch with a probability $q$.

## 4 Adaptive DP algorithm

The result of applying the learning rate adaptation to DP-SGD is depicted in Algorithm 2. We abbreviate this method as ADADP. Instead of (2.5), we use for the estimate the 2-norm of the function $\mathrm{err}(\theta, \widehat{\theta})$, where

$$\mathrm{err}(\theta, \widehat{\theta})_i = \frac{|\theta_i - \widehat{\theta}_i|}{\max(1, |\theta_i|)} \qquad (4.1)$$

as this was found to perform better numerically. In the neural network experiments the algorithm was found to be stable also without the acceptance condition $\mathrm{err}_i \leq \tau$ so we omit it there.

### 4.1 Privacy preserving properties of the method

By the very construction of Algorithm 2, we have the following result.

**Theorem 9.** *Let $q = |B|/N$, $\sigma \geq 1$ and $C > 0$. Let $\alpha_{\mathcal{M}}(\lambda)$ be the moments accountant of a mechanism $\mathcal{M}$ of the form (3.3) for these parameter values. Let $\widetilde{\mathcal{M}}$ denote the mechanism of Algorithm 2 using these parameter values. Then,*

$$\alpha_{\widetilde{\mathcal{M}}}(\lambda) \leq 2\alpha_{\mathcal{M}}(\lambda).$$

**Algorithm 2** ADADP update mechanism defined by the parameters $\alpha_{\min}$, $\alpha_{\max}$, $\tau$

Draw a batch $B_1$, with probability $q = |B|/N$.

Clip the gradients (with constant $C$) and evaluate at $\theta_\ell$:

$$G_1 \leftarrow \frac{1}{|B|}\left(\sum_{i \in B_1} \widetilde{\nabla} f_{\theta_\ell}(x_i) + \mathcal{N}(0, C^2\sigma^2 I)\right).$$

Take a step of size $\eta_\ell$:

$$\theta_{\ell+1} \leftarrow \theta_\ell - \eta_\ell G_1,$$

Take a step of size $\frac{\eta_\ell}{2}$:

$$\theta_{\ell+1/2} \leftarrow \theta_\ell - \frac{\eta_\ell}{2} G_1.$$

Draw a batch $B_2$, with probability $q = |B|/N$, clip the gradients (with constant $C$) and evaluate at $\theta_{\ell+1/2}$:

$$G_2 \leftarrow \frac{1}{|B|}\left(\sum_{i \in B_2} \widetilde{\nabla} f_{\theta_{\ell+1/2}}(x_i) + \mathcal{N}(0, C^2\sigma^2 I)\right).$$

Take a step of size $\frac{\eta_\ell}{2}$:

$$\widehat{\theta}_{\ell+1} \leftarrow \theta_{\ell+1/2} - \frac{\eta_\ell}{2} G_2.$$

Evaluate: $\mathrm{err}_\ell \leftarrow \|\mathrm{err}(\theta_{\ell+1}, \widehat{\theta}_{\ell+1})\|$

Update:
**if** $\mathrm{err}_\ell > \tau$ **then**
  $\theta_{\ell+1} \leftarrow \theta_\ell$  (Discard)
**end if**
$\eta_{\ell+1} \leftarrow \min\left(\max\left(\frac{\tau}{\mathrm{err}_\ell}, \alpha_{\min}\right), \alpha_{\max}\right) \cdot \eta_\ell.$

*Proof.* Looking at Alg. 2, we see that $\widetilde{\mathcal{M}}$ is a composition of two mechanisms, $\mathcal{M}_{G_1}(X)$ and $\mathcal{M}_{G_2}(X)$. The SGD approximations of these mechanisms are independent, both with the sampling ratio $q$, and the additive Gaussian noises are independent, both with the variance $C^2\sigma^2$. Thus, by Theorem 7, for all $\lambda > 0$,

$$\begin{aligned}\alpha_{\widetilde{\mathcal{M}}}(\lambda) &= \alpha_{\mathcal{M}_{G_1} \circ \mathcal{M}_{G_2}}(\lambda) \\ &\leq \alpha_{\mathcal{M}_{G_1}}(\lambda) + \alpha_{\mathcal{M}_{G_2}}(\lambda) = 2\alpha_{\mathcal{M}}(\lambda).\end{aligned}$$

□

By Theorem 9, using the same parameter values $q, \sigma$ and $C$, we are allowed to run Algorithm 2 half as many iterations as DP-SGD in order to have the same privacy for the data.

### 4.2 Choice of the tolerance parameter $\tau$

With the help of the following result we are able to choose the tolerance parameter $\tau$ such that the accumulated additive noise stays bounded.

**Theorem 10.** *Consider the intermediate variables $\theta_{\ell+1}$ and $\widehat{\theta}_{\ell+1}$ in Algorithm 2 (the results of one full step and two half steps). It holds*

$$\eta_\ell^2 \frac{dC^2\sigma^2}{2\,|B|^2} \leq \mathbb{E}\,\|\theta_{\ell+1} - \widehat{\theta}_{\ell+1}\|^2 \leq \eta_\ell^2 \frac{dC^2\sigma^2}{2\,|B|^2} + \eta_\ell^2 C^2,$$

*where the expectation is over the additive DP noise.*

*Proof.* From Algorithm 2 we see that

$$\begin{aligned}
&\|\theta_{\ell+1} - \widehat{\theta}_{\ell+1}\|^2 \\
&= \frac{\eta_\ell^2}{4\,|B|^2}\|g(\theta_\ell) - g(\theta_{\ell+1/2}) + X\|^2 \\
&= \frac{\eta_\ell^2}{4\,|B|^2}\Big(\|g(\theta_\ell) - g(\theta_{\ell+1/2})\|^2 \\
&\qquad + 2\,\langle\, g(\theta_\ell) - g(\theta_{\ell+1/2}), X\rangle + \|X\|^2\Big),
\end{aligned}$$

where

$$g(\theta_\ell) = \sum_{i\in B_1} \widetilde{\nabla} f_{\theta_\ell}(x_i), \quad g(\theta_{\ell+1/2}) = \sum_{i\in B_2} \widetilde{\nabla} f_{\theta_{\ell+1/2}}(x_i)$$

and $X \sim \mathcal{N}(0, 2C^2\sigma^2 I)$. As $\|g(\theta)\| \leq C\,|B|$, $\mathbb{E}X = 0$ and $\mathbb{E}\|X\|^2 = 2dC^2\sigma^2$, the claim follows. □

The parameter $\tau$ should be chosen such that, in addition to preventing instabilities caused by the SGD gradient, the algorithm should be keep the accumulated DP noise bounded. To this end, guided by Thm. 10, we set the parameter $\tau$ such that the accumulated DP noise is after $T$ iterations approximately $\mathcal{O}(1)$ elementwise.

Consider the situation where we apply DP-SGD with a step size sequence $\{\eta_\ell\}$. Then, after $T$ steps

$$\theta_T = \theta_0 - \frac{1}{|B|}\sum_{\ell=0}^{T-1} \eta_\ell g(\theta_\ell) + \mathcal{N}\left(0, \sum_{\ell=0}^{T-1} \eta_\ell^2 \frac{C^2\sigma^2}{|B|^2} I\right).$$

For simplicity assume $\mathrm{err}(\theta, \hat{\theta}) = \|\theta - \hat{\theta}\|$. Algorithm 2 then iteratively forces the estimate $\|\theta_{\ell+1} - \widehat{\theta}_{\ell+1}\|$ towards $\tau$. Setting $\|\theta_{\ell+1} - \widehat{\theta}_{\ell+1}\|$ to $\tau$ and assuming $\sqrt{d}\sigma \gg |B|$ we may approximate using Thm. 10

$$\eta_\ell^2 \frac{C^2\sigma^2}{|B|^2} \approx \frac{2\tau^2}{d}$$

Using this approximation, we see that the covariance of the additive noise after $T$ iterations is

$$\sum_{\ell=0}^{T-1} \eta_\ell^2 \frac{C^2\sigma^2}{|B|^2} I \approx \sum_{\ell=0}^{T-1} \frac{2\tau^2}{d} I = T\frac{2\tau^2}{d} I.$$

Setting the tolerance parameter

$$\tau = \sqrt{\frac{d}{2T}},$$

the covariance becomes $I$.

In our experiments with neural networks $\frac{d}{2T} = \mathcal{O}(1)$ and we use $\tau = 1.0$. In the small dimensional experiments for Gaussian mixture models, $\frac{d}{2T} = \mathcal{O}(10^{-2})$ and we use $\tau = 0.1$. In small dimensional examples also the Lipschitz constant of $g$ affects more the suitable step size $\eta_\ell$. We found this to imply a sharp transition from stable to unstable behaviour as the step size grows and therefore the acceptance condition of Alg. 1 was needed in the small dimensional experiments. In the neural network experiments the condition was omitted.

## 5 Experiments

We compare ADADP with DP-SGD and with Adam combined with DP gradients (DP-Adam). The Poisson subsampling of minibatches is approximated as by Abadi et al. (2016), i.e., by randomly permuting the data elements and then partitioning them into minibatches of a fixed size. Algorithm 2 needs two minibatches per iteration: one to compute the vector $G_1$ and then the other one to compute $G_2$. Therefore, in one epoch we run $\frac{N}{2|B|}$ iterations. Then the number of gradient evaluations per epoch is the same as for SGD and Adam and thus the computation times are essentially equivalent. When using ADADP, the per epoch privacy cost is then also the same for all the methods considered, for fixed values of the parameters $q$ and $\sigma$.

In the DP setting the methods are compared by measuring the test accuracy for a given $\varepsilon$-value, when $\delta = 10^{-6}$. The $\varepsilon$-values are computed using the moments accountant (Abadi et al., 2016). In the neural network experiments we use all the methods with minibatch size $|B| = 200$ and run each method for 100 epochs. The initial learning rate for ADADP is set to $10^{-1}$, but the results are quite insensitive to this value as the algorithm will converge to the desired learning rate already during the first epoch.

The values $\alpha_{\min} = 0.9$ and $\alpha_{\max} = 1.1$ were used in all experiments. In the neural network experiments we use the value $\tau = 1.0$ and in the small scale experiment the value $\tau = 0.1$ (see Sec. 4.2).

All experiments are implemented using PyTorch. Code for the experiments is provided in the supplementary material. In the supplementary material we also demonstrate that the ADADP method can help stabilise federated learning in case the data are non-uniformly distributed to different clients.

### 5.1 Datasets and test architectures for the neural networks

We compare the methods on two standard datasets: MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky and Hinton, 2009).

In MNIST each example is a $28 \times 28$ size gray-level image. The training set contains 60000 and the test set 10000 examples. For MNIST we use a feedforward neural network with 2 hidden layers with 256 hidden units. As a result, the total number of parameters for this network is 334336. We use ReLU units and the last layer is passed to softmax of 10 classes with cross-entropy loss. Without additional noise ($\sigma = 0$) we reach an accuracy of around 96%.

CIFAR-10 consists of colour images classified into 10 classes. The training set contains 50000 and the test set 10000 examples. Each example is a $32 \times 32$ image with three RGB channels. The CIFAR-100 dataset has similar images classified into 100 classes. For CIFAR-10 we use a neural network, which consists of two convolutional layers followed by three fully connected layers. The convolutional layers use $3 \times 3$ convolutions with stride 1, followed by ReLU and max pools, with 64 channels each. The output of the second convolutional layer is flattened into a vector of dimension 1600. The fully connected layers have 500 hidden units. Last layer is passed to softmax of 10 classes with cross-entropy loss. The total number of parameters for this network is about $10^6$. Similarly to the experiments of (Abadi et al., 2016), in the DP setting we pre-train the convolutional layers using the CIFAR-100 dataset and the differentially private optimisation is carried out only for the fully connected layers.

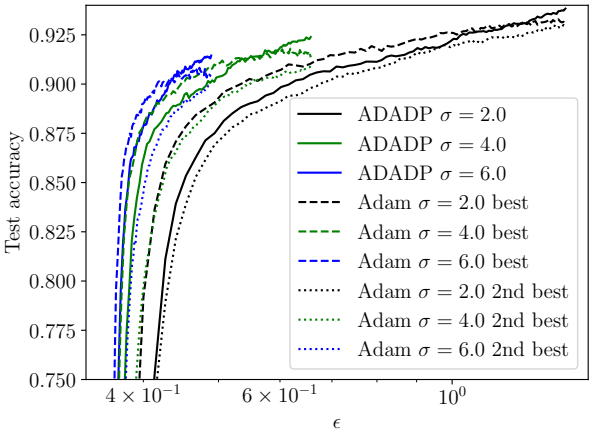
### 5.2 Comparison of ADADP and DP-Adam

We first compare ADADP with optimally tuned DP-Adam. This means that in each case we search the best and the second best initial learning rate $\eta_0$ for Adam on a grid $\{\ldots, 10^{-4.5}, 10^{-4}, 10^{-3.5}, \ldots\}$. We apply ADADP for 50 steps, then fix the learning rate (denoted $\eta_{50}$) and apply DP-SGD with the decaying learning rate

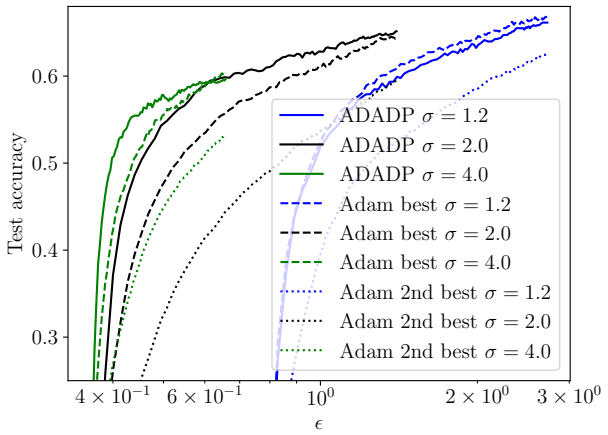
$$\eta_k = \frac{\eta_{50}}{1 + 0.1 \cdot (k - 50)}, \tag{5.1}$$

where $k$ denotes the number of epoch ($k > 50$).

As Figure 1a illustrates, in case of MNIST and the feedforward network, ADADP is competitive with the learning rate optimised Adam and gives better results than Adam with the second best learning rate found from the grid. We see from Figure 1b, that in the case of CIFAR-10 and convolutional network, ADADP is again competitive with the learning rate optimised Adam.




(a) MNIST




(b) CIFAR-10

Figure 1: DP learning using ADADP and Adam with optimal and nearly optimal initial learning rate $\eta_0$ sought from the grid $\{\ldots, 10^{-4.5}, 10^{-4}, 10^{-3.5}, \ldots\}$ for each $\sigma$.

### 5.3 Comparison against the private selection algorithm by Liu and Talwar

We next compare ADADP to the private selection algorithm given by Liu and Talwar (2019, Alg. 2) (see Figure 2). The private algorithm runs constant learning rate DP-SGD for 50 epochs and then decays as

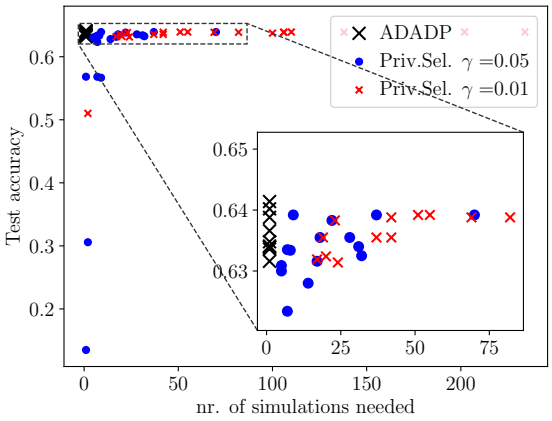



Figure 2: Realisations of ADADP and the private selection algorithm (Liu and Talwar, 2019, Alg. 2). Each run of ADADP requires one training of 100 epochs. The private selection algorithm needs the more training runs the smaller the parameter $\gamma$. Both methods have the same $(\varepsilon, \delta)$-privacy for the training dataset.

(5.1). The private candidates for the learning rates are $\eta_i = 10^{-i/2}$, $i = 1, \ldots, 10$. The private selection algorithm has a random stopping time and $\gamma$ denotes the probability of halting the iteration and accepting the best candidate found so far. Table 1 shows the mean and standard deviation of the test accuracy of the resulting model for different values of $\gamma$ (a table with more results is given in the supplementary material). We use the CIFAR-10 test set as a validation set for the private selection algorithm and add Laplace noise to the score function such that the same privacy is provided for the validation set. Notice that ADADP does not need a validation set, there is exposure only for the training data.

| | Mean acc. | Std acc. | Mean evals |
|---|---|---|---|
| ADADP | 0.6349 | 0.0033 | 1 |
| Priv. $\gamma = 2^{-8}$ | 0.6368 | 0.0113 | 224.30 |
| Priv. $\gamma = 2^{-7}$ | 0.6317 | 0.0399 | 131.76 |
| Priv. $\gamma = 2^{-4}$ | 0.6053 | 0.0662 | 14.79 |
| Priv. $\gamma = 2^{-2}$ | 0.5467 | 0.1278 | 3.42 |
| Priv. $\gamma = 2^{-1}$ | 0.4728 | 0.1688 | 2.04 |

Table 1: Comparison of ADADP and the private selection algorithm (Liu and Talwar, 2019, Alg. 2) for different values of the parameter $\gamma$. 'Mean evals' denotes the mean of the number of training runs (100 epochs each) needed for one evaluation of each algorithm. 'Mean acc.' and 'Std acc.' denote the mean and standard deviation of the test accuracy of the resulting model, respectively.

### 5.4 Comparison of ADADP and DP-SGD

In the next experiment we illustrate the benefits of ADADP when compared to the plain DP-SGD. We search an optimal learning rate for DP-SGD on a grid $\{\ldots, 10^{-2.5}, 10^{-2.0}, 10^{-1.5}, \ldots\}$ in the case $\sigma = 2.0$. Using this learning rate for DP-SGD, we compare the performance of DP-SGD and ADADP when $\sigma = 4.0$, 6.0 and 8.0. As we see from Figures 3 and 4, ADADP finds an appropriate learning rate and gives better results than DP-SGD for these values of $\sigma$.

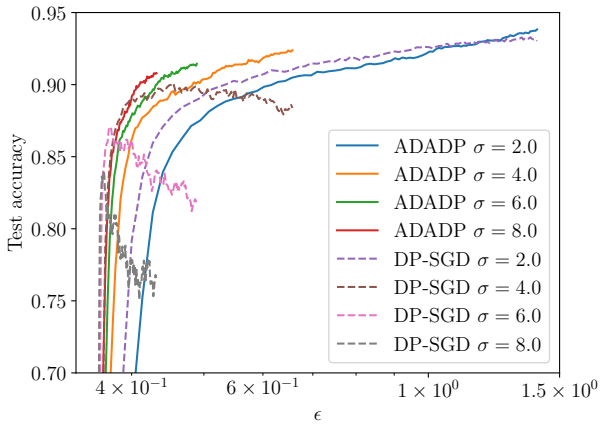



Figure 3: ADADP and DP-SGD for MNIST. The fixed learning rate $\eta$ of SGD is tuned in the $\sigma = 2.0$-case using the grid $\{\ldots, 10^{-2.5}, 10^{-2.0}, 10^{-1.5}, \ldots\}$.

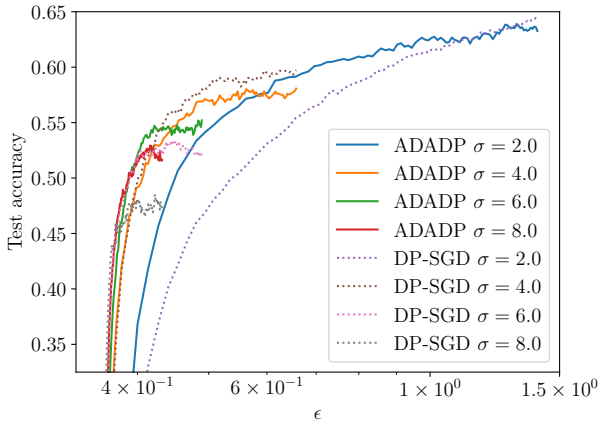



Figure 4: ADADP and DP-SGD for CIFAR-10. The fixed learning rate $\eta$ of SGD is tuned in the $\sigma = 2.0$-case using the grid $\{\ldots, 10^{-2.5}, 10^{-2.0}, 10^{-1.5}, \ldots\}$.

### 5.5 Gaussian mixture model

We also compare the performance of ADADP to a differentially private variational inference technique

called DPVI (Jälkö et al., 2017). We test the methods on a Gaussian mixture model, which for $K$ components is given by $\pi_k \sim \mathrm{Dir}(\alpha)$, $\mu^{(k)} \sim \mathcal{N}(0, I)$ and $\sigma^{(k)} \sim \mathrm{Inv-Gamma}(1, 1)$, $1 \leq k \leq K$, and for which the likelihood is given by

$$p(x_i|\pi, \mu, \sigma) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x_i; \mu^{(k)}, \sigma^{(k)} I).$$

The loss function is then given by

$$\mathcal{L}(q_\xi) = \sum_{i=1}^{n} \mathcal{L}_i(q_\xi),$$

where

$$\mathcal{L}_i(q_\xi) = \langle \ln p(x_i|\theta) \rangle_{q_\xi(\theta)} - \frac{1}{N} \mathrm{KL}(q||p).$$

For the description of the posterior $q$ and rest of the details we refer to (Jälkö et al., 2017). We consider synthetic training data of $n = 1000$ samples drawn from $K = 5$ two-dimensional Gaussian distributions centred at 0 and $[\pm 1, \pm 1]$ with covariances $I$. The test data consist of 100 samples drawn from the same distribution. This model has been used also by Honkela et al. (2010).

We set the batch size $|B| = 20$, clipping constant $C = 1$, and run the algorithms for 3000 and 8000 iterations, when $\sigma = 1.2$ and $\sigma = 6.0$, respectively. We run DPVI for three best learning rates found from the grid $\{\ldots, 10^{-2.5}, 10^{-2.0}, 10^{-1.5}, \ldots\}$. Because of the low dimensionality, we use here the tolerance parameter $\tau = 0.1$ (see Sec. 4.2). We apply a learning rate decay similar to (5.1) for both methods.

Figure 5 shows likelihoods of the test data as the learning progresses for $\sigma = 6.0$ (results for $\sigma = 1.2$ given in the supplementary material). We see that ADADP is competitive with the optimally tuned DPVI. Figure 6 illustrates how ADADP finds the suitable learning rate for different initial learning rates $\eta_0$.

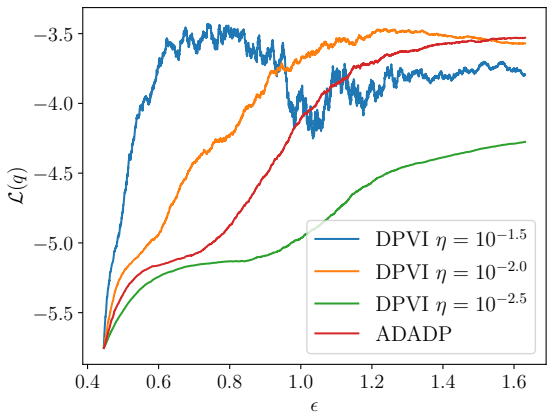



Figure 5: Log-likelihood $\mathcal{L}(q_\xi)$ for ADADP and for DP-SGD for different learning rates $\eta$, when $\sigma = 6.0$. Values of $\varepsilon$ are for $\delta = 10^{-6}$.

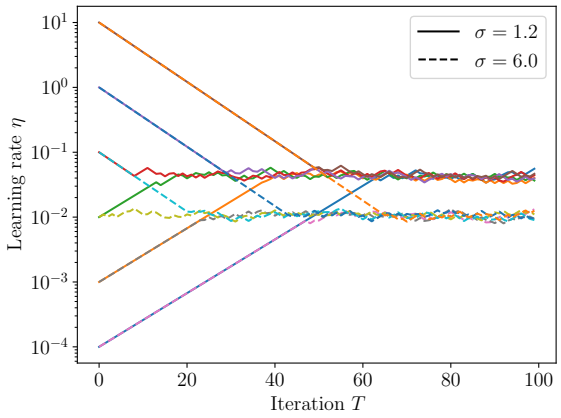



Figure 6: Learning rate found by ADADP for different initial learning rates $\eta_0$, when $\sigma = 1.2$ and $\sigma = 6.0$. Here 50 iterations equals one epoch of training.

## 6 Conclusions

We have proposed the first learning rate adaptive DP-SGD method. By simple derivations, we have shown how to determine the additional tolerance hyperparameter in the algorithm. Based on this, we developed a rule for selecting the parameter and verified the efficiency of the resulting algorithm in a number of diverse learning problems. The results show that our approach is competitive in performance with commonly used optimisation methods even without any tuning. Comparisons to the DP meta selection algorithm further illustrate the benefits of our approach. Overall, our work takes an important step towards truly DP and automated learning for SGD-based algorithms.

As future work, it would be useful to develop a better understanding of the tolerance hyperparameter. Furthermore, it would be important to study the adaptation of other key algorithmic parameters of DP-SGD, such as the gradient clipping threshold and the minibatch size. Adaptive clipping in the federated learning setting is considered by Thakkar et al. (2019). Balles et al. (2017) provide an interesting non-private implementation of minibatch adaptation, but unfortunately the approach cannot easily be applied in the DP case. One interesting alternative is given by the adaptive momentum approach (Wang et al., 2015) based on the SGNHT algorithm (Ding et al., 2014).

## References


Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *Proc. CCS 2016*.

Balles, L., Romero, J., and Hennig, P. (2017). Coupling adaptive batch sizes with learning rates. In *Proc. UAI 2017*.

Chaudhuri, K. and Monteleoni, C. (2008). Privacy-preserving logistic regression. In *Adv. Neural Inf. Process. Syst. 21*, pages 289–296.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. (2011). Differentially private empirical risk minimization. *J. Mach. Learn. Res.*, 12:1069–1109.

Dimitrakakis, C., Nelson, B., Mitrokotsa, A., and Rubinstein, B. I. P. (2014). Robust and private Bayesian inference. In *Proc. ALT 2014*, pages 291–305. Springer Science + Business Media.

Ding, N., Fang, Y., Babbush, R., Chen, C., Skeel, R. D., and Neven, H. (2014). Bayesian sampling using stochastic gradient thermostats. In *Advances in neural information processing systems*, pages 3203–3211.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Proc. TCC 2006*, pages 265–284. Springer Berlin Heidelberg.

Dwork, C. and Roth, A. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407.

Dwork, C., Talwar, K., Thakurta, A., and Zhang, L. (2014). Analyze gauss: Optimal bounds for privacy-preserving principal component analysis. In *Proc. STOC 2014*, pages 11–20.

Hairer, E., Norsett, S., and Wanner, G. (1987). *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Computational Mathematics*. Springer, Berlin.

Honkela, A., Raiko, T., Kuusela, M., Tornio, M., and Karhunen, J. (2010). Approximate Riemannian conjugate gradient learning for fixed-form variational Bayes. *J Mach Learn Res*, 11:3235–3268.

Jälkö, J., Honkela, A., and Dikmen, O. (2017). Differentially private variational inference for non-conjugate models. In *Proc. UAI 2017*.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proc. ICLR 2015*.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. *Technical Report. University of Toronto*.

Kusner, M. J., Gardner, J. R., Garnett, R., and Weinberger, K. Q. (2015). Differentially private Bayesian optimization. In *Proc. ICML 2015*, pages 918–927.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Li, B., Chen, C., Liu, H., and Carin, L. (2019). On connecting stochastic gradient MCMC and differential privacy. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 557–566.

Liu, J. and Talwar, K. (2019). Private selection from private candidates. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 298–309. ACM.

Park, M., Foulds, J., Chaudhuri, K., and Welling, M. (2016). Variational Bayes in private settings (VIPS). arXiv:1611.00340 [stat.ML].

Rajkumar, A. and Agarwal, S. (2012). A differentially private stochastic gradient descent algorithm for multiparty classification. In *Proc. AISTATS 2012*, pages 933–941.

Song, S., Chaudhuri, K., and Sarwate, A. D. (2013). Stochastic gradient descent with differentially private updates. In *Proc. GlobalSIP 2013*, pages 245–248.

Strubell, E., Ganesh, A., and McCallum, A. (2019). Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Thakkar, O., Andrew, G., and McMahan, H. B. (2019). Differentially private learning with adaptive clipping. *arXiv preprint arXiv:1905.03871*.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Wang, Y., Fienberg, S. E., and Smola, A. J. (2015). Privacy for free: Posterior sampling and stochastic gradient Monte Carlo. In *Proc. ICML 2015*, pages 2493–2502.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. In *Adv. Neural Inf. Process. Syst. 30*, pages 4148–4158. Curran Associates, Inc.

Zhang, Z., Rubinstein, B., and Dimitrakakis, C. (2016). On the differential privacy of Bayesian inference. In *Proc. Conf. AAAI Artif. Intell. 2016*.