OPT2024: 16th Annual Workshop on Optimization for Machine Learning

DiSK: Differentially Private Optimizer with Simplified Kalman Filter for Noise Reduction

Xinwei Zhang[*], Zhiqi Bu, Borja Balle, Mingyi Hong, Meisam Razaviyayn[*], Vahab Mirrokni

Abstract

Differentially private optimizers have been widely used to train modern machine learning models while protecting the privacy of training data. A popular approach to privatize an optimizer is to clip the individual gradients and add sufficiently large noise to the clipped gradient. However, a significant performance drop is observed when these optimizers are applied to large-scale model (pre-)training. This degradation stems from the substantial noise injection required to maintain DP, which disrupts the optimizer's dynamics. This paper introduces DiSK, a novel framework designed to significantly enhance the performance of differentially private (DP) optimizers. DiSK employs Kalman filtering, a technique drawn from control and signal processing, to effectively denoise privatized gradients and generate progressively refined gradient estimations. To ensure practicality for large-scale training, we simplify the Kalman filtering process, minimizing its memory and computational demands. We establish theoretical privacy-utility trade-off guarantees for DiSK, and demonstrating provable improvements over standard DP optimizers like DPSGD. Extensive experiments across diverse tasks, including vision tasks such as CIFAR-100 and ImageNet-1k and language fine-tuning tasks such as GLUE, E2E, and DART, validate the effectiveness of DiSK. The results showcase its ability to significantly improve the performance of DP optimizers, surpassing state-of-the-art results under the same privacy constraints.

1. Introduction

In this paper, we aim to solve the empirical risk minimization (ERM) problem with the differential privacy (DP) guarantee.

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}), F(\mathbf{x}) = \frac{1}{N}\sum_{\xi\in\mathcal{D}} f(\mathbf{x};\xi), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the optimization variable, $\mathcal{D}$ is the training dataset with $|\mathcal{D}| = N$ samples $\xi_i, i \in \{1,\dots,N\}$, and $f(\cdot)$ denotes the (possibly non-convex) loss function parameterized by $\mathbf{x}$ and evaluated on sample ξ. The definition of DP is proposed by [16]:

Definition 1 **((ϵ,δ)-DP [16])** *A randomized mechanism $\mathcal{M}$ is said to be (ϵ,δ)-differentially private, if for any two neighboring datasets $\mathcal{D},\mathcal{D}'$ ($\mathcal{D},\mathcal{D}'$ differ only by one sample) and for any measurable output set $\mathcal{S}$, it holds that* $\Pr[\mathcal{M}(\mathcal{D})\in\mathcal{S}] \le e^{\epsilon}\Pr[\mathcal{M}(\mathcal{D}')\in\mathcal{S}] + \delta$.

The popular practical differentially private approaches to finding an (approximate) solution to the ERM problem (1) are Differentially Private Stochastic Gradient Descent (DPSGD) [1] and its variants, including DP-Adam and DP-Lora [46]. To protect DP, DPSGD applies the commonly used Gaussian mechanism [1, 16] to privatize the mini-batch gradient at each iteration of the SGD optimizer. The Gaussian mechanism provides a privacy guarantee by injecting a large enough Gaussian noise into the algorithms' output.

[*] xinweiz,razaviya@usc.edu

© .

The DPSGD algorithm, presented in Algorithm 1, first samples a mini-batch $\mathcal{B}^t$ of size B and computes the per-sample gradient at each step t. Then, it applies the Gaussian mechanism by clipping the per-sample gradient and injecting DP noise. The clipping operation bounds the sensitivity of the stochastic gradients to C, e.g., $\text{clip}\left(\nabla f, C\right) = \min\left\{1, \frac{C}{\|\nabla f\|}\right\}\nabla f$ or $\frac{C}{\|\nabla f\|}\nabla f$. Finally, the algorithm updates the model parameter with the privatized mini-batch gradient. It has been shown that DPSGD guarantees (ϵ, δ)-DP with sufficiently large injected noise [1].

Algorithm 1 DPSGD algorithm

```
Input: $\mathbf{x}_0, \mathcal{D}, C, \eta, \sigma_{\text{DP}}$
for $t = 0, \ldots, T-1$ do
    Uniformly draw minibatch $\mathcal{B}_t$ from $\mathcal{D}$
    $\mathbf{g}_t = \frac{1}{B}\sum_{\xi_i \in \mathcal{B}_t} \text{clip}\left(\nabla f(\mathbf{x}_t; \xi_i), C\right) + \mathbf{w}_t$
        where $\mathbf{w}_t \sim \mathcal{N}(0, \sigma_{\text{DP}}^2 \cdot \mathbf{I}_d)$
    $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \mathbf{g}_t$,
end for
```

Theorem 2 (Privacy Guarantee [1]) *Given N, B, T and C, there exist positive constants u, v, such that for any $\epsilon < \frac{uB^2T}{N^2}, \delta > 0$, by choosing $\sigma_{\text{DP}}^2 \geq v\frac{C^2 T \ln(\frac{1}{\delta})}{N^2\epsilon^2}$, Algorithm 1 is guaranteed to be (ϵ, δ)-DP.*

Theorem 2 implies that the DP noise variance $\mathbb{E}[\|\mathbf{w}_t\|^2] \geq d\sigma_{\text{DP}}^2$ is proportional to the number of iterations T and the number of trainable model parameters d. When training small machine learning models, e.g., regression and multi-layer perceptron, or fine-tuning partial model parameters for large models, T or d or both are relatively small. Therefore, the impact of DP is not significant. However, in modern deep learning, the model size d is huge and requires many pre-training steps T. Therefore, the injected DP noise can be large in pre-training tasks or fine-tuning large foundation models, which causes a significant performance drop.

1.1. Contributions

In this paper, we adopt the **Kalman filter**, a commonly used tool for accurately estimating signals from a series of noisy observations in the control theory, to improve the quality of the gradient estimates and the performance of DP optimizers. The main idea of our approach is to construct the gradient dynamic using Hessian information and view the privatized gradient as a noisy observation of the true gradient. With the gradient dynamics and the noisy observation, we apply the Kalman filter to obtain an accurate estimation of the true gradient. With an improved estimate of the gradient, the performance of DP optimizers can be improved. Due to the inefficiency of the Kalman filter, we apply a series of simplifications to the algorithm to reduce its extra memory and computational cost. We summarize our contribution as follows:

- **Algorithm Design:** We introduce DiSK, a novel Kalman filter-based approach designed to mitigate DP noise and enhance the performance of various DP optimizers.
- **Algorithm Simplification:** We simplify the Kalman filtering process to significantly reduce memory and computational overhead. This simplification requires only one additional forward step and two extra optimizer states.
- **Theoretical Analysis:** We provide theoretical analyses of DiSK, demonstrating that with careful hyperparameter selection, our approach provably improves convergence by a constant factor .
- **Numerical Results:** Extensive experiments across various models, datasets, and optimizers demonstrate that DiSK significantly boosts DP training performance. Specifically, under the same privacy

budgets, DiSK exhibits substantial improvements in test accuracy for training-from-scratch scenarios: a notable increase from 32.4% to 36.9% on the ImageNet-1k dataset, a considerable rise from 63% to 75% on CIFAR-10, and a remarkable improvement from 21% to 42% on CIFAR-100. Furthermore, in fine-tuning tasks, DiSK demonstrates remarkable improvements: an increase from from 85% to 89% on CIFAR-100 and an improvement from 81% to 86% on the GLUE dataset. *These results surpass state-of-the-art DP training performance under the same privacy guarantees.*

2. Algorithm design

In this section, we propose a general **Di**fferentially Private Optimization with **S**implified **K**alman Filter for Noise reduction (DiSK) approach. The approach utilizes the gradient dynamic and applies the Kalman filter to accurately estimate the true gradient from the noisy privatized gradient. We simplify the Kalman filter for memory and computation-efficient training for modern deep learning applications.

2.1. Gradient dynamic and Kalman filter

The Kalman filter estimates an unknown variable iteratively given a series of measurements over time [43], see Appendix A.2 for an introduction to the Kalman filter. To apply the Kalman filter, we construct the gradient's system dynamics consisting of the **system update** and the **observation**.

The **system update** of the gradient can be derived as follows: By Taylor expansion, the change of the gradient at iteration t can be expressed as:

$$\begin{aligned}\nabla F(\mathbf{x}_t) &= \nabla F(\mathbf{x}_{t-1}) + \nabla^2 F(\mathbf{x}_{t-1})^\top(\mathbf{x}_t - \mathbf{x}_{t-1}) + R(\mathbf{x}_t)\\ &= \nabla F(\mathbf{x}_{t-1}) + \mathbf{H}_t(\mathbf{x}_t - \mathbf{x}_{t-1}) + \mathbf{v}_t,\end{aligned} \tag{2}$$

where $\mathbf{H}_t \in \mathbb{R}^{d\times d} := \nabla^2 F(\mathbf{x}_{t-1})$ and $\mathbf{v}_t = \frac{1}{2}\int_0^1 (z\mathbf{x}_t + (1-z)\mathbf{x}_{t-1})^\top \nabla^3 F(\mathbf{x}_{t-1})(z\mathbf{x}_t + (1-z)\mathbf{x}_{t-1})\mathrm{d}z$ is the remainder. The **observation** of the system is defined as the privatized gradient $\mathbf{g}_t$, which is a stochastic oracle of the true gradient:

$$\mathbf{g}_t = \frac{1}{B}\sum_{\xi\in\mathcal{B}} \mathrm{clip}\left(\nabla f(\mathbf{x},\xi), C\right) + \mathbf{w}_t = \mathbf{C}_t \nabla F(\mathbf{x}_t) + \mathbf{w}'_t, \tag{3}$$

where $\mathbf{C}_t$ is a (noisy) observation matrix, and $\mathbf{w}'_t$ is the observation noise containing the DP noise and the sub-sampling noise. Note that if the clipping operation is inactive, then $\mathbf{C}_t = \mathbf{I}_d$

Combining the system update (2) and the observation (3), the gradient dynamics is:

$$\begin{aligned}\nabla F(\mathbf{x}_t) &= \nabla F(\mathbf{x}_{t-1}) + \mathbf{H}_t(\mathbf{x}_t - \mathbf{x}_{t-1}) + \mathbf{v}_t, && \textit{(System update)}\\ \mathbf{g}_t &= \mathbf{C}_t \nabla F(\mathbf{x}_t) + \mathbf{w}_t, && \textit{(Observation)}\end{aligned} \tag{4}$$

Note that $\mathbf{H}_t$ is the (time-varying) Hessian matrix; $\mathbf{v}_t$ is a random variable that models the remainder; $\mathbf{C}_t, \mathbf{w}_t$ are random variables that model multiplicative and additive observation noise in the privatized gradient. If $\mathbf{H}_t$ is known or can be estimated accurately, then we can apply the Kalman filter that combines the **system update** and the **observation** of the gradient to improve the overall estimation quality of the actual gradient beyond only using the observation $\mathbf{g}_t$. However, since it is hard to compute the Hessian matrix in large-scale training, we further assume that $\mathbf{H}_t$ is also a random matrix that can only be approximated.

Algorithm 2 DiSK

```
1: Input: $\mathbf{x}_0, \mathcal{D}, \eta, \gamma, \kappa, C, \sigma_{\mathrm{DP}}$
2: Initialize: $\tilde{\mathbf{g}}_{-1} = \mathbf{g}_0, \mathbf{d}_{-1} = 0$
3: for $t = 0, \dots, T-1$ do
4:     Randomly draw minibatch $\mathcal{B}_t$ from $\mathcal{D}$
5:     Compute $\mathbf{g}_t = \frac{1}{B}\sum_{\xi\in\mathcal{B}_t} \mathrm{clip}\left(\frac{1-\kappa}{\kappa\gamma}\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) + (1-\frac{1-\kappa}{\kappa\gamma})\nabla f(\mathbf{x}_t;\xi), C\right) + \mathbf{w}_t$
           where $\mathbf{w}_t \sim \mathcal{N}(0, \sigma_{\mathrm{DP}}^2 \cdot \mathbf{I}_d)$
6:     $\tilde{\mathbf{g}}_t = (1-\kappa)\tilde{\mathbf{g}}_{t-1} + \kappa\mathbf{g}_t$                # Apply filter
7:     $\mathbf{x}_{t+1} = \mathrm{OptimizerUpdate}(\mathbf{x}_t, \eta, \tilde{\mathbf{g}}_t)$      # Parameter update
8:     $\mathbf{d}_t = \mathbf{x}_{t+1} - \mathbf{x}_t$                         # Record update direction
9: end for
```

When $\mathbf{C}_t$ and $\mathbf{H}_t$ are random, the Kalman filter for the above system (4) is [45]:

$$\begin{aligned}
\tilde{\mathbf{g}}_{t|t-1} &= \tilde{\mathbf{g}}_{t-1} + \tilde{\mathbf{H}}_t(\mathbf{x}_t - \mathbf{x}_{t-1}) && \textit{(Prediction)}\\
\mathbf{P}_{t|t-1} &= \mathbf{P}_{t-1} + \Sigma_{\mathbf{H},t} + \Sigma_{\mathbf{v},t}\\
\mathbf{K}_t &= \mathbf{P}_{t|t-1}\,\mathbb{E}[\mathbf{C}_t](\Sigma_{\mathbf{w}_t} + \mathbb{E}[\mathbf{C}_t](\Sigma_{\mathbf{C},t}\mathbf{S}_t + \mathbf{P}_{t|t-1})\,\mathbb{E}[\mathbf{C}_t]^\top - \Sigma_{\mathbf{H},t})^{-1} && (5)\\
\tilde{\mathbf{g}}_t &= \tilde{\mathbf{g}}_{t|t-1} + \mathbf{K}_t(\mathbf{g}_t - \mathbb{E}[\mathbf{C}_t]\tilde{\mathbf{g}}_{t|t-1}) && \textit{(Correction)}\\
\mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t\,\mathbb{E}[\mathbf{C}_t])\mathbf{P}_{t|t-1}, \quad \mathbf{S}_t = \mathbb{E}[\tilde{\mathbf{g}}_t\tilde{\mathbf{g}}_t^\top],
\end{aligned}$$

where $\mathbf{P}$ denotes the covariance matrix of $\tilde{\mathbf{g}}$, Σ denotes the covariance matrices of the random variables, and $\tilde{\mathbf{H}}_t$ is an instantiation/observation of the unknown Hessian matrix $\mathbf{H}_t$. Instead of directly using $\mathbf{g}_t$ with large variance, the output of the Kalman filter $\tilde{\mathbf{g}}_t$ has a smaller variance and therefore improves the performance of DP optimizers.

2.2. Algorithm simplification

Equations (5) is the general form of a Kalman filter for arbitrary optimizers and is generally hard to implement. To obtain a memory-efficient, implementable algorithm, we make the following simplification. Detailed simplification steps is given in Appendix A.4.

1. Use constant $\mathbf{C} = \mathbf{I}_d$.
2. Use the finite difference of stochastic gradients to approximate $\mathbf{H}\mathbf{d}_{t-1}$, i.e.,

$$\mathbf{H}_t\mathbf{d}_{t-1} = \frac{\nabla F(\mathbf{x}_t + \gamma\mathbf{d}_{t-1}) - \nabla F(\mathbf{x}_t)}{\gamma} + \Delta(\gamma) \approx \frac{1}{B}\sum_{\xi\in\mathcal{B}} \frac{\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma}.$$

3. Simplify the covariance matrix $\Sigma_{\mathbf{H}} = \sigma_H^2\mathbf{I}_d$, and $\mathbf{P}_t = p_t\mathbf{I}_d, \mathbf{K}_t = k_t\mathbf{I}_d$.
4. Simplify the Kalman gain to a time-invariant constant, i.e., $k_t = \kappa, \forall t$.

With the above simplification, (5) simplifies to Algorithm 2. In the simplified algorithm, we compute and privatize the linear combination of the gradient evaluated at two points, $\mathbf{x}_t, \mathbf{x}_t + \gamma\mathbf{d}_t$. Then $\tilde{\mathbf{g}}_t$ is an exponential weighted averaging of the privatized gradient, which serves as the input to the base optimizer (Line 10).

Memory and computation cost: Algorithm 2 requires one additional forward step to compute $\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi)$ and two additional buffers to store $\tilde{\mathbf{g}}_t$ and $\mathbf{d}_t$. It would, at most, double the computational and memory costs compared with the base algorithm.

2.3. Theoretical analysis

2.3.1. Convergence analysis

We provide the following convergence results for Algorithm 2, assuming σ_{DP} being a constant. For the optimal choice of γ for evaluating the Hessian matrix, we have:

Theorem 3 *Assume $f(\cdot;\xi)$ in (1) is G-Lipschitz and L-smooth for all $\xi \in \mathcal{D}$, and the stochastic gradient variance is bounded by σ_{SGD}^2. For any σ_{DP}^2, choosing $C \geq (1+\frac{2(1-\kappa)}{\kappa})G$, κ, η satisfy $\eta < \frac{1+\kappa}{2L(1+2(1-\kappa)^2\beta L(2+|1+\gamma|C_\gamma))}$, $\kappa > 1 - \frac{1}{\sqrt{1+4\eta^2L^2+|1+\gamma|(\kappa+2\eta^2L^2C_\gamma)}}$, and run Algorithm 2 for T iterations. Then,*

$$\frac{1}{T}\sum_{t=0}^{T}\mathbb{E}\left\|\nabla F(\mathbf{x}_t)\right\|^2 \leq \frac{2(F(\mathbf{x}_0)+\beta\left\|\nabla F(\mathbf{x}_0)\right\|^2 - F^\star)}{C_1\eta T} + \frac{2(\beta+\eta^2L)\kappa^2}{C_1\eta}\left(\frac{(2+|1+\gamma|)\sigma_{\text{SGD}}^2}{B} + d\sigma_{\text{DP}}^2\right), \tag{6}$$

where $C_\gamma = 1+\frac{4(2+1/\kappa+|1+\gamma|)}{(1-\kappa)^2}$, $C_1 = (1+\kappa-2\eta L)-4(\beta+\eta^2L)(1-\kappa)^2L^2\eta\left(2+|1+\gamma|\,C_\gamma\right) > 0$, and $\beta \geq \frac{\eta(1-\kappa)/2+\eta^2L(1-\kappa)^2(1+4\eta^2L^2+|1+\gamma|(\kappa+2\eta^2L^2C_\gamma))}{1-(1-\kappa)^2(1+4\eta^2L^2+|1+\gamma|(\kappa+2\eta^2L^2C_\gamma)} \geq 0$ are some non-negative constants.

The proof is given in Appendix B. The convergence result for the general choice of $\gamma \neq 0$ is given in Theorem 6, which has a slightly different choice of κ, β.

Corollary 4 *Under the conditions of Theorem 3, by optimizing η, β, κ, and define $C_\kappa = \kappa/L\eta = \min\left\{\frac{\|\nabla F(\mathbf{x}_0)\|^2}{2L(F(\mathbf{x}_0)-F^\star)}, 1\right\}$, Algorithm 2 satisfies:*

$$\mathbb{E}[\|\nabla F(\mathbf{x}_t)\|^2] \leq 4\sqrt{\frac{C_\kappa L(F(\mathbf{x}_0)-F^\star))(2\sigma_{\text{SGD}}^2/B+d\sigma_{\text{DP}}^2)}{T}} = \mathcal{O}\left(\sqrt{\frac{d}{T}}\right). \tag{7}$$

2.3.2. Privacy-utility trade-off

In Algorithm 2, the step for DP protection that applies the Gaussian mechanism is in Lines 4 and 5. Instead of $\nabla f(\mathbf{x}_t;\xi)$, we treat $\frac{1-\kappa}{\kappa\gamma}\nabla f(\mathbf{x}_t-\gamma\mathbf{d}_{t-1};\xi)+(1-\frac{1-\kappa}{\kappa\gamma})\nabla f(\mathbf{x}_t;\xi)$ as the per-sample gradient and apply the Subsampled Gaussian mechanism to privatize it. Therefore, Algorithm 2 and DPSGD share the same privacy guarantee, as the privacy proof directly follows the Subsampled Gaussian mechanism and the composition of T iterations in Theorem 2.

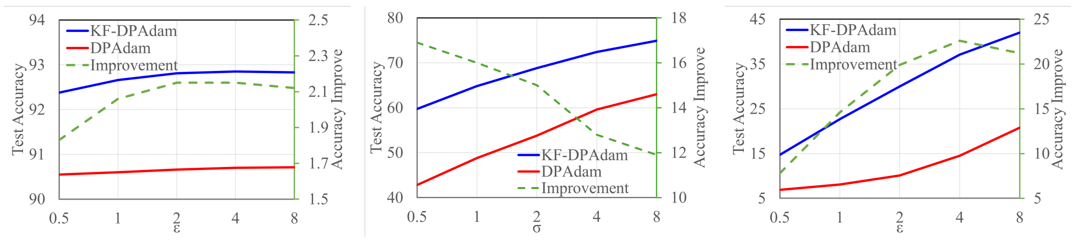
Combining Theorem 2 and Theorem 3, we directly obtain the privacy-utility trade-off:

Theorem 5 *Under the assumptions in Theorem 3. Run Algorithm 2 for $T = \frac{\sqrt{2}N\epsilon\sigma_{\text{SGD}}}{C\sqrt{Bd\ln(1/\delta)}}$ iterations, κ, β are chosen the same as in Corollary 4, then we have:*

$$\mathbb{E}\left\|\nabla F(\mathbf{x}_t)\right\|^2 \leq \frac{8C\sigma_{\text{SGD}}\sqrt{C_\kappa L(F(\mathbf{x}_0)-F^\star)d\ln(1/\delta)}}{\sqrt{B}N\epsilon} = \mathcal{O}\left(\frac{\sqrt{d\ln(1/\delta)}}{N\epsilon}\right) \tag{8}$$

Similarly, the privacy-utility utility trade-off of Algorithm 2 reduces by a constant factor of $\sqrt{1/C_\kappa}$ compared with vanilla DPSGD. To the best of our knowledge, this is the first known result of the theoretical performance reduction for the DPSGD-type algorithm without additional assumptions on the problem.

Let us comment on the value of the reduction factor. When $\|\nabla F(\mathbf{x}_0)\|^2 < 2L(F(\mathbf{x}_0)-F^\star)$, $\sqrt{1/C_\kappa}$ is greater than one and applying DiSK improves upon vanilla DPSGD. **Case I:** For (μ-strongly) convex problems, it is guaranteed that $2\mu(F(\mathbf{x}_0)-F^\star) \leq \|\nabla F(\mathbf{x}_0)\|^2 \leq 2L(F(\mathbf{x}_0)-F^\star)$. Therefore, the factor $C_\kappa \in [\mu/L, 1]$. **Case II:** When training highly non-convex deep learning models, the Lipschitz constant L can be large [17], and $2L(F(\mathbf{x}_0)-F^\star)$ can be much larger than $\|\nabla F(\mathbf{x}_0)\|^2$, which results in a considerable performance improvement compared with DPSGD.



(*a*) MNIST (*b*) CIFAR-10 (*c*) CIFAR-100

Figure 1: Test accuracy of pre-training on MNIST, CIFAR-10, and CIFAR-100 datasets with and without DiSK for different privacy budgets.

Table 1: Test accuracy of fine-tuning result on the GLUE dataset.

Task	($\epsilon = \infty$)	$\epsilon = 6.7$			$\epsilon = 1$		
	Non-DP	DP	**KF-DP**	**KF-DPLora**	DP	**KF-DP**	**KF-DPLora**
MNLI	87.6	83.2	84.8	85.9	80.7	82.0	84.7
QNLI	92.8	87.5	88.9	90.5	86.0	88.7	90.3
SST-2	94.8	91.5	92.8	93.1	91.4	91.5	92.9
QQP	91.9	85.8	88.5	89.0	84.2	86.9	87.8

3. Numerical experiments

In this section, we empirically validate the proposed approach. We perform pre-training and fine-tuning on various image classification (CV) and natural language processing (NLP) tasks using different base algorithms, privacy budgets, and models. In the results, we use **KF-** to denote the DP algorithm with DiSK. In the experiments, we tune the hyper-parameters using a grid search. Specifically, we conduct a grid search on the batch size B, total epochs $E = NT/B$, and step size η for each given privacy budget ϵ. For all experiments, we fix the privacy parameter $\delta = 1/N^{1.1}$ to obtain a reasonable privacy notion. Detailed discussions on the model, data, and base algorithms are in Appendix C.1, and hyper-parameter choices are discussed in Appendix C.2.

CV tasks: We first train the CV models with randomly initialized weights on different image datasets. The results for 5-layer CNN on the MNIST dataset, 5-layer CNN on the CIFAR-10 dataset, and WRN-16-4 on the CIFAR-100 datasets with different privacy budgets are given in Figure 1. The algorithm performance with DiSK significantly outperforms the base algorithm across all used privacy budgets. Additional results for the training curves, results on Imagenet-1k, and ablation studies on hyper-parameter choices are provided in Appendix C.3.

NLP tasks: The final test accuracy for the GLUE dataset is given in Table 1. We follow the same training script and hyper-parameter choices in the experiments as Bu et al. [8]. Compared with the base algorithm (DPAdamW), DiSK improves the average final accuracy on all tasks by 3.4% when $\epsilon = 1$ and 2.6% when $\epsilon = 6.7$. Additional results are provided in Appendix C.4.

4. Conclusion

This paper proposed DiSK, an approach to improve DP optimizers' performance. The approach uses the Kalman filter that combines the noisy observation of the privatized gradient and the dynamics of the true gradient to improve the gradient estimation quality. We provide the theoretical result for the proposed DiSK, and conduct numerical experiments to show that DP optimizers with DiSK outperform the ones without it on various datasets and models for pre-training and fine-tuning tasks.

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, 2016.

[2] Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pages 699–707. PMLR, 2016.

[3] Galen Andrew, Om Thakkar, Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *Advances in Neural Information Processing Systems*, 34:17455–17466, 2021.

[4] Apurva Badithela and Peter Seiler. Analysis of the heavy-ball algorithm using integral quadratic constraints. In *2019 American control conference (ACC)*, pages 4081–4085. IEEE, 2019.

[5] Wenxuan Bao, Francesco Pittaluga, Vijay Kumar BG, and Vincent Bindschaedler. Dp-mix: mixup-based data augmentation for differentially private learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[6] Barbara Bittner and Luc Pronzato. Kalman filtering in stochastic gradient algorithms: construction of a stopping rule. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages ii–709. IEEE, 2004.

[7] Zhiqi Bu, Yu-Xiang Wang, Sheng Zha, and George Karypis. Differentially private optimization on large model at small cost. In *International Conference on Machine Learning*, pages 3192–3218. PMLR, 2023.

[8] Zhiqi Bu, Yu-Xiang Wang, Sheng Zha, and George Karypis. Automatic clipping: Differentially private deep learning made easier and stronger. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Xin Chen, Yujie Tang, and Na Li. Improve single-point zeroth-order optimization using high-pass and low-pass filters. In *International Conference on Machine Learning*, pages 3603–3620. PMLR, 2022.

[10] Christopher A Choquette-Choo, Krishnamurthy Dj Dvijotham, Krishna Pillutla, Arun Ganesh, Thomas Steinke, and Abhradeep Guha Thakurta. Correlated noise provably beats independent noise for differentially private learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[11] Saman Cyrus, Bin Hu, Bryan Van Scoy, and Laurent Lessard. A robust accelerated optimization algorithm for strongly convex functions. In *2018 Annual American Control Conference (ACC)*, pages 1376–1381. IEEE, 2018.

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

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[14] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[16] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Theoretical Computer Science*, 9(3-4):211–407, 2014.

[17] Calypso Herrera, Florian Krach, and Josef Teichmann. Estimating full lipschitz constants of deep neural networks. *arXiv preprint arXiv:2004.13135*, 2020.

[18] Bin Hu and Laurent Lessard. Dissipativity theory for nesterov's accelerated method. In *International Conference on Machine Learning*, pages 1549–1557. PMLR, 2017.

[19] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *International Conference on Machine Learning*, pages 5213–5225. PMLR, 2021.

[20] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960. ISSN 0021-9223. doi: 10.1115/1.3662552. URL https://doi.org/10.1115/1.3662552.

[21] Anastasiia Koloskova, Ryan McKenna, Zachary Charles, John Rush, and H Brendan McMahan. Gradient descent with linearly correlated noise: Theory and applications to differential privacy. *Advances in Neural Information Processing Systems*, 36, 2023.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[23] Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

[24] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations*, 2021.

[25] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[26] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2022.

[27] Zelun Luo, Daniel J Wu, Ehsan Adeli, and Li Fei-Fei. Scalable differential privacy with sparse network finetuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5059–5068, 2021.

[28] Hesameddin Mohammadi, Meisam Razaviyayn, and Mihailo R Jovanović. Tradeoffs between convergence rate and noise amplification for momentum-based accelerated optimization algorithms. *IEEE Transactions on Automatic Control*, 2024.

[29] Michael Muehlebach and Michael Jordan. A dynamical systems perspective on nesterov acceleration. In *International Conference on Machine Learning*, pages 4656–4662. PMLR, 2019.

[30] Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.37. URL https://aclanthology.org/2021.naacl-main.37.

[31] Jekaterina Novikova, Ondrej Dušek, and Verena Rieser. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Saarbrücken, Germany, 2017. URL https://arxiv.org/abs/1706.09254. arXiv:1706.09254.

[32] Nicolas Papernot, Abhradeep Thakurta, Shuang Song, Steve Chien, and Úlfar Erlingsson. Tempered sigmoid activations for deep learning with differential privacy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9312–9321, 2021.

[33] Maria Isabel Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, 43(46):3736–3741, 2004.

[34] Carsten W Scherer, Christian Ebenbauer, and Tobias Holicki. Optimization algorithm synthesis based on integral quadratic constraints: A tutorial. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 2995–3002. IEEE, 2023.

[35] Li Shen, Congliang Chen, Fangyu Zou, Zequn Jie, Ju Sun, and Wei Liu. A unified analysis of adagrad with weighted aggregation and momentum acceleration. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[36] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.

[37] James Vuckovic. Kalman gradient descent: Adaptive variance reduction in stochastic optimization. *arXiv preprint arXiv:1810.12273*, 2018.

[38] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter. *Kalman filtering and neural networks*, pages 221–280, 2001.

[39] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, 2018.

[40] Bao Wang, Quanquan Gu, March Boedihardjo, Lingxiao Wang, Farzin Barekat, and Stanley J Osher. Dp-lssgd: A stochastic optimization method to lift the utility in privacy-preserving erm. In *Mathematical and Scientific Machine Learning*, pages 328–351. PMLR, 2020.

[41] Haoqian Wang, Yi Luo, Wangpeng An, Qingyun Sun, Jun Xu, and Lei Zhang. Pid controller-based stochastic optimization acceleration for deep neural networks. *IEEE transactions on neural networks and learning systems*, 31(12):5079–5091, 2020.

[42] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd international conference on artificial intelligence and statistics*, pages 1226–1235. PMLR, 2019.

[43] Greg Welch, Gary Bishop, et al. *An introduction to the Kalman filter*. Chapel Hill, NC, USA, 1995.

[44] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[45] Yilin Wu, Qian Zhang, and Zhiping Shen. Kalman filtering with multiplicative and additive noises. In *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pages 483–487. IEEE, 2016.

[46] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, 2021.

[47] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. *Advances in neural information processing systems*, 31, 2018.

[48] Xinwei Zhang, Mingyi Hong, and Nicola Elia. Understanding a class of decentralized and federated optimization algorithms: A multirate feedback control perspective. *SIAM Journal on Optimization*, 33(2):652–683, 2023.

[49] Xinwei Zhang, Zhiqi Bu, Mingyi Hong, and Meisam Razaviyayn. Doppler: Differentially private optimizers with low-pass filter for privacy noise reduction. *arXiv preprint arXiv:2408.13460*, 2024.

Appendix A. Additional discussion

A.1. Related Works

Optimization with filters and controllers: The use of filters and controllers in designing and analyzing optimization algorithms has a rich history. Researchers have leveraged high-pass and low-pass filters to enhance gradient estimation in zeroth-order optimization [9], employed PID controllers for both centralized and distributed optimization [41], and analyzed optimizers through the lens of control theory, treating them as dynamic systems [4, 11, 18, 23, 28, 29, 34, 48].

Kalman filter for optimization: The Kalman filter has been utilized in convex optimization for reducing stochastic gradient noise [6, 37]. Vuckovic [37] uses the dynamics of the optimization variable and the gradient to construct the Kalman filter to analyze and improve the performance of momentum methods. Bittner and Pronzato [6] uses gradient and Hessian as its states to construct the dynamic system for SGD to construct a stopping rule. However, these approaches, with their direct application of the Kalman filter, incur prohibitively high computational and memory costs, ranging from $\mathcal{O}(d^3)$ to $\mathcal{O}(d^6)$, rendering them impractical for training large-scale machine learning models.

Improving DP optimization: Numerous techniques have been proposed to enhance DP optimization by mitigating the impact of DP noise. These include adaptive gradient clipping methods that dynamically adjust clipping thresholds [3, 8], parameter-efficient training strategies employing adapters, low-rank weights, or quantization [27, 46, 46], and the design of specialized model architectures less susceptible to noise perturbations [12, 32, 40]. Furthermore, drawing inspiration from signal processing, researchers have explored the use of colored high-frequency DP noise to separate it from the gradient [21] and the application of low-pass filters to extract the gradient signal from noisy observations [49].

A.2. Background on Kalman Filter

In this section, we provide an introduction and derivation of the Kalman filter. Kalman filter is introduced in [20] and wildly used for control systems in accurately estimating system states with noisy observation and known system dynamics. Specifically, given a linear system with **System update** and **Observation**:

$$\begin{aligned}
\boldsymbol{\theta}_t &= \mathbf{A}_t \boldsymbol{\theta}_{t-1} + \mathbf{u}_t + \mathbf{v}_t, && \textit{(System update)} \\
\boldsymbol{\psi}_t &= \mathbf{C}_t \boldsymbol{\theta}_t + \mathbf{w}_t, && \textit{(Observation)}
\end{aligned}$$

where $\mathbf{A}_t \in \mathbb{R}^{d_\theta \times d_\theta}$, $\mathbf{C}_t \in \mathbb{R}^{d_\psi \times d_\theta}$ are the transition and observation matrices; $\boldsymbol{\theta}_t \in \mathbb{R}^{d_\theta}$ is the unknown variable to be estimated; $\boldsymbol{\psi}_t \in \mathbb{R}^{d_\psi}$ denotes the observation of the system; $\mathbf{u}_t \in \mathbb{R}^{d_\theta}$ denotes the known input; and $\mathbf{v}_t \in \mathbb{R}^{d_\theta}, \mathbf{w}_t \in \mathbb{R}^{d_\psi}$ are the process and observation noises that follow Gaussian distribution $\mathcal{N}(0, \Sigma_\mathbf{v}), \mathcal{N}(0, \Sigma_\mathbf{w})$, respectively. Then, the Kalman filter takes the following form [20, 43]:

$$\begin{aligned}
\tilde{\boldsymbol{\theta}}_{t|t-1} &= \mathbf{A}_t \tilde{\boldsymbol{\theta}}_{t-1} + \mathbf{u}_t && \textit{(Prediction)} \\
\mathbf{P}_{t|t-1} &= \mathbf{A}_t \mathbf{P}_{t-1} + \Sigma_\mathbf{v} \\
\mathbf{K}_t &= \mathbf{P}_{t|t-1} \mathbf{C}_t^\top (\mathbf{C}_t \mathbf{P}_{t|t-1} \mathbf{C}_t^\top + \Sigma_\mathbf{w})^{-1} \\
\tilde{\boldsymbol{\theta}}_t &= (\mathbf{I} - \mathbf{K}_t \mathbf{C}_t) \tilde{\boldsymbol{\theta}}_{t|t-1} + \mathbf{K}_t \boldsymbol{\psi}_t && \textit{(Correction)}
\end{aligned}$$

$$\mathbf{P}_t = (\mathbf{I}_{d_\theta} - \mathbf{K}_t\mathbf{C}_t)\mathbf{P}_{t|t-1}.$$

The filter first *predicts* the state $\tilde{\boldsymbol{\theta}}_{t|t-1}$ by the system dynamics, and compute the *filter gain* $\mathbf{K}_t \in \mathbb{R}^{d_\theta \times d_\psi}$ based on the covariance matrix $\mathbf{P}_t \in \mathbb{R}^{d_\theta \times d_\theta}$, and *corrects* the prediction with system observation $\boldsymbol{\psi}_t$ to obtain $\tilde{\boldsymbol{\theta}}_t$. The Kalman filter makes use of both the noisy observation and the prior knowledge of the system dynamics to obtain an accurate estimation of the state $\boldsymbol{\theta}_t$. The goal of the Kalman filter is to estimate $\boldsymbol{\theta}_t$ with the observation of $\boldsymbol{\psi}_t$ with the least mean-square error, and serves as the Best Linear Unbiased Estimator (BLUE) [43].

Derivation: First, we denote the estimation of $\boldsymbol{\theta}_t$ as $\tilde{\boldsymbol{\theta}}_t$, and the covariance of $\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_t$ as

$$\mathbf{P}_t = \mathbb{E}[(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_t)^\top].$$

Then, based on the knowledge at time $t-1$, the system dynamics give an unbiased prediction:

$$\tilde{\boldsymbol{\theta}}_{t|t-1} = \mathbf{A}\tilde{\boldsymbol{\theta}}_{t-1} + \mathbf{u}_t, \tag{9}$$

and its covariance is

$$\mathbf{P}_{t|t-1} = \mathbb{E}[(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_{t|t-1})(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_{t|t-1})^\top] = \mathbf{A}\mathbf{P}_{t-1}\mathbf{A}^\top + \Sigma_{\mathbf{v}}. \tag{10}$$

With $\tilde{\boldsymbol{\theta}}_{t|t-1}$, we have an (unbiased) prediction of the observation $\tilde{\boldsymbol{\psi}}_{t|t-1} = \mathbf{C}\tilde{\boldsymbol{\theta}}_{t|t-1}$ at time $t-1$. At time t, by observing $\boldsymbol{\psi}_t$, we have the prediction error $\Delta\boldsymbol{\psi}_t = \boldsymbol{\psi}_t - \tilde{\boldsymbol{\psi}}_{t|t-1} = \boldsymbol{\psi}_t - \mathbf{C}\tilde{\boldsymbol{\theta}}_{t|t-1}$. Since the system is linear, we would like to use the prediction error to correct the prediction:

$$\tilde{\boldsymbol{\theta}}_t = \tilde{\boldsymbol{\theta}}_{t|t-1} + \mathbf{K}_t\Delta\boldsymbol{\psi}_t = \mathbf{K}_t\boldsymbol{\psi}_t + (\mathbf{I} - \mathbf{K}_t\mathbf{C})\tilde{\boldsymbol{\theta}}_{t|t-1}. \tag{11}$$

The goal of the Kalman filter is to minimize the mean-square error: $\min_{\mathbf{K}} \mathbb{E}[\left\|\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_t\right\|^2]$, which is equivalent to minimizing $\mathrm{tr}(\mathbf{P}_t)$. From the definition of $\tilde{\boldsymbol{\theta}}_t$, we have:

$$\begin{aligned}
\mathbf{P}_t &= \mathbb{E}[(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_t)^\top] \\
&= \mathbb{E}[(\boldsymbol{\theta}_t - \mathbf{K}_t\boldsymbol{\psi}_t + (\mathbf{I} - \mathbf{K}_t\mathbf{C})\tilde{\boldsymbol{\theta}}_{t|t-1})(\boldsymbol{\theta}_t - \mathbf{K}_t\boldsymbol{\psi}_t + (\mathbf{I} - \mathbf{K}_t\mathbf{C})\tilde{\boldsymbol{\theta}}_{t|t-1})^\top] \\
&= \mathbb{E}[(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_{t|t-1} - \mathbf{K}_t\mathbf{C}(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_{t|t-1}) - \mathbf{K}_t\mathbf{w}_t)(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_{t|t-1} - \mathbf{K}_t\mathbf{C}(\boldsymbol{\theta}_t - \tilde{\boldsymbol{\theta}}_{t|t-1}) - \mathbf{K}_t\mathbf{w}_t)^\top] \\
&= \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{C}\mathbf{P}_{t|t-1} - (\mathbf{K}_t\mathbf{C}\mathbf{P}_{t|t-1})^\top + \mathbf{K}_t(\mathbf{C}\mathbf{P}_{t|t-1}\mathbf{C}^\top + \Sigma_{\mathbf{w}})\mathbf{K}_t^\top.
\end{aligned}$$

Taking partial derivative to the trace of $\mathbf{P}_t$ with respect to $\mathbf{K}_t$, and set it to zero, we have:

$$\frac{\partial \mathrm{tr}(\mathbf{P}_t)}{\partial \mathbf{K}_t} = -2(\mathbf{C}\mathbf{P}_{t|t-1})^\top + 2(\mathbf{C}\mathbf{P}_{t|t-1}\mathbf{C}^\top + \Sigma_{\mathbf{w}})\mathbf{K}_t^\top = 0,$$

which gives

$$\mathbf{K}_t = \mathbf{P}_{t|t-1}\mathbf{C}^\top(\mathbf{C}\mathbf{P}_{t|t-1}\mathbf{C}^\top + \Sigma_{\mathbf{w}})^{-1}. \tag{12}$$

Substitute $\mathbf{K}_t$ back to $\mathbf{P}_t$, we can simplify

$$\begin{aligned}
\mathbf{P}_t &= \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{C}\mathbf{P}_{t|t-1} - (\mathbf{K}_t\mathbf{C}\mathbf{P}_{t|t-1})^\top + \mathbf{K}_t(\mathbf{C}\mathbf{P}_{t|t-1}\mathbf{C}^\top + \Sigma_{\mathbf{w}})\mathbf{K}_t^\top \\
&= \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{C}\mathbf{P}_{t|t-1} = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\mathbf{P}_{t|t-1}.
\end{aligned} \tag{13}$$

Combining (9)-(13) together, we have the update of the Kalman filter:

$$\tilde{\boldsymbol{\theta}}_{t|t-1} = \mathbf{A}\tilde{\boldsymbol{\theta}}_{t-1} + \mathbf{u}_t$$

$$\begin{aligned}
\mathbf{P}_{t|t-1} &= \mathbf{A}\mathbf{P}_{t-1}\mathbf{A}^\top + \Sigma_{\mathbf{v}} \\
\mathbf{K}_t &= \mathbf{P}_{t|t-1}\mathbf{C}^\top(\mathbf{C}\mathbf{P}_{t|t-1}\mathbf{C}^\top + \Sigma_{\mathbf{w}})^{-1} \\
\tilde{\boldsymbol{\theta}}_t &= (\mathbf{I} - \mathbf{K}_t\mathbf{C})\tilde{\boldsymbol{\theta}}_{t|t-1} + \mathbf{K}_t\boldsymbol{\psi}_t \\
\mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t\mathbf{C})\mathbf{P}_{t|t-1}.
\end{aligned}$$

Variants of Kalman filter: Kalman filter is designed for estimating the states following linear dynamics and achieves optimal performance, i.e., gives the smallest mean square error of the estimation when the system is linear [43]. Extended Kalman filter (EKF) and unscented Kalman filter (UKF) are developed to deal with non-linear systems. EKF linearizes the non-linear system at each step and performs the Kalman filter on the linearized system [33], while UKF takes the effect of the system non-linearity to the noise distribution into consideration and applies the unscented transform on the noise distribution and applies the Kalman filter on the system and the noise distribution [38]. Other extensions of the Kalman filter have been developed for special cases, including multiplicative noise and noisy input $\mathbf{u}_t$ [45].

A.3. Other system dynamics for DPSGD

In this section, we would like to discuss other possible formulations of the system dynamics for DPSGD to apply the Kalman filter.

Optimization variable and gradient version 1: Other than only using the gradients' dynamics in the main paper, we can construct the dynamic system with both the optimization variable $\mathbf{x}$ and the gradient $\nabla F(\mathbf{x})$ as its states [37]:

$$\begin{bmatrix} \mathbf{x}_{t+1} \\ \nabla F(\mathbf{x}_{t+1}) \end{bmatrix} = \begin{bmatrix} \mathbf{I} & -\eta\mathbf{I} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_t \\ \nabla F(\mathbf{x}_t) \end{bmatrix} + \begin{bmatrix} \eta\mathbf{w}_t \\ 0 \end{bmatrix},$$
$$\mathbf{y}_t = \mathbf{x}_t.$$

However, this dynamic is inaccurate as the dynamics of the gradient are simplified to $\nabla F(\mathbf{x}_{t+1}) = \nabla F(\mathbf{x}_t)$. Although the update can further incorporate with the momentum methods, i.e., by adding a momentum $\mathbf{m}_t$ into the system, it fails to reveal the actual dynamic of the system.

Optimization variable and gradient version 2: Instead of treating the gradient as a constant, we can assume the Hessian $\mathbf{H}$ is a constant and utilize the Hessian to reveal the gradient dynamics:

$$\begin{bmatrix} \mathbf{x}_{t+1} \\ \mathbf{x}_t \\ \nabla F(\mathbf{x}_t) \end{bmatrix} = \begin{bmatrix} \mathbf{I} - \eta\mathbf{H} & \eta\mathbf{H} & -\eta\mathbf{I} \\ \mathbf{I} & 0 & 0 \\ \mathbf{H} & -\mathbf{H} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_t \\ \mathbf{x}_{t-1} \\ \nabla F(\mathbf{x}_{t-1}) \end{bmatrix} + \begin{bmatrix} \eta\mathbf{w}_t \\ 0 \\ 0 \end{bmatrix},$$
$$\mathbf{y}_t = \mathbf{x}_t.$$

This system is more accurate in evaluating the gradient at the cost of using an extra $\mathbf{x}_{t-1}$ state and a larger transition matrix. For non-linear problems $F(\mathbf{x})$, where $\mathbf{H}$ is not a constant, we can apply the extended Kalman filter and replace $\mathbf{H}$ with $\mathbf{H}_t$ that linearizes the problem at each step t.

Gradient and Hessian: In work [6], the dynamics of the gradient and Hessian have been used to construct the system:

$$\begin{bmatrix} \nabla F(\mathbf{x}_{t+1}) \\ \mathbf{h}_{t+1} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \Delta\mathbf{X}_t \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \nabla F(\mathbf{x}_t) \\ \mathbf{h}_t \end{bmatrix},$$

$$\mathbf{g}_t = \nabla F(\mathbf{x}_t) + \mathbf{w}_t,$$

where $\mathbf{h}_t = [\mathbf{H}_{1,1}, \ldots, \mathbf{H}_{i,i+j}, \ldots, \mathbf{H}_{d,d}]^\top$, with $j \in [0, \ldots, d-i]$ represents the entries in the upper triangular part of the Hessian matrix. $\Delta\mathbf{X}_t$ is constructed such that $\Delta\mathbf{X}_t\mathbf{h}_t = \mathbf{H}_t(\mathbf{x}_{t+1} - \mathbf{x}_t)$. The system treats the Hessian matrix as a constant matrix (i.e., $\mathbf{h}_{t+1} = \mathbf{h}_t$), and the transition matrix is of size $(d + d(d-1)/2) \times d + d(d-1)/2$.

Although there are different ways to construct the Kamlan filter for gradient noise reduction, the above systems are not implementable in practical deep-learning applications. Because the transition matrices of these systems are non-diagonal, the Kalman filters have non-diagonal gain $\mathbf{K}_t$ and $\mathbf{P}_t$. Therefore, the matrix inversion operation is unavoidable when Kalman filters are implemented based on these systems. The computation complexity for the matrix inversion can be $\mathcal{O}(d^3)$ to $\mathcal{O}(d^6)$, and the memory consumption is $\mathcal{O}(d^2)$ to $\mathcal{O}(d^4)$ for storing the matrices of the Kalman filter.

A.4. Algorithm Simplification

In this section, we explain how the simplification proceeds from Algorithm 5 to Algorithm 2. Recall that updates of Algorithm 5 is

$$\begin{aligned}
\tilde{\mathbf{g}}_{t|t-1} &= \tilde{\mathbf{g}}_{t-1} + \tilde{\mathbf{H}}_t(\mathbf{x}_t - \mathbf{x}_{t-1}) && \textit{(Prediction)}\\
\mathbf{P}_{t|t-1} &= \mathbf{P}_{t-1} + \Sigma_{\mathbf{H}} + \Sigma_{\mathbf{v}}\\
\mathbf{K}_t &= \mathbf{P}_{t|t-1}\,\mathbb{E}[\mathbf{C}_t]^\top \left(\Sigma_{\mathbf{w}} + \mathbb{E}[\mathbf{C}_t]\left(\Sigma_{\mathbf{C}}\mathbf{S}_t + \mathbf{P}_{t|t-1}\right)\mathbb{E}[\mathbf{C}_t]^\top - \Sigma_{\mathbf{H}}\right)^{-1}\\
\tilde{\mathbf{g}}_t &= \tilde{\mathbf{g}}_{t|t-1} + \mathbf{K}_t(\mathbf{g}_t - \mathbb{E}[\mathbf{C}_t]\tilde{\mathbf{g}}_{t|t-1}) && \textit{(Correction)}\\
\mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t\,\mathbb{E}[\mathbf{C}_t])\mathbf{P}_{t|t-1}\\
\mathbf{S}_t &= \mathbb{E}[\tilde{\mathbf{g}}_t\tilde{\mathbf{g}}_t^\top].
\end{aligned}$$

We apply four steps of simplification, including

1. Replacing random $\mathbf{C}_t$ with constant $\mathbf{I}_d$. The randomness of $\mathbf{C}_t$ comes from the sub-sampling and the clipping operation. To simplify the algorithm, the clipping operation $\mathrm{clip}\left(\nabla f(\mathbf{x};\xi), C\right)$ can be viewed as a change of the problem to be optimized. Specifically, instead of optimizing the ERM problem (1) $F(\mathbf{x})$, the DPSGD algorithm with clipping optimizes $F_C(\mathbf{x})$, where

$$F_C(\mathbf{x}) = \int_0^1 \nabla F_C(z\mathbf{x})^\top \mathbf{x}\mathrm{d}z,\ \nabla F_C(\mathbf{x}) = \frac{1}{N}\sum_{\xi\in\mathcal{D}} \mathrm{clip}\left(\nabla f(\mathbf{x};\xi), C\right). \tag{14}$$

   Further by assuming that the sub-sampled mini-batch gradient only causes additive noise, i.e., $\frac{1}{B}\sum_{\xi\in\mathcal{B}} \mathrm{clip}\left(\nabla f(\mathbf{x},\xi), C\right) = \nabla F(\mathbf{x}) + \mathbf{w}_{\mathrm{SGD}}$, then $\mathbf{C} = \mathbf{I}_d$ is an identity matrix, and $\Sigma_{\mathbf{C}} = 0$.

2. Use finite difference to estimate $\mathbf{H}_t\mathbf{d}_{t-1}$. Note that we cannot directly obtain $\mathbf{H}_t\mathbf{d}_{t-1}$, as it requires computing the Hessian of the problem. In practice, we can use a finite difference to approximate such value:

$$\mathbf{H}_t\mathbf{d}_{t-1} = \frac{\nabla F(\mathbf{x}_t + \gamma\mathbf{d}_{t-1}) - \nabla F(\mathbf{x}_t)}{\gamma} + \Delta(\gamma) \approx \frac{1}{B}\sum_{\xi\in\mathcal{B}} \frac{\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma},$$

where the approximation error $\Delta(\gamma) = \mathcal{O}(\gamma)$. By using such an estimation, we can efficiently estimate $\mathbf{H}_t\mathbf{d}_{t-1}$ with only first-order information in the DP optimization, greatly reducing memory and computation complexity.

3. Replace $\Sigma_{\mathbf{H}}$ with diagonal matrix $\sigma_H^2\mathbf{I}_d$, $\Sigma_{\mathbf{v}}$ with $\sigma_{\mathbf{v}}^2\mathbf{I}_d$, and $\Sigma_{\mathbf{w}}$ with $\sigma_{\mathbf{w}}^2\mathbf{I}_d$. The matrix computation in Algorithm 5 is extremely time-consuming and memory inefficient. In line 8 of Algorithm 5, the matrix inversion requires $\mathcal{O}(d^3)$ computation and memory complexity, impractical for optimizing large models with billions of trainable parameters. Therefore, to simplify the algorithm, we assume Σ_H in the Kalman filter is a time-invariant *diagonal* matrices, $\Sigma_H = \sigma_H^2\mathbf{I}_d$. By using this simplification, matrices $\mathbf{P}, \mathbf{K}$ become $p\mathbf{I}_d, k\mathbf{I}_d$, for some scalars p, k, and all matrix computations involved in the algorithm become scalar-vector multiplication and memory consumption is reduced to $\mathcal{O}(1)$, which is affordable for DP training.

4. Use fixed filter gain κ. With the above simplification, k_t converges to its stable value with a linear rate, i.e., $\|k_t - k_\infty\| = \mathcal{O}(c_k^t)$, for some $c_k \in (0,1)$, we can use $k_t = \kappa, \forall\, t$ to further simplify the algorithm and avoid iteratively updating p_t and recomputing k_t for each step.

Step 1. By replacing $\mathbf{C}_t$ with $\mathbf{I}_d$, and $\Sigma_{\mathbf{C}} = 0$, the update becomes

$$\begin{aligned}
\tilde{\mathbf{g}}_{t|t-1} &= \tilde{\mathbf{g}}_{t-1} + \tilde{\mathbf{H}}_t(\mathbf{x}_t - \mathbf{x}_{t-1}) && \textit{(Prediction)}\\
\mathbf{P}_{t|t-1} &= \mathbf{P}_{t-1} + \Sigma_{\mathbf{H}} + \Sigma_{\mathbf{v}}\\
\mathbf{K}_t &= \mathbf{P}_{t|t-1}\left(\Sigma_{\mathbf{w}} + \mathbf{P}_{t|t-1} - \Sigma_{\mathbf{H}}\right)^{-1}\\
\tilde{\mathbf{g}}_t &= \tilde{\mathbf{g}}_{t|t-1} + \mathbf{K}_t(\mathbf{g}_t - \tilde{\mathbf{g}}_{t|t-1}) && \textit{(Correction)}\\
\mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t)\mathbf{P}_{t|t-1}.
\end{aligned}$$

Step 2. By using the finite difference to estimate $\mathbf{H}_t\mathbf{d}_{t-1}$, the prediction step becomes:

$$\tilde{\mathbf{g}}_{t|t-1} = \tilde{\mathbf{g}}_{t-1} + \frac{1}{B}\sum_{\xi\in\mathcal{B}}\frac{\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma} \qquad \textit{(Prediction)}.$$

Step 3. By replacing Σ's with diagonal matrix $\sigma^2\mathbf{I}_d$'s, the update becomes:

$$\begin{aligned}
\tilde{\mathbf{g}}_{t|t-1} &= \tilde{\mathbf{g}}_{t-1} + \frac{1}{B}\sum_{\xi\in\mathcal{B}}\frac{\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma} && \textit{(Prediction)}\\
p_{t|t-1} &= p_{t-1} + \sigma_H^2 + \sigma_{\mathbf{v}}^2, \mathbf{P}_{t|t-1} = p_{t|t-1}\mathbf{I}_d\\
k_t &= \frac{p_{t|t-1}}{p_{t|t-1} + \sigma_{\mathbf{w}}^2 - \sigma_H^2} = \frac{p_{t-1} + \sigma_H^2 + \sigma_{\mathbf{v}}^2}{p_{t-1} + \sigma_{\mathbf{w}}^2 + \sigma_{\mathbf{v}}^2}, \mathbf{K}_t = k_t\mathbf{I}_d\\
\tilde{\mathbf{g}}_t &= \tilde{\mathbf{g}}_{t|t-1} + k_t(\mathbf{g}_t - \tilde{\mathbf{g}}_{t|t-1}) && \textit{(Correction)}\\
p_t &= (1 - k_t)p_{t|t-1} = \frac{(\sigma_{\mathbf{w}}^2 - \sigma_H^2)(p_{t-1} + \sigma_H^2 + \sigma_{\mathbf{v}}^2)}{p_{t-1} + \sigma_{\mathbf{w}}^2 + \sigma_{\mathbf{v}}^2}.
\end{aligned}$$

Step 4. As discussed in the main paper, the update of k_t, p_t becomes:

$$k_t = \frac{p_{t-1} + \sigma_H^2 + \sigma_{\mathbf{v}}^2}{p_{t-1} + \sigma_{\mathbf{w}}^2 + \sigma_{\mathbf{v}}^2},$$

$$p_t = \frac{(\sigma_{\mathbf{w}}^2 - \sigma_H^2)(p_{t-1} + \sigma_H^2 + \sigma_{\mathbf{v}}^2)}{p_{t-1} + \sigma_{\mathbf{w}}^2 + \sigma_{\mathbf{v}}^2}.$$

Therefore, p_t converges to $p_\infty = \frac{\sqrt{\sigma_H^2+\sigma_{\mathbf{v}}^2}\sqrt{4\sigma_{\mathbf{w}}^2-3\sigma_H^2+\sigma_{\mathbf{v}}^2}-(\sigma_H^2+\sigma_{\mathbf{v}}^2)}{2}$, k_t converges to $k_\infty = \frac{p_\infty+\sigma_H^2+\sigma_{\mathbf{v}}^2}{p_\infty+\sigma_{\mathbf{w}}^2+\sigma_{\mathbf{v}}^2}$, with rate $c_k = \frac{2\sigma_{\mathbf{w}}^2+3\sigma_H^2+\sigma_{\mathbf{v}}^2-\sqrt{(\sigma_{\mathbf{v}}^2+\sigma_H^2)(4\sigma_{\mathbf{w}}^2+\sigma_{\mathbf{v}}^2-3\sigma_H^2)}}{2\sigma_{\mathbf{w}}^2+3\sigma_H^2+\sigma_{\mathbf{v}}^2+\sqrt{(\sigma_{\mathbf{v}}^2+\sigma_H^2)(4\sigma_{\mathbf{w}}^2+\sigma_{\mathbf{v}}^2-3\sigma_H^2)}}$. Therefore, we define $\kappa = k_\infty$ and replace k_t with κ, and the update becomes:

$$\begin{aligned}
\tilde{\mathbf{g}}_{t|t-1} &= \tilde{\mathbf{g}}_{t-1} + \frac{1}{B}\sum_{\xi\in\mathcal{B}} \frac{\nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma} && \textit{(Prediction)}\\
\tilde{\mathbf{g}}_t &= \tilde{\mathbf{g}}_{t|t-1} + \kappa(\mathbf{g}_t - \tilde{\mathbf{g}}_{t|t-1}) && \textit{(Correction)}
\end{aligned}$$

Rearrange the terms, we have:

$$\begin{aligned}
\tilde{\mathbf{g}}_t &= (1-\kappa)\tilde{\mathbf{g}}_t + \kappa \mathbf{g}_t + (1-\kappa)\frac{1}{B}\sum_{\xi\in\mathcal{B}} \frac{\nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma}\\
&= (1-\kappa)\tilde{\mathbf{g}}_t + \kappa\hat{\mathbf{g}}_t, \text{ with}\\
\hat{\mathbf{g}}_t &= \mathbf{g}_t + \frac{1-\kappa}{\kappa}\frac{1}{B}\sum_{\xi\in\mathcal{B}} \frac{\nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)}{\gamma}\\
&= \frac{1}{B}\sum_{\xi\in\mathcal{B}}\left(\frac{1-\kappa}{\kappa\gamma}\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) + \left(1 - \frac{1-\kappa}{\kappa\gamma}\right)\nabla f(\mathbf{x}_t;\xi)\right).
\end{aligned}$$

Then, by privatizing $\hat{\mathbf{g}}_t$ and rename it as $\mathbf{g}_t$, we have:

$$\begin{aligned}
\mathbf{g}_t &= \frac{1}{B}\sum_{\xi\in\mathcal{B}} \operatorname{clip}\left(\frac{1-\kappa}{\kappa\gamma}\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) + \left(1-\frac{1-\kappa}{\kappa\gamma}\right)\nabla f(\mathbf{x}_t;\xi), C\right) + \mathbf{w}_t\\
\tilde{\mathbf{g}}_t &= (1-\kappa)\tilde{\mathbf{g}}_t + \kappa\mathbf{g}_t,
\end{aligned}$$

which is the update of Algorithm 2 (Lines 5 and 6).

Remark 1. When the clipping is inactive, the update of Algorithm 2 is:

$$\begin{aligned}
\mathbf{g}_t &= \frac{1}{B}\sum_{\xi\in\mathcal{B}_t} \nabla f(\mathbf{x}_t;\xi) + \mathbf{w}_t && \textit{(Observation)}\\
\tilde{\mathbf{g}}_{t|t-1} &= \tilde{\mathbf{g}}_{t-1} + \frac{1}{\gamma B}\sum_{\xi\in\mathcal{B}}(\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_t;\xi) - \nabla f(\mathbf{x}_t;\xi)) && \textit{(Prediction)}\\
\tilde{\mathbf{g}}_t &= (1-\kappa)\tilde{\mathbf{g}}_{t|t-1} + \kappa\mathbf{g}_t && \textit{(Correction)}\\
\mathbf{x}_{t+1} &= \mathbf{x}_t - \eta\tilde{\mathbf{g}}_t, \qquad \mathbf{d}_t = \mathbf{x}_{t+1} - \mathbf{x}_t,
\end{aligned} \tag{15}$$

where $\mathbf{K}_t = \kappa\mathbf{I}_d, \forall t$, and the above update matches the ones in the Kalman filter.

A.5. Algorithm Connection

Connection to NAG: The algorithm in Algorithm 2 has an inner connection to the (unified) Nesterov accelerated gradient (NAG) method [35, 36]. The update of NAG writes

$$\begin{aligned}
\mathbf{m}_t &= \mu\mathbf{m}_{t-1} - \eta\nabla F(\mathbf{x}_t + \mu\mathbf{m}_{t-1})\\
\mathbf{x}_{t+1} &= \mathbf{x}_t + \mathbf{m}_t.
\end{aligned} \tag{16}$$

In comparison, in Algorithm 2, by letting $\gamma = \frac{1-\kappa}{\kappa}$, $\eta = -\kappa$, and letting the update be OptimizerUpdate be SGD, then the algorithm becomes

$$\begin{aligned}\tilde{\mathbf{g}}_t &= (1-\kappa)\tilde{\mathbf{g}}_{t-1} - \eta\left(\frac{1}{B}\sum_{\xi\in\mathcal{B}_t}\nabla f\left(\mathbf{x}_t + (1-\kappa)\tilde{\mathbf{g}}_{t-1};\xi\right) + \mathbf{w}_t\right) \\ \mathbf{x}_{t+1} &= \mathbf{x}_t + \kappa\tilde{\mathbf{g}}_t.\end{aligned} \tag{17}$$

Notice that (16) and (17) are different by an extra learning rate κ in (17) in the update of $\mathbf{x}_{t+1}$. Therefore, NAG is a special case of Algorithm 2 with specific choices of η, γ.

Connection to DOPPLER: DOPPLER and DiSK both use a filter to separate the gradient signal from the DP noise. If $\mathbf{g}_t$ only evaluates the gradient at one point instead of using the linear combination of gradients at two points in Algorithm 2, DiSK becomes DOPPLER with a first-order filter. The key difference is that DOPPLER assumes an underlying low-frequency dynamic of the gradient and applies a time-invariant low-pass filter. While in DiSK, we do not assume the frequency property of the gradient signal. Instead, we incorporate the gradient dynamics into the filtering procedure and use the Kalman filter, a predictive filtering approach, to reduce the impact of DP noise.

Appendix B. Proof for Section 2.3

In this section, we provide the detailed proof for the results in Section 2.3 and Theorem 6 for the case $\gamma \neq -1$.

Theorem 6 *Assume A1-A2 holds. For any fixed $\sigma_{\mathrm{DP}}^2, C, \gamma \neq 0$, by choosing $\eta < \frac{1}{2L(1+8\beta L)}$, $\kappa > 8\eta^2 L^2$, and run Algorithm 2 for T iterations, we have*

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T}\mathbb{E}\left\|\nabla F_C(\mathbf{x}_t)\right\|^2 \\ &\quad\leq \frac{2(F_C(\mathbf{x}_0) + \beta\left\|\nabla F_C(\mathbf{x}_0)\right\|^2 - F_C^\star)}{C_2\eta T} + \frac{2(\beta+\eta^2 L)\kappa^2}{C_2\eta}\left(\frac{2\sigma_{\mathrm{SGD}}^2}{B} + d\sigma_{\mathrm{DP}}^2\right),\end{aligned}$$

where $\beta \geq \frac{\eta(1-\kappa)/2 + \eta^2 L(1-\kappa)^2(1+8(1+1/\kappa)\eta^2L^2)}{1-(1-\kappa)^2(1+\kappa+8(1+1/\kappa)\eta^2L^2)} \geq 0$ and $C_2 = 1+\kappa-2\eta L(1+\kappa+8L(\beta+(1+1/\kappa)\eta^2 L)(1-\kappa)^2) > 0$ are non-negative constants.

To facilitate our analysis, we make the following assumptions to (1):

A 1 (Smoothness) *$f(\cdot,\xi)$ is L-smooth for any ξ, i.e.,*

$$\left\|\nabla f(\mathbf{x};\xi) - \nabla f(\mathbf{y};\xi)\right\| \leq L\left\|\mathbf{x}-\mathbf{y}\right\|,\ \forall \xi\in\mathcal{D},\ \forall \mathbf{x},\mathbf{y}\in\mathbb{R}^d.$$

A 2 (Bounded Variance) *The per-sample gradient has bounded variance with*

$$\mathbb{E}_{\xi\in\mathcal{D}}\left\|\nabla f(\mathbf{x};\xi) - \nabla F(\mathbf{x})\right\|^2 \leq \sigma_{SGD}^2,\ \forall \mathbf{x}\in\mathbb{R}^d,$$

where $\mathbb{E}_{\xi\in\mathcal{D}}$ denotes the expectation taken on the randomness over ξ that is uniformly sampled from dataset $\mathcal{D}$.

A 3 (Bounded Gradient) *Each per-sample gradient has a bounded norm, i.e.,*

$$\|\nabla f(\mathbf{x};\xi)\| \le G,\ \forall \mathbf{x} \in \mathbb{R}^d, \forall \xi \in \mathcal{D}.$$

Let us briefly comment on these assumptions: A1 and A2 are standard in non-convex optimization [1, 2, 47]; and A3 is commonly used in analyzing the convergence of DP algorithms [1, 3, 40] to avoid introducing the clipping bias. Since the impact of clipping is not the major focus of this paper, we follow the existing analyses and use A3 to simplify our theoretical analysis.

We will use the following inequalities in our proofs:

$$\langle \mathbf{a}, \mathbf{b}\rangle \le \frac{1}{2\alpha}\|\mathbf{a}\|^2 + \frac{\alpha}{2}\|\mathbf{b}\|^2, \tag{18}$$
$$\|\mathbf{a}+\mathbf{b}\|^2 \le (1+\alpha)\|\mathbf{a}\|^2 + (1+1/\alpha)\|\mathbf{b}\|^2. \tag{19}$$

In the following sections, we use $\mathbb{E}_t$ to denote the expectation conditioned on all the information before iteration t. To prove Theorem 3, we first provide the following lemma to bound the difference between $\tilde{\mathbf{g}}_t$ defined in Line 6 of Algorithm 2, and $\nabla F(\mathbf{x}_t)$. Let us define $\Delta_t = \nabla F(\mathbf{x}_t) - \tilde{\mathbf{g}}_t$. We have:

Lemma 7 *Assume A1, A2, and A3 holds and choose $C \ge \left(1 + \frac{2(1-\kappa)}{\kappa}\right) G$, we have:*

$$\begin{aligned}\mathbb{E}_t\|\Delta_t\|^2 \le\ & (1-\kappa)^2\left(1 + 4\eta^2L^2 + |1+\gamma|\left(\kappa + 2\eta^2L^2C_\gamma\right)\right)\|\Delta_{t-1}\|^2 \\ & + 2\eta^2L^2(1-\kappa)^2\left(2 + |1+\gamma|\,C_\gamma\right)\|\nabla F(\mathbf{x}_{t-1})\|^2 \\ & + \kappa^2\left((2+|1+\gamma|)\frac{\sigma_{SGD}^2}{B} + d\sigma_{DP}^2\right),\end{aligned} \tag{20}$$

where we define $C_\gamma = \left(1 + \frac{4(2+1/\kappa+|1+\gamma|)}{\gamma^2}\right)$.

B.1. Proof of Lemma 7

First notice that when choosing $C \ge \left(1 + \frac{2(1-\kappa)}{\kappa}\right) G$, the clipping operation is inactive. By the update of $\tilde{\mathbf{g}}_t$ in Line 6 of Algorithm 2, we have:

$$\begin{aligned}&\mathbb{E}_t\|\Delta_t\|^2 \\ &= \mathbb{E}_t\Bigg\|\nabla F(\mathbf{x}_t) - (1-\kappa)\tilde{\mathbf{g}}_{t-1} - \frac{\kappa}{B}\sum_{\xi\in\mathcal{B}_t}\nabla f(\mathbf{x}_t,\xi) - \kappa\mathbf{w}_t \\ &\qquad - \frac{1-\kappa}{\gamma B}\sum_{\xi\in\mathcal{B}_t}\left(\nabla f(\mathbf{x}_t + \gamma\mathbf{d}_{t-1};\xi) - \nabla f(\mathbf{x}_t;\xi)\right)\Bigg\|^2 \\ &\overset{(a)}{=} \mathbb{E}_t\|(1-\kappa)(\nabla F(\mathbf{x}_t) - \nabla F(\mathbf{x}_{t-1})) + (1-\kappa)(\nabla F(\mathbf{x}_{t-1}) - \tilde{\mathbf{g}}_{t-1}) \\ &\qquad + \kappa\left(\nabla F(\mathbf{x}_t) - \frac{1}{B}\sum_{\xi\in\mathcal{B}_t}\nabla f(\mathbf{x}_t,\xi) - \mathbf{w}_t\right)\end{aligned}$$

$$
- \frac{1-\kappa}{\gamma B} \sum_{\xi \in \mathcal{B}_t} \left(\nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1}; \xi) - \nabla f(\mathbf{x}_t; \xi)\right) \Big\|^2
$$

$$
\overset{(b)}{=} \mathbb{E}_t \Bigg\| (1-\kappa) \underbrace{\left(\nabla F(\mathbf{x}_t) - \nabla F(\mathbf{x}_{t-1}) - \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \left(\nabla f(\mathbf{x}_t; \xi) - \nabla f(\mathbf{x}_{t-1}; \xi)\right)\right)}_{:=D_1}
$$

$$
+ (1-\kappa)\Delta_{t-1} + \kappa \underbrace{\left(\nabla F(\mathbf{x}_t) - \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \nabla f(\mathbf{x}_t, \xi)\right)}_{:=D_2} - \kappa \mathbf{w}_t
$$

$$
- (1-\kappa) \underbrace{\frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \left(\frac{1}{\gamma} \nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1}; \xi) + \nabla f(\mathbf{x}_{t-1}; \xi) - \frac{1+\gamma}{\gamma} \nabla f(\mathbf{x}_t; \xi)\right)}_{:=D_3} \Bigg\|^2
$$

$$
\begin{aligned}
&\overset{(c)}{=} (1-\kappa)^2 \mathbb{E}_t \|D_1\|^2 + (1-\kappa)^2 \|\Delta_{t-1}\|^2 + \kappa^2 \mathbb{E}_t[\|D_2\|^2] + (1-\kappa)^2 \mathbb{E}_t \|D_3\|^2 + \kappa^2 \mathbb{E}[\|\mathbf{w}_t\|^2] \\
&\quad + 2(1-\kappa)^2 \langle \mathbb{E}_t[D_1], \Delta_{t-1} \rangle + 2(1-\kappa)\kappa \langle \mathbb{E}_t[D_2], \Delta_{t-1} \rangle - 2(1-\kappa)^2 \langle \mathbb{E}_t[D_3], \Delta_{t-1} \rangle \\
&\quad + 2\,\mathbb{E}_t[\langle (1-\kappa)D_1, \kappa D_2 \rangle] - 2(1-\kappa)^2 \mathbb{E}_t[\langle D_1, D_3 \rangle] - 2\,\mathbb{E}_t[\langle \kappa D_2, (1-\kappa) D_3 \rangle], \\
&\overset{(d)}{\leq} (1-\kappa)^2 (2 + |1+\gamma|) \mathbb{E}_t \|D_1\|^2 + (1-\kappa)^2 (1 + \kappa |1+\gamma|) \|\Delta_{t-1}\|^2 + \kappa^2 (2 + |1+\gamma|) \mathbb{E}_t[\|D_2\|^2] \\
&\quad + (1-\kappa)^2 \left(1 + \frac{2}{|1+\gamma|} + \frac{1}{\kappa |1+\gamma|}\right) \mathbb{E}_t \|D_3\|^2 + \kappa^2 d \sigma_{\text{DP}}^2, \qquad (21)
\end{aligned}
$$

where (a) we add and subtract $(1-\kappa)\nabla F(\mathbf{x}_{t-1})$ and rearrange the terms; (b) adds and subtracts $\frac{1-\kappa}{B} \sum_{\xi \in \mathcal{B}_t} (\nabla f(\mathbf{x}_t; \xi) - \nabla f(\mathbf{x}_{t-1}; \xi))$; (c) directly expands the square and use the fact that $\mathbb{E}_t[\mathbf{w}_t] = 0$ and $\mathbf{w}_t$ is independent of other terms; and in (d), we notice that $\mathbb{E}_t[D_1] = 0, \mathbb{E}_t[D_2] = 0$, so the first two inner products are zero, and we apply (18) to the other four inner products, with $\alpha = \kappa |1+\gamma|, 1, |1+\gamma|$, respectively. Next, we bound each term separately. For $\mathbb{E}_t[\|D_1\|^2]$, we have:

$$
\begin{aligned}
\mathbb{E}_t[\|D_1\|^2] &= \mathbb{E}_t \left\| \nabla F(\mathbf{x}_t) - \nabla F(\mathbf{x}_{t-1}) - \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \left(\nabla f(\mathbf{x}_t; \xi) - \nabla f(\mathbf{x}_{t-1}; \xi)\right) \right\|^2 \\
&\overset{(a)}{\leq} \mathbb{E}_t \left\| \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \left(\nabla f(\mathbf{x}_t; \xi) - \nabla f(\mathbf{x}_{t-1}; \xi)\right) \right\|^2 \qquad (22) \\
&\overset{(b)}{\leq} L^2 \eta^2 \|\tilde{\mathbf{g}}_{t-1}\|^2 \\
&\overset{(c)}{\leq} 2L^2 \eta^2 (\|\Delta_{t-1}\|^2 + \|\nabla F(\mathbf{x}_{t-1})\|^2),
\end{aligned}
$$

where (a) uses the fact that $\mathbb{E}\|X - \mathbb{E}[X]\|^2 \leq \mathbb{E}\|X\|^2$, with

$$
\nabla F(\mathbf{x}_t) - \nabla F(\mathbf{x}_{t-1}) = \mathbb{E}_t[\frac{1}{B} \sum_{\xi \in \mathcal{B}_t} (\nabla f(\mathbf{x}_t; \xi) - \nabla f(\mathbf{x}_{t-1}; \xi))]
$$

; (b) applies A1 and (c) adds and subtracts $\nabla F(\mathbf{x}_{t-1})$, and applies (19). For $\mathbb{E}_t[\|D_2\|^2]$, we have:

$$\mathbb{E}_t[\|D_2\|^2] = \mathbb{E}_t \left\| \nabla F(\mathbf{x}_t) - \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \nabla f(\mathbf{x}_t, \xi) \right\|^2 \overset{A2}{\leq} \frac{\sigma_{\text{SGD}}^2}{B}. \tag{23}$$

For $\mathbb{E}_t[\|D_3\|^2]$, we have:

$$\begin{aligned}
\mathbb{E}_t[\|D_3\|^2] &= \mathbb{E}_t \left\| \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \left(\frac{1}{\gamma} \nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1}; \xi) + \nabla f(\mathbf{x}_{t-1}; \xi) - \frac{1+\gamma}{\gamma} \nabla f(\mathbf{x}_t; \xi) \right) \right\|^2 \\
&\overset{(a)}{\leq} \frac{1}{B} \sum_{\xi \in \mathcal{B}_t} \left\| \frac{1}{\gamma} \nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1}; \xi) - \frac{1}{\gamma} \nabla f(\mathbf{x}_{t-1}; \xi) \right. \\
&\qquad \left. + \frac{1+\gamma}{\gamma} \nabla f(\mathbf{x}_{t-1}; \xi) - \frac{1+\gamma}{\gamma} \nabla f(\mathbf{x}_t; \xi) \right\|^2 \\
&\overset{(19)}{\leq} \frac{2}{B} \sum_{\xi \in \mathcal{B}_t} \left\| \frac{1}{\gamma} \nabla f(\mathbf{x}_t + \gamma \mathbf{d}_{t-1}; \xi) - \frac{1}{\gamma} \nabla f(\mathbf{x}_{t-1}; \xi) \right\|^2 \\
&\quad + \frac{2}{B} \sum_{\xi \in \mathcal{B}_t} \left\| \frac{1+\gamma}{\gamma} \nabla f(\mathbf{x}_{t-1}; \xi) - \frac{1+\gamma}{\gamma} \nabla f(\mathbf{x}_t; \xi) \right\|^2 \\
&\overset{A1}{\leq} \frac{2L^2}{\gamma^2} \|\mathbf{x}_t + \gamma \mathbf{d}_{t-1} - \mathbf{x}_{t-1}\|^2 + \frac{2L^2(1+\gamma)^2}{\gamma^2} \|\mathbf{x}_{t-1} - \mathbf{x}_t\|^2 \\
&\overset{(b)}{=} \frac{4L^2(1+\gamma)^2\eta^2}{\gamma^2} \|\tilde{\mathbf{g}}_{t-1}\|^2 \\
&\overset{(c)}{\leq} \frac{8L^2(1+\gamma)^2\eta^2}{\gamma^2} (\|\Delta_{t-1}\|^2 + \|\nabla F(\mathbf{x}_{t-1})\|^2),
\end{aligned} \tag{24}$$

where (a) applies Jensens' inequality to $\|\cdot\|^2$, and we add and subtract $\frac{1}{\gamma} \nabla f(\mathbf{x}_{t-1}; \xi)$; (b) apples (19); (b) uses the fact that $\mathbf{d}_{t-1} = \mathbf{x}_t - \mathbf{x}_{t-1} = -\eta \tilde{\mathbf{g}}_{t-1}$; and (c) adds and subtracts $\nabla F(\mathbf{x}_{t-1})$, and applies (19). Plug in (22) – (24) to (21), we have:

$$\begin{aligned}
\mathbb{E}_t \|\Delta_t\|^2 &\leq (1-\kappa)^2 (2 + |1+\gamma|)\, \mathbb{E}_t \|D_1\|^2 + (1-\kappa)^2 (1 + \kappa |1+\gamma|) \|\Delta_{t-1}\|^2 + \kappa^2 d \sigma_{\text{DP}}^2 \\
&\quad + \kappa^2 (2 + |1+\gamma|)\, \mathbb{E}_t[\|D_2\|^2] + (1-\kappa)^2 \left(1 + \frac{2}{|1+\gamma|} + \frac{1}{\kappa |1+\gamma|} \right) \mathbb{E}_t \|D_3\|^2 \\
&\leq (1-\kappa)^2 \left(1 + 4\eta^2 L^2 + |1+\gamma| \left(\kappa + 2\eta^2 L^2 C_\gamma \right) \right) \|\Delta_{t-1}\|^2 \\
&\quad + 2\eta^2 L^2 (1-\kappa)^2 \left(2 + |1+\gamma|\, C_\gamma \right) \|\nabla F(\mathbf{x}_{t-1})\|^2 \\
&\quad + \kappa^2 \left((2 + |1+\gamma|) \frac{\sigma_{\text{SGD}}^2}{B} + d \sigma_{\text{DP}}^2 \right),
\end{aligned} \tag{25}$$

where we define $C_\gamma := \left(1 + \frac{4(2 + 1/\kappa + |1+\gamma|)}{\gamma^2} \right)$. This completes the proof of Lemma 7.

B.2. Proof of Theorem 3

Now, we are ready to prove Theorem 3. By choosing $C \geq G(1 + \frac{2(1-\kappa)}{\kappa\gamma})$, the clipping is inactive. Recall that from the update rule, we have

$$\begin{aligned}\mathbb{E}_t[\tilde{\mathbf{g}}_t] &= (1-\kappa)\tilde{\mathbf{g}}_{t-1} + \kappa\nabla F(\mathbf{x}_t) + (1-\kappa)(\nabla F(\mathbf{x}_t) - \nabla F(\mathbf{x}_{t-1})) \\ &= \nabla F(\mathbf{x}_t) - (1-\kappa)\Delta_{t-1}.\end{aligned} \tag{26}$$

Then, by A1 we have $F(\cdot)$ is also L-smooth, so it satisfies

$$F(\mathbf{y}) \leq F(\mathbf{x}) + \langle \nabla F(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle, \frac{L}{2}\|\mathbf{y}-\mathbf{x}\|^2.$$

Substitute $\mathbf{y} = \mathbf{x}_{t+1}, \mathbf{x} = \mathbf{x}_t$ to the above relation and take expectation over the randomness in iteration t, we have:

$$\begin{aligned}\mathbb{E}_t[F(\mathbf{x}_{t+1})] - F(\mathbf{x}_t) &\leq -\eta\langle\nabla F(\mathbf{x}_t), \mathbb{E}_t[\tilde{\mathbf{g}}_t]\rangle + \frac{\eta^2 L}{2}\mathbb{E}_t\|\tilde{\mathbf{g}}_t\|^2 \\ &\overset{(a)}{=} -\eta\|\nabla F(\mathbf{x}_t)\|^2 + \eta(1-\kappa)\langle\nabla F(\mathbf{x}_t), \Delta_{t-1}\rangle + \frac{\eta^2 L}{2}\mathbb{E}_t\|\tilde{\mathbf{g}}_t\|^2 \\ &\overset{(18)}{\leq} -\eta\|\nabla F(\mathbf{x}_t)\|^2 + \frac{\eta(1-\kappa)}{2}\|\nabla F(\mathbf{x}_t)\|^2 + \frac{\eta(1-\kappa)}{2}\|\Delta_{t-1}\|^2 + \frac{\eta^2 L}{2}\mathbb{E}_t\|\tilde{\mathbf{g}}_t\|^2 \\ &\overset{(b)}{=} -\eta\|\nabla F(\mathbf{x}_t)\|^2 + \frac{\eta(1-\kappa)}{2}\|\nabla F(\mathbf{x}_t)\|^2 + \frac{\eta(1-\kappa)}{2}\|\Delta_{t-1}\|^2 \\ &\quad + \frac{\eta^2 L}{2}\mathbb{E}_t\|\tilde{\mathbf{g}}_t - \nabla F(\mathbf{x}_t) + \nabla F(\mathbf{x}_t)\|^2 \\ &\overset{(c)}{\leq} -\frac{\eta(1+\kappa-2\eta L)}{2}\|\nabla F(\mathbf{x}_t)\|^2 + \frac{\eta(1-\kappa)}{2}\|\Delta_{t-1}\|^2 + \eta^2 L\,\mathbb{E}_t\|\Delta_t\|^2,\end{aligned} \tag{27}$$

where (a) substitute (26); (b) we add and subtract $\nabla F(\mathbf{x}_t)$ to the last term and (c) applies (19) to the last term.

Define $\mathcal{L}_t := F(\mathbf{x}_t) + \beta\|\Delta_{t-1}\|^2$, we have:

$$\begin{aligned}&\mathbb{E}_t[\mathcal{L}_{t+1}] - \mathcal{L}_t \\ &\quad\leq -\frac{\eta(1+\kappa-2\eta L)}{2}\|\nabla F(\mathbf{x}_t)\|^2 - (\beta - \frac{\eta(1-\kappa)}{2})\|\Delta_{t-1}\|^2 + (\beta+\eta^2 L)\,\mathbb{E}_t\|\Delta_t\|^2 \\ &\quad\overset{(a)}{\leq} -\frac{\eta(1+\kappa-2\eta L)}{2}\|\nabla F(\mathbf{x}_t)\|^2 - (\beta - \frac{\eta(1-\kappa)}{2})\|\Delta_{t-1}\|^2 \\ &\qquad + (\beta+\eta^2 L)(1-\kappa)^2(1 + 4L^2\eta^2 + |1+\gamma|\left(\kappa + 2\eta^2 L^2 C_\gamma\right))\|\Delta_{t-1}\|^2 \\ &\qquad + (\beta+\eta^2 L)\kappa^2\left(\frac{(2+|1+\gamma|)\sigma_{\text{SGD}}^2}{B} + d\sigma_{\text{DP}}^2\right) \\ &\qquad + 2(\beta+\eta^2 L)(1-\kappa)^2 L^2\eta^2\left(2 + |1+\gamma|\,C_\gamma\right)\|\nabla F(\mathbf{x}_{t-1})\|^2 \\ &\quad= -\frac{\eta(1+\kappa-2\eta L)}{2}\|\nabla F(\mathbf{x}_t)\|^2 + 2(\beta+\eta^2 L)(1-\kappa)^2 L^2\eta^2\left(2+|1+\gamma|\,C_\gamma\right)\|\nabla F(\mathbf{x}_{t-1})\|^2 \\ &\qquad - \left(\beta - \frac{\eta(1-\kappa)}{2} - (\beta+\eta^2 L)(1-\kappa)^2(1 + 4L^2\eta^2 + |1+\gamma|\left(\kappa + 2\eta^2 L^2 C_\gamma\right))\right)\|\Delta_{t-1}\|^2\end{aligned}$$

$$+ (\beta + \eta^2 L)\kappa^2 \left(\frac{(2 + |1+\gamma|)\sigma_{\text{SGD}}^2}{B} + d\sigma_{\text{DP}}^2 \right), \tag{28}$$

where (a) applies Lemma 7 and (b) rearrange the terms. By choosing

$$\eta < \frac{1}{2L(1 + 2(1-\kappa)^2 \beta L (2 + |1+\gamma|\, C_\gamma))},\ \kappa > 1 - \frac{1}{\sqrt{1 + 4\eta^2 L^2 + |1+\gamma|\,(\kappa + 2\eta^2 L^2 C_\gamma)}},$$
$$\beta \geq \frac{\eta(1-\kappa)/2 + \eta^2 L (1-\kappa)^2 (1 + 4\eta^2 L^2 + |1+\gamma|\,(\kappa + 2\eta^2 L^2 C_\gamma))}{1 - (1-\kappa)^2 (1 + 4\eta^2 L^2 + |1+\gamma|\,(\kappa + 2\eta^2 L^2 C_\gamma))},$$

we have:

$$\begin{aligned}
&\frac{\eta(1 + \kappa - 2\eta L)}{2} - 2(\beta + \eta^2 L)(1-\kappa)^2 L^2 \eta^2 \left(2 + |1+\gamma|\, C_\gamma\right) > 0, \\
&1 - (1-\kappa)^2 (1 + 4\eta^2 L^2 + |1+\gamma|\,(\kappa + 2\eta^2 L^2 C_\gamma)) > 0, \\
&\beta - \frac{\eta(1-\kappa)}{2} - (\beta + \eta^2 L)(1-\kappa)^2 (1 + 4L^2\eta^2 + |1+\gamma|\,(\kappa + 2\eta^2 L^2 C_\gamma)) \geq 0.
\end{aligned} \tag{29}$$

Average from $t = 0$ to $T - 1$ and rearrange the terms, we have:

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T} \mathbb{E}\left\|\nabla F(\mathbf{x}_t)\right\|^2 &\leq \frac{2(\mathcal{L}_0 - \mathbb{E}[\mathcal{L}_{T+1}])}{C_1 \eta T} + \frac{2(\beta + \eta^2 L)\kappa^2}{C_1 \eta} \left(\frac{(2 + |1+\gamma|)\sigma_{\text{SGD}}^2}{B} + d\sigma_{\text{DP}}^2 \right) \\
&\leq \frac{2(F(\mathbf{x}_0) + \beta \left\|\nabla F(\mathbf{x}_0)\right\|^2 - F^\star)}{C_1 \eta T} + \frac{2(\beta + \eta^2 L)\kappa^2}{C_1 \eta} \left(\frac{(2 + |1+\gamma|)\sigma_{\text{SGD}}^2}{B} + d\sigma_{\text{DP}}^2 \right),
\end{aligned} \tag{30}$$

where we define $C_1 := (1 + \kappa - 2\eta L) - 4(\beta + \eta^2 L)(1-\kappa)^2 L^2 \eta \left(2 + |1+\gamma|\, C_\gamma\right)$, and in the last inequality we notice that $\mathcal{L}_{T+1} = F(\mathbf{x}_{T+1}) + \beta \left\|\Delta_T\right\|^2 \geq F^\star$, and $\mathcal{L}_0 = F(\mathbf{x}_0) + \beta \left\|\nabla F(\mathbf{x}_0)\right\|^2$, as we initialize $\tilde{\mathbf{g}}_0 = 0$. This completes the proof of Theorem 3.

On the choice of $\gamma = -1$. From the above proof, we see that in Appendix B.1, (21) (c), we directly apply (18) to upper bound the cross-product terms by positive terms for the worst case, which results in the optimal choice of $\gamma = -1$. However, the inner products may be smaller than zero in some cases, making $\gamma = -1$ sub-optimal in practice.

Appendix C. Additional numerical results

C.1. Experiment details

Dataset: We train the models on one synthetic dataset, four CV datasets, including MNIST [14], CIFAR-10/CIFAR-100 [22], and Imagenet-1k [13], and three NLP dataset, including GLUE [39], E2E [31], and DART [30].

Model: For the CV tasks, we use three different models, including a 5-layer CNN, WideResNet (WRN) [12], and ViT [15], representing three typical CV model structures. For the NLP task, we use the RoBERTa model [25]. For pre-training, the models are initialized with random weights, and for fine-tuning with ViT and RoBERTa, we directly use the checkpoints on HuggingFace [44].

Algorithm: We use the differentially private version of SGD, Adam for CV tasks, and AdamW for NLP tasks as base algorithm and apply DiSK to compare their performance. Additional results on LoRA [24] are given in Appendix C.4. In the results, we use **KF-** to denote the DP algorithm with DiSK.

Coding: The code for the experiments will be provided online. We use PyTorch as the code base and the FastDP package [7] to privatize the optimizers. We use the Renyi differential privacy (RDP) accountant in the Opacus and FastDP packages to numerically calculate the required injected DP noise to the gradient for fixed (ϵ, δ)-DP budget. A detailed derivation of the RDP accountant can be found in Wang et al. [42]. The Algorithm 2 is implemented as a PyTorch optimizer, which can be easily combined with any training scripts based on PyTorch. The modification is minimum:

```
from KFOptimizer import KFOptimizer
# define base optimizer
optimizer = KFOptimizer(model.parameters(), base_optimizer, kappa, gamma
    )
# ...
# in training loop:
    def closure(): # warp up the loss and backward computation
        loss = model(input)
        loss.backward()
        return loss
    loss = optimizer.prestep(closure)
    # ...
```

Hardware: All the experiments except the Imagenet-1k dataset are running with one Nvidia A40 (48GB memory) or one Nvidia V100 (32GB memory). The experiment on the Imagenet-1k dataset is running on one Nvidia H100 (80GB memory) GPU. The training time varies for different tasks depending on the data size and model size.

Training method: We use gradient accumulation to deal with the large batch size and use learning rate warm-up for $1/20$ of the training steps when training from randomly initialized weights. We also use the Cosine Annealing learning rate scheduler [26], which gradually decreases the learning rate.

C.2. Choice of hyper-parameters

The main hyper-parameters in the algorithms are: epoch E, batch size B, step size η, clipping threshold C, Kalman filter parameters κ, and γ. In all experiments, we fix the clipping method as automatic clipping used in Bu et al. [8], i.e., $\text{clip}\left(\nabla f(\mathbf{x};\xi), C\right) = \nabla f(\mathbf{x};\xi)\frac{C}{\|\nabla f(\mathbf{x};\xi)\|}$, and set $C = 1$ for all experiments. We fix $\delta = 1/N^{1.1}$ for reasonable privacy notions. This choice matches or is tighter than the SOTA results using $\delta = 1/2N$ or $1/N^{1.1}$ [8, 24, 46]. We list the δ's used in the experiments in Tab. 6.

For each set of experiments, we conduct a grid search on the hyper-parameters E, B, η and choose the optimal ones for the DP optimizer without DiSK. The search grids of each hyper-parameter are listed in Table 2;

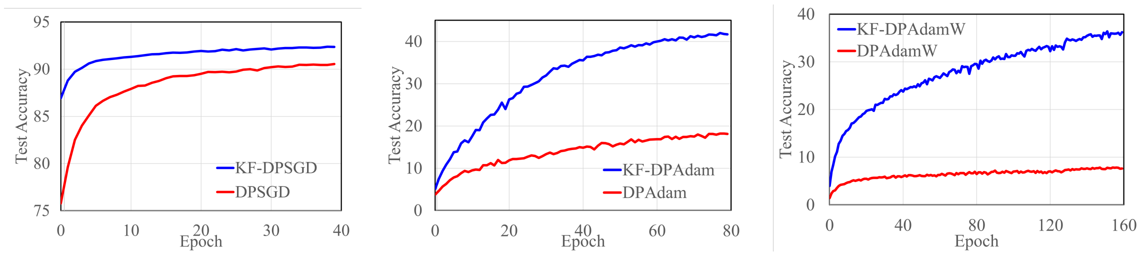
Then, we fix $E, B.\eta$ and conduct the ablation study on κ, γ as shown in Figure 6.

C.3. Additional experiments on CV tasks

Training different models from scratch: The test accuracy curves during the training for 5-layer CNN on the MNIST dataset, WRN-16-4 on the CIFAR-100 dataset, and ViT-small on the Imagenet-1k dataset are given in Figure 2. The optimizer with DiSK convergence faster than the base algorithm on all tasks and reaches a higher final accuracy at a given privacy budget. The test accuracy

	Search gird		
	MNIST	CIFAR	Imagenet
E	$\{1, \mathbf{2}, 3\} \times 20$	$\{1, \mathbf{2}, 3, 4\} \times 40$	$\{3, \mathbf{4}\} \times 40$
B	$\{2, \mathbf{5}\} \times 10^3$	$\{0.5, 1, 2, \mathbf{5}\} \times 10^3$	$\{\mathbf{5}, 10\} \times 10^3$
η	$\{3, \mathbf{2.5}, 1, 0.3, 0.1\} \times 10^{-1}$	$\{1, 3, \mathbf{5}, 7, 10\} \times 10^{-3}$	$\{10, 3, 1, \mathbf{0.3}, 0.1\} \times 10^{-3}$
κ	$\{\mathbf{0.7}\}$	$\{9.9, 9, 8, \mathbf{7}, 6, 5\} \times 10^{-1}$	$\{\mathbf{0.7}\}$
γ	$\{\mathbf{0.5}\}$	$\{0.2, 0.3, \mathbf{0.5}, 1, 2, 3, \frac{1-\kappa}{\kappa}\}$	$\{\mathbf{0.5}\}$

Table 2: Search grid of the CV pre-training experiments, the optimal hyper-parameters are in **bold**.

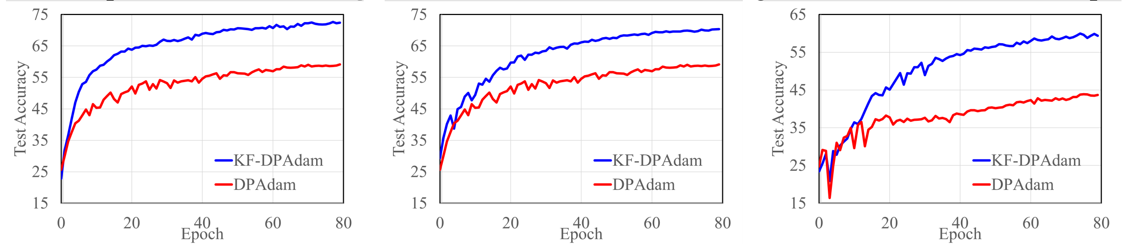


(a) MNIST, $\epsilon = 0.5$ (b) CIFAR-100, $\epsilon = 8$ (c) Imagenet-1k, $\epsilon = 8$

Figure 2: Test accuracy of pre-training on MNIST, CIFAR-100, and Imagenet-1k datasets with and without DiSK for fixed privacy budgets.

of CIFAR-100 achieves 41.8%, and Imagenet-1k achieves 36.4%, which outperforms the SOTA results that apples data augmentation under the same privacy budget (40.6% for CIFAR-100 [5] and 32.4% for Imagenet-1k [12]).

We additionally train the WRN-16-4 and the ViT-small models on the CIFAR-10 with randomly initialized weights for different privacy budgets, and the test accuracies during the training are shown in Figure 3 for $\epsilon = 4$. From the results, we can see that DiSK consistantly outperforms the base optimizer. **Fine-tuning on CIFAR-100:** Besides training from scratch, we also compare



(a) 5-layer CNN (b) WRN-16 (c) ViT-small

Figure 3: Test accuracy of pre-training 5-layer CNN, WRN-16, and ViT-small on CIFAR-10 dataset with and without DiSK for fixed privacy budget $\epsilon = 4$.

the performance of fine-tuning a pre-trained ViT-small model on the CIFAR-100 dataset. The results for different ϵ are shown in Figure 4. For fine-tuning on the complex CIFAR-100 dataset, DiSK still improves he performance compared with DPAdam, and has less performance drop under large DP noise.

Comparison with existing methods: We conduct comparisons with existing approaches for improving DP training performance. In Figure 5(a)subfigure, we train a linear regression model

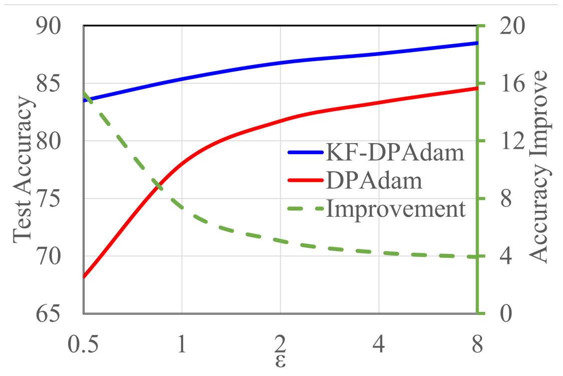
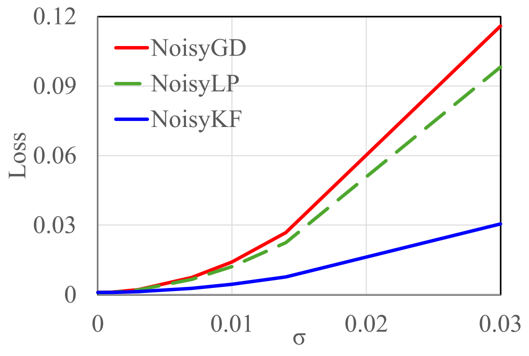
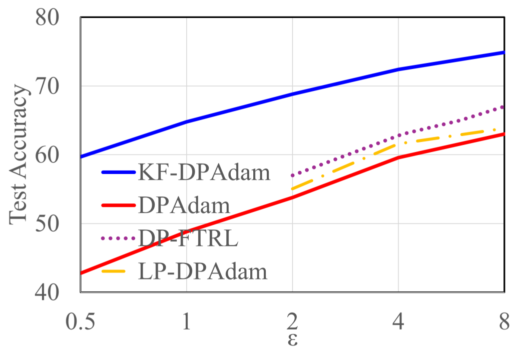


Figure 4: Fine-tuning ViT-small on CIFAR-100 with different ϵ.



(*a*) Synthetic data



(*b*) CIFAR-10

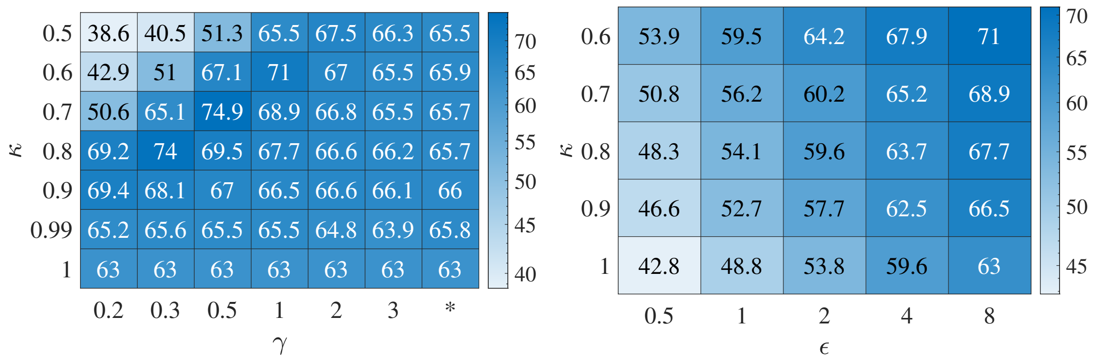
Figure 5: Comparison with existing approaches. a) Kalman filter and low-pass filter; b) Kalman filter, DP-FTRL, and low-pass filter.

with synthetic data and compared the performance of Noisy GD, Noisy GD with DOPPLER (NoisyLP), and with DiSK (NoisyKF). We inject Gaussian noise with different variances into the gradient and compare the final performance. We observe that DiSK has the lowest regression loss under all noise levels, indicating that the Kalman filter performs better in noise reduction than the Low-pass filter. In Figure 5(*b*)subfigure, we compare the test accuracy of different methods, including DOPPLER [49] and DP-FTRL [10, 19] on the CIFAR-10 dataset training the WRN from scratch. We observe that DiSK significantly outperforms the SOTA algorithms on all privacy budgets.

Ablation study: We conduct ablation studies on the choice of the hyper-parameters of DiSK, specifically, how κ, γ impact the algorithm performance. In Figure 6, we plot the accuracy on different combinations of (κ, γ), and (κ, ϵ). We observe a clear trend of performance change for different combinations of the parameters, and there is an optimal choice of κ, γ for different ϵ's.

C.4. Additional experiments on NLP tasks

In this section, we provide additional results for NLP tasks.



(*a*) $(\kappa, \gamma), \epsilon = 8$ (*b*) $(\kappa, \epsilon), \gamma = 1$

Figure 6: Test accuracy for different combinations of the hyper-parameters when training CNN on the CIFAR-10 dataset.

Parameter-efficient gine-tuning on GLUE. We fine-tune a RoBERTa-base and a RoBERTa-large model from the Huggingface checkpoints[1] on the GLUE dataset. We follow the same training scripts in Bu et al. [8] on the hyper-parameter choices of η, B, E on the tasks and use rank $r = 16$ for LoRA. We choose $\kappa = 0.7, \gamma = 0.5$ for DiSK. The results are listed in Table 3. With privacy budget $\epsilon = 1, 6.7$, DPLoRA with DiSK significantly outperforms SOTA results with vanilla DPLoRA on all tasks.

Table 3: Test accuracy of fine-tuning result on the GLUE dataset.

	$\epsilon = 1$				$\epsilon = 6.7$			
Algorithm	MNLI	QNLI	SST2	QQP	MNLI	QNLI	SST2	QQP
RoBERTa-base								
AdamW ($\epsilon = \infty$)	87.6	92.8	94.8	91.9	87.6	92.8	94.8	91.9
Lora ($\epsilon = \infty$)	87.5	93.3	95.1	90.8	87.5	93.3	95.1	90.8
DPLora	81.1	85.5	90.9	83.9	83.5	87.4	91.5	85.7
KF-DPLora	84.7	90.3	92.9	87.8	85.9	90.5	93.1	89.0
RoBERTa-large								
AdamW ($\epsilon = \infty$)	90.3	94.7	96.4	92.2	90.3	94.7	96.4	92.2
Lora ($\epsilon = \infty$)	90.6	94.9	96.2	91.6	90.6	94.9	96.2	91.6
DPLora	85.6	89.5	90.9	85.1	87.8	90.8	94.3	87.4
KF-DPLora	87.9	92.5	95.2	88.2	89.4	92.6	95.4	89.6

Fine-tuning GPT-2 on text generation tasks. We fine-tune a GPT-2-small model with $137M$ parameters from the Huggingface checkpoints[2] on two text generation datasets, E2E and DART. We follow the same training scripts in Li et al. [24] on the hyper-parameter choices of η, B, E on the tasks and choose $\kappa = 0.7, \gamma = 0.5$ for DiSK. The results on different metrics for the E2E dataset are given in Table 4, and the results for the DART dataset are in Table 5. With privacy budget $\epsilon = 3, 8$, DPAdamW with DiSK significantly outperforms SOTA results with vanilla DPAdamW on all metrics.

1. https://huggingface.co/FacebookAI/roberta-base,https://huggingface.co/FacebookAI/roberta-large
2. https://huggingface.co/openai-community/gpt2

Table 4: Performance of fine-tuning gpt-2 on the E2E dataset. (All metrics are higher the better)

Algorithm	BLEU (%)	ROUGE-L (%)	METEOR	NIST	CIDEr
AdamW ($\epsilon = \infty$)	69.46	71.36	0.461	8.780	2.422
DPAdamW ($\epsilon = 3$)	61.52	65.87	0.417	7.071	2.167
KF-DPAdamW ($\epsilon = 3$)	68.35	70.23	0.456	8.636	2.399
DPAdamW ($\epsilon = 8$)	64.99	67.34	0.425	8.387	2.192
KF-DPAdamW ($\epsilon = 8$)	68.73	70.58	0.460	8.697	2.463

Table 5: Performance of fine-tuning gpt-2 on the DART dataset. Val. Perp. stands for validation perplexity. (All metrics except Val. Perp. are higher the better)

Algorithm	Val. Perp. $\downarrow$	BLEU (%)	ROUGE-L (%)	METEOR	NIST	CIDEr
AdamW ($\epsilon = \infty$)	0.921	44.56	58.66	0.379	8.733	2.773
DPAdamW ($\epsilon = 3$)	1.427	33.96	52.38	0.310	6.090	1.864
KF-DPAdamW ($\epsilon = 3$)	1.149	41.01	57.53	0.359	7.949	2.553
DPAdamW ($\epsilon = 8$)	1.362	35.30	54.58	0.320	6.365	1.995
KF-DPAdamW ($\epsilon = 8$)	1.102	42.12	58.11	0.364	8.111	2.628

Table 6: The privacy parameter δ's used in our experiments and in SOTA results.

Dataset	Our δ	SOTA δ
MNIST	5.5×10^{-6}	10^{-5}
CIFAR-10/100	6.8×10^{-6}	10^{-5}
Imagenet-1k	1.9×10^{-7}	8×10^{-7}
MNLI	6.3×10^{-7}	1.1×10^{-6}
QNLI	4.8×10^{-7}	9×10^{-7}
SST-2	4.9×10^{-6}	7.4×10^{-6}
QQP	7.6×10^{-7}	1.4×10^{-6}
E2E	1.2×10^{-5}	1.2×10^{-5}
DART	6.1×10^{-6}	6.1×10^{-6}