PRIVATE ZEROTH-ORDER NONSMOOTH NONCONVEX OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a new zeroth-order algorithm for private stochastic optimization on nonconvex and nonsmooth objectives. Given a dataset of size M, our algorithm ensures $(\alpha, \alpha\rho^2/2)$-Rényi differential privacy and finds a (δ, ϵ)-stationary point so long as $M = \tilde{\Omega}\left(\frac{d}{\delta\epsilon^3} + \frac{d^{3/2}}{\rho\delta\epsilon^2}\right)$. This matches the optimal complexity of its non-private zeroth-order analog. Notably, although the objective is not smooth, we have privacy "for free" whenever $\rho \geq \sqrt{d}\epsilon$.

1 INTRODUCTION

We study the stochastic optimization problem of the form

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} F(\boldsymbol{x}) = \mathbb{E}_z[f(\boldsymbol{x}, z)],$$

where the stochastic function $f(\boldsymbol{x}, z)$ might be both non-convex and non-smooth with respect to $\boldsymbol{x}$. Our focus is on zeroth-order algorithms, often referred to as gradient-free algorithms. Unlike first-order algorithms that have access to the stochastic gradient $\nabla f(\boldsymbol{x}, z)$, zeroth-order algorithms can access only the function value $f(\boldsymbol{x}, z)$.

Non-convex stochastic optimization is fundamental to modern machine learning. For example, in deep learning, $\boldsymbol{x}$ is the weight of some neural network model, z is a datapoint, and $f(\boldsymbol{x}, z)$ represents the loss of the model evaluated on $\boldsymbol{x}$ and z. Consequently, training the machine learning model equates to minimizing the non-convex objective $F(\boldsymbol{x})$. Given its significant role, there has been a marked increase in research focusing on non-convex optimization in recent years (Ghadimi & Lan, 2013; Arjevani et al., 2019; 2020; Carmon et al., 2019; Fang et al., 2018; Cutkosky & Orabona, 2019). Since finding a global minimum of non-convex $F(\boldsymbol{x})$ is intractable, many previous works assume that F is smooth, and their goal is to find an ϵ-stationary point $\boldsymbol{x}$ where $\|\nabla F(\boldsymbol{x})\| \leq \epsilon$. However, objectives are not always smooth, especially with the ever-increasing complexity of modern machine learning models. Making the problem even more difficult, recent research shows that finding an ϵ-stationary point of a non-smooth objective might be intractable (Zhang et al., 2020; Kornowski & Shamir, 2022).

To address this issue, Zhang et al. (2020) introduces an alternative objective for the convergence analysis of non-smooth non-convex optimization. Specifically, they propose the notion of Goldstein stationary point (Goldstein, 1977): $\boldsymbol{x}$ is said to be an (δ, ϵ)-stationary point of F if there exists a subset S in the ball of radius δ centered at $\boldsymbol{x}$ such that $\|\mathbb{E}[\nabla F(\boldsymbol{y})]\| \leq \epsilon$ where $\boldsymbol{y}$ is uniformly distributed over S. Recently, there have been many works on first-order algorithms using this framework (Zhang et al., 2020; Tian et al., 2022; Cutkosky et al., 2023). Notably, Cutkosky et al. (2023) designs an "Online-to-non-convex Conversion" algorithm that finds a (δ, ϵ)-stationary point with $O(\delta^{-1}\epsilon^{-3})$ samples, which is proven to be the optimal rate. However, first-order algorithms have their own limitations. For many applications, computing first-order gradients can be computationally expensive or even impossible (Nelson, 2010; Mania et al., 2018). Hence, there are many recent results focusing on designing efficient zeroth-order algorithms for non-smooth non-convex optimization (Lin et al., 2022; Chen et al., 2023; Kornowski & Shamir, 2023). Recently, Kornowski & Shamir (2023) proposes a zeroth-order algorithm that requires only $O(d\delta^{-1}\epsilon^{-3})$ samples, which is the current state-of-the-art rate for zeroth-order optimization on non-smooth non-convex objectives. The additional dimension dependence is an inevitable cost of zeroth-order algorithms even when the objective is smooth or convex (Duchi et al., 2015).

In this paper, we study zeroth-order stochastic optimization on non-convex and non-smooth objectives. Zeroth-order optimization is particularly important in cases where memory is constrained or the model is excessively large so that computing a backwards pass is forbiddingly expensive. In addition, we also aim to protect the privacy of user's data. For example, consider the practical problem of training a recommendation model or a chat response model. At each step the algorithm is given a set of different users who report the model's performance in the form of "upvotes" or "downvotes". It's important to make updates using such zeroth-order feedback while preserving the privacy of feedback from the individual users.

To this end, we require our algorithm to be *differentially private* (DP) (Dwork et al., 2006), which means with high probability, the output of our algorithm operating on any particular dataset is *almost* indistinguishable from the output when one datapoint in the dataset is perturbed.

The primary objective of private stochastic optimization is to minimize the dataset size required to solve the optimization problem while maintaining differential privacy guarantees. This area has seen extensive research efforts. While problems with convex objectives have been well-studied over the past decade (Chaudhuri et al., 2011; Jain et al., 2012; Kifer et al., 2012; Bassily et al., 2014; Talwar et al., 2014; Jain & Thakurta, 2014; Bassily et al., 2019; Feldman et al., 2020; Bassily et al., 2021b; Zhang et al., 2022), more recent efforts have focused on private non-convex optimization (Wang et al., 2017; 2019a; Tran & Cutkosky, 2022; Wang et al., 2019b; Zhou et al., 2020; Bassily et al., 2021a; Arora et al., 2023). Recently, Arora et al. (2023) designs an algorithm that, assuming the objective is smooth, finds an ϵ-stationary point and satisfies (ρ, γ)-DP with $\tilde{O}(\epsilon^{-3}+\sqrt{d}\rho^{-1}\epsilon^{-2})$ data complexity. While private non-convex optimization has seen rapid advancements, certain challenges remain. Many previous studies hinge on the assumption of smooth objectives since their methods are essentially extensions of non-private non-convex optimization algorithms. In fact, this limitation emerges even in situations where the objective is convex, in which case smoothness provides a critical tool for reducing sensitivity. Since the results in non-smooth non-convex optimization are recent, its differentially private counterpart remains largely unexplored.

1.1 Our Contributions

The main contribution of this paper is the introduction of a zeroth-order algorithm for differentially private stochastic optimization on non-convex and non-smooth objectives. To our knowledge, this is the first work under this framework. Our algorithm satisfies $(\alpha, \alpha\rho^2/2)$-Rényi differential privacy (RDP) (Mironov, 2017) (which is approximately (ρ, γ)-DP) and finds a (δ, ϵ)-stationary point with $\tilde{O}(d\delta^{-1}\epsilon^{-3}+d^{3/2}\rho^{-1}\delta^{-1}\epsilon^{-2})$ data complexity. Notably, the non-private term $O(d\delta^{-1}\epsilon^{-3})$ matches the state-of-the-art complexity found in its non-private counterpart. Moreover, when $\rho \geq \sqrt{d}\epsilon$, the non-private term is dominating, suggesting that privacy can be attained without additional cost. This is consistent with the observations when the objective is smooth (Arora et al., 2023).

Our algorithm incorporates four essential components, each crucial for achieving the optimal rate. We leverage the non-private Online-to-non-convex Conversion (O2NC) framework by Cutkosky et al. (2023), which finds a (δ, ϵ)-stationary point of a non-smooth objective using a first-order oracle. We then build an approximate first-order oracle (i.e. a gradient estimator) with a zeroth-order oracle. Although this high-level strategy is a common technique, our approach distinguishes itself through its gradient oracle design. In contrast to prior non-private approaches that directly approximate the gradient with a standard zeroth-order estimator (Kornowski & Shamir, 2023), we introduce a variance-reduced gradient oracle. Our approach incorporates two zeroth-order estimators: one for the gradient and another for its difference between two points. Our gradient oracle also differs subtly from the standard zeroth-order estimator by sampling d i.i.d. estimators for each data point, which is required to achieve the optimal dimension dependence. Both estimators exhibit reduced sensitivity. We reduce the privacy cost of our variance-reduced estimators by using the "tree mechanism" (Dwork et al., 2010; Chan et al., 2011), yielding a new state-of-the-art privacy guarantee.

Omitting any component from our algorithm significantly impacts the results. Neither utilizing the O2NC technique nor sampling d i.i.d. estimators results in an additional dimension dependence of $\sqrt{d}$. Moreover, the combination of the variance-reduced oracle and the tree mechanism is crucial for optimal privacy. A naive approach might directly make O2NC private adding noise to classical zeroth-order gradient estimators, but this yields a sub-optimal rate with an extra factor of ϵ^{-1}. More details about the intuitions and challenges of our algorithm can be found in Section 4.1.

2 PRELIMINARY

Notation We use bold font $\boldsymbol{x}$ to denote a vector $\boldsymbol{x} \in \mathbb{R}^d$, and denote the Euclidean norm of $\boldsymbol{x}$ by $\|\boldsymbol{x}\|$. We denote the unit ball and unit sphere in $\mathbb{R}^d$ by $\mathbb{B}$ and $\mathbb{S}$, and denote the uniform distribution on $\mathbb{B}$ and $\mathbb{S}$ by $\mathcal{U}_{\mathbb{B}}$ and $\mathcal{U}_{\mathbb{S}}$ respectively. We denote an open ball in $\mathbb{R}^d$ centered at $\boldsymbol{x}$ of radius r by $B(\boldsymbol{x}, r) = \{\boldsymbol{y} : \|\boldsymbol{x} - \boldsymbol{y}\| < r\}$. For $n \in \mathbb{N}$, we denote the set $\{1, \dots, n\}$ by $[n]$, and we denote a sequence $a_1, \dots, a_n$ by $a_{1:n}$. We use the standard big-O notation, and use $\tilde{O}$ to hide additional logarithmic factors. We interchangeably use $f(x) \lesssim g(x)$ to denote $f(x) = \tilde{O}(g(x))$.

Non-smooth Optimization A function $h : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz if $|h(\boldsymbol{x}) - h(\boldsymbol{y})| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$; h is H-smooth if it is differentiable and $\|\nabla h(\boldsymbol{x}) - \nabla h(\boldsymbol{y})\| \leq H\|\boldsymbol{x} - \boldsymbol{y}\|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$. A point $\boldsymbol{x}^* \in \mathbb{R}^d$ is an ϵ-stationary point of h if $\|\nabla h(\boldsymbol{x}^*)\| \leq \epsilon$.

Definition 2.1. Let $\delta > 0$ and $h : \mathbb{R}^d \to \mathbb{R}$ be differentiable. The Goldstein δ-subdifferential of h at $\boldsymbol{x}$ is $\partial_\delta h(\boldsymbol{x}) = \mathrm{conv}(\cup_{\boldsymbol{y} \in B(\boldsymbol{x}, \delta)} \nabla h(\boldsymbol{y}))$. We define $\|\nabla h(\boldsymbol{x})\|_\delta = \inf\{\|\boldsymbol{g}\| : \boldsymbol{g} \in \partial_\delta h(\boldsymbol{x})\}$. Then $\boldsymbol{x}^*$ is said to be a Goldstein (δ, ϵ)-stationary point of h if $\|\nabla h(\boldsymbol{x}^*)\|_\delta \leq \epsilon$. Note that

$$\|\nabla h(\boldsymbol{x})\|_\delta \leq \inf_{S \subset B(\boldsymbol{x}, \delta)} \left\| \frac{1}{|S|} \sum_{\boldsymbol{y} \in S} \nabla h(\boldsymbol{y}) \right\|.$$

Differential Privacy A stochastic optimization algorithm can be considered as a randomized algorithm that takes a dataset Z (a collection of data points $z_1, \dots, z_M$) and outputs an output $\overline{\boldsymbol{w}}$. Throughout this paper, we denote $\mathcal{Z}$ as the set of all possible datasets of size M. Two datasets $Z, Z' \in \mathcal{Z}$ are neighboring if they differ only in one data point. A randomized algorithm $\mathcal{A} : \mathcal{Z} \to \mathcal{R}$ is said to be (ϵ, δ)-differentially private ((ϵ, δ)-DP) for $\epsilon, \delta > 0$ if for all neighboring datasets Z, Z' and measurable $E \subseteq \mathcal{R}$, we have $\mathbb{P}\{\mathcal{A}(Z) \in E\} \leq e^\epsilon \, \mathbb{P}\{\mathcal{A}(Z') \in E\} + \delta$ (Dwork et al., 2006).

Another common privacy measure is Rényi differential privacy (RDP). Algorithm $\mathcal{A}$ is (α, ρ)-RDP for $\alpha > 1, \rho > 0$ if for all neighboring Z, Z', $D_\alpha(\mathcal{A}(Z) \| \mathcal{A}(Z')) \leq \rho$ (Mironov, 2017), where $D_\alpha(\mu \| \nu)$ is the Rényi divergence of distributions μ, ν. RDP can be converted to (ϵ, δ)-DP as follows: if an algorithm is $(\alpha, \alpha\rho^2/2)$-RDP for all $\alpha > 1$, then it is also $(2\rho \ln(1/\delta)^{1/2}, \delta)$-DP for all $\delta \geq \exp(-\rho^2)$ (Mironov, 2017, Proposition 3). Therefore, we use $(\alpha, \alpha\rho^2/2)$-RDP as a measure of differential privacy in our paper.

If $\|\mathcal{A}(Z) - \mathcal{A}(Z')\| \leq s$ for any neighboring Z, Z', we say the sensitivity of $\mathcal{A}$ is bounded by s. It is well known that in this case, adding a Gaussian noise $\mathcal{N}(0, \sigma^2 I)$ to the output of $\mathcal{A}$, where $\sigma = s/\rho$, ensures that $\mathcal{A}$ is $(\alpha, \alpha\rho^2/2)$-RDP (Mironov, 2017).

We also make use of the "tree mechanism" Dwork et al. (2010); Chan et al. (2011), which is a technique that allows for private release of *running sums* of potentially sensitive data (Algorithm 5).

Online Learning Our algorithm builds on the Online-to-non-convex algorithm by Cutkosky et al. (2023), so we briefly introduce the setting of online convex optimization (OCO) (Cesa-Bianchi & Lugosi, 2006; Hazan, 2019; Orabona, 2019). An OCO algorithm proceeds in rounds. In each round t, it outputs $\boldsymbol{x}_t$, receives a convex loss function $\ell_t(\boldsymbol{x})$, and suffers loss $\ell_t(\boldsymbol{x}_t)$. It is common to use a linear loss $\ell_t(\boldsymbol{x}) = \langle \boldsymbol{v}_t, \boldsymbol{x} \rangle$. An OCO algorithm has domain bounded by D if $\|\boldsymbol{x}_t\| \leq D$ for all t.

The goal of online learning is to minimize the *static regret* defined as

$$\mathrm{Reg}_T(\boldsymbol{u}) = \sum_{t=1}^T \ell_t(\boldsymbol{x}_t) - \ell_t(\boldsymbol{u}) = \sum_{t=1}^T \langle \boldsymbol{v}_t, \boldsymbol{x}_t - \boldsymbol{u} \rangle,$$

where $\boldsymbol{u}$ is some comparator vector (often, $\boldsymbol{u} = \arg\min \sum_{t=1}^T \ell_t(\boldsymbol{x})$). There are many OCO algorithms that achieve the minimax optimal $O(\sqrt{T})$ regret. For example, projected Online Subgradient Descent (OSD) (Zinkevich, 2003) with domain bounded by D satisfies

$$\mathbb{E}[\mathrm{Reg}_T(\boldsymbol{x})] \leq D\sqrt{\sum_{t=1}^T \mathbb{E}\|\boldsymbol{v}_t\|^2}.$$

Uniform Smoothing Randomized smoothing is a well known technique that converts a possibly non-smooth function to a smooth approximation (Flaxman et al., 2005; Duchi et al., 2012; Lin et al., 2022). Given a Lipschitz function $h : \mathbb{R}^d \to \mathbb{R}$ and $\delta > 0$, we define the *uniform smoothing* of h as $\hat{h}_\delta(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{v} \sim \mathcal{U}_{\mathbb{B}}}[h(\boldsymbol{x} + \delta \boldsymbol{v})]$. In later sections, we denote $\hat{F}_\delta(\boldsymbol{x})$ and $\hat{f}_\delta(\boldsymbol{x}, z)$ as the uniform

smoothing of the objective $F(\boldsymbol{x})$ and $f(\boldsymbol{x}, z)$ respectively. As shown in prior works (see (Yousefian et al., 2012) and Duchi et al. (2012, Section E)), uniform smoothing is a smooth approximation whose gradient can be estimated by finite differentiation. We rephrase the key properties in the following lemma, whose proof is presented in the appendix for completeness.

Lemma 2.2. *Suppose $h : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz. Then (i) $\hat{h}_\delta$ is L-Lipschitz; (ii) $\|\hat{h}_\delta(\boldsymbol{x}) - h(\boldsymbol{x})\| \leq L\delta$; (iii) $\hat{h}_\delta$ is differentiable and $\frac{\sqrt{d}L}{\delta}$-smooth; (iv)*

$$\nabla \hat{h}_\delta(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{u}\sim\mathcal{U}_\mathbb{S}}[\tfrac{d}{\delta} h(\boldsymbol{x} + \delta \boldsymbol{u})\boldsymbol{u}] = \mathbb{E}_{\boldsymbol{u}\sim\mathcal{U}_\mathbb{S}}[\tfrac{d}{2\delta}(h(\boldsymbol{x} + \delta \boldsymbol{u}) - h(\boldsymbol{x} - \delta \boldsymbol{u}))\boldsymbol{u}];$$

As a corollary, finding an (δ, ϵ)-stationary point of $\hat{F}_\delta$ is sufficient to find an $(2\delta, \epsilon)$-stationary point of F. The formal statement is as follows (Kornowski & Shamir (2023, Lemma 4); proof is presented in Appendix A for completeness).

Corollary 2.3. *Suppose $F : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz. Then for any $\epsilon, \delta > 0$, $\|\nabla \hat{F}_\delta(\boldsymbol{x})\|_\delta \leq \epsilon$ implies that $\|\nabla F(\boldsymbol{x})\|_{2\delta} \leq \epsilon$.*

2.1 Analysis Organization

As a brief overview, our algorithm leverages O2NC to privately optimize the uniform smoothing of the objective. Note that O2NC does not require any smoothness itself - we employ uniform smoothing because it allows us to construct a low-sensitivity gradient oracle from a zeroth-order oracle. The smoothness property is tangential. We construct a variance-reduced gradient oracle for O2NC and incorporate the tree mechanism for enhanced privacy. This oracle is based on two zeroth-order estimators: one for the stochastic gradient of the uniform smoothing and another for the gradient difference.

This paper is organized in the following order. In Section 3, we introduce the zeroth-order estimators. In Section 4.1, we present the O2NC algorithm combined with the private variance-reduced gradient oracle. In Section 4.2, we demonstrate the improved privacy guarantee provided by the tree mechanism. We conclude with the convergence analysis in Section 4.3.

3 Zeroth-Order Estimators

Let $F(\boldsymbol{x}) = \mathbb{E}_z[f(\boldsymbol{x}, z)]$ and denote $\hat{F}_\delta$ and $\hat{f}_\delta$ as the uniform smoothing of F and f respectively. Using the randomized smoothing technique from Lemma 2.2, we can construct unbiased zeroth-order estimators for both $\nabla \hat{F}_\delta(\boldsymbol{x})$ and $\nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})$. The key properties of these estimators Grad and Diff, as defined in Algorithm 1 and 2, are summarized in the following results.

Lemma 3.1. *If $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$, then for any $\delta > 0, \boldsymbol{x} \in \mathbb{R}^d$ and neighboring data batches $z_{1:b}, z'_{1:b}$ of size b,*

$$\mathbb{E}[\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b})] = \nabla \hat{F}_\delta(\boldsymbol{x}), \tag{unbiased}$$

$$\mathbb{E}\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2 \leq \tfrac{16dL^2}{b}, \tag{variance}$$

$$\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \textsc{Grad}_{f,\delta}(\boldsymbol{x}, z'_{1:b})\| \leq \tfrac{2dL}{b}. \tag{sensitivity}$$

Lemma 3.2. *If $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$, then for any $\delta > 0$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ and neighboring data batches $z_{1:b}, z'_{1:b}$ of size b,*

$$\mathbb{E}[\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b})] = \nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y}), \tag{unbiased}$$

$$\mathbb{E}\|\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - [\nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})]\|^2 \leq \tfrac{16dL^2}{b\delta^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2, \tag{variance}$$

$$\|\textsc{Diff}_f(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - \textsc{Diff}_f(\boldsymbol{x}, \boldsymbol{y}, z'_{1:b})\| \leq \tfrac{2dL}{b\delta}\|\boldsymbol{x} - \boldsymbol{y}\|. \tag{sensitivity}$$

Proofs for Lemmas 3.1 and 3.2 are both stemming from the uniform smoothing properties detailed in Lemma 2.2. We defer the proofs to Appendix B. Notably, these lemmas indicate that both the gradient and gradient difference estimators are unbiased, with low variance and sensitivity. Specifically, in our final algorithm, we will be able to control the distance between parameters $\boldsymbol{x}, \boldsymbol{y}$ in two successive iterations by δ/T. As a result, the variance in Lemma 3.2 is bounded by $O(dL^2/bT^2)$, and

Algorithm 1 Zeroth-order gradient oracle $\text{GRAD}_{f,\delta}(\boldsymbol{x}, z_{1:b})$

Input: loss function f, constant $\delta > 0$, parameter $\boldsymbol{x}$, i.i.d. data batch $z_{1:b}$

1: Sample $\boldsymbol{u}_{11}, \ldots, \boldsymbol{u}_{bd} \sim \text{Uniform}(\mathbb{S})$ i.i.d.

2: Return $\text{GRAD}_{f,\delta}(\boldsymbol{x}, z_{1:b}) = \frac{1}{b}\sum_{i=1}^{b}\frac{1}{d}\sum_{j=1}^{d}\frac{d}{2\delta}(f(\boldsymbol{x}+\delta\boldsymbol{u}_{ij}, z_i) - f(\boldsymbol{x}-\delta\boldsymbol{u}_{ij}, z_i))\boldsymbol{u}_{ij}$.

Algorithm 2 Zeroth-order gradient difference oracle $\text{DIFF}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b})$

Input: loss function f, constant $\delta > 0$, parameter $\boldsymbol{x}, \boldsymbol{y}$, i.i.d. data batch $z_{1:b}$

1: Sample $\boldsymbol{u}_{11}, \ldots, \boldsymbol{u}_{bd} \sim \text{Uniform}(\mathbb{S})$ i.i.d.

2: Return $\text{DIFF}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) = \frac{1}{b}\sum_{i=1}^{b}\frac{1}{d}\sum_{j=1}^{d}\frac{d}{\delta}(f(\boldsymbol{x}+\delta\boldsymbol{u}_{ij}, z_i) - f(\boldsymbol{y}+\delta\boldsymbol{u}_{ij}, z_i))\boldsymbol{u}_{ij}$.

its sensitivity is bounded by $O(dL/bT)$. Using these estimators, we construct zeroth-order gradient oracles by exploiting the decomposition $\nabla\hat{F}_\delta(\boldsymbol{x}_t) = \nabla\hat{F}_\delta(\boldsymbol{x}_1) + \sum_{i=1}^{t}\nabla\hat{F}_\delta(\boldsymbol{x}_i) - \nabla\hat{F}_\delta(\boldsymbol{x}_{i-1}) \approx \text{GRAD}_{f,\delta}(\boldsymbol{x}_1, z) + \sum_{i=1}^{t}\text{DIFF}_{f,\delta}(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}, z)$. Our constructed oracles have low variance and low sensitivity, thus allowing us to incorporate the tree mechanism to privately aggregate the sum with less noise. A more detailed exploration of this idea is presented in the subsequent sections, especially in Corollary 4.4 and Lemma 4.8.

In addition, we would like to discuss the rationale behind sampling d i.i.d. uniform vectors $\boldsymbol{u}_{ij}$ for each data point z_i. For the gradient estimator GRAD, it is feasible to approximate $\nabla\hat{f}_\delta(\boldsymbol{x}, z_i)$ with a single two-point estimator, namely $\boldsymbol{g}_i = \frac{d}{2\delta}(f(\boldsymbol{x}+\delta\boldsymbol{u}, z_i) - f(\boldsymbol{x}-\delta\boldsymbol{u}, z_i))\boldsymbol{u}$, as it is proved that $\mathbb{E}\|\boldsymbol{g}_i\|^2 = O(dL^2)$ (Shamir, 2017, Lemma 10). The central argument hinges on a concentration inequality for Lipschitz functions, which states that if h is L-Lipschitz on $\boldsymbol{u}$ and $\boldsymbol{u} \sim \mathcal{U}_{\mathbb{S}}$, then $\mathbb{P}\{|h(\boldsymbol{u}) - \mathbb{E}h(\boldsymbol{u})| \geq t\} \leq 2\exp(-\frac{dt^2}{2L^2})$, as per (Wainwright, 2019) Proposition 3.11. In the context of their proof, $h(\boldsymbol{u}) = f(\boldsymbol{x}+\delta\boldsymbol{u})$ which is $L\delta$-Lipschitz.

However, this argument isn't applicable to the gradient difference estimator DIFF. In order to apply the concentration inequality, we need to bound the Lipschitz constant of $h(\boldsymbol{u}) = f(\boldsymbol{x}+\delta\boldsymbol{u}) - f(\boldsymbol{y}+\delta\boldsymbol{u})$. Although this function is indeed $2L\delta$-Lipschitz, using this would bound the variance at $O(dL^2)$, which doesn't match our target of $O(dL^2/T^2)$. On the other hand, directly using the Lipschitzness of f bounds $\mathbb{E}\|\frac{d}{\delta}(f(\boldsymbol{x}+\delta\boldsymbol{u}, z_i) - f(\boldsymbol{y}+\delta\boldsymbol{u}, z_i))\boldsymbol{u}\|^2 \leq (\frac{dL}{\delta}\|\boldsymbol{x}-\boldsymbol{y}\|)^2 = O(d^2L^2/T^2)$. Consequently, we need to sample d i.i.d. uniform vectors to reduce the dimension dependence of the variance by a factor of d in order to match the optimal rate. While a more efficient analysis that avoids sampling d uniform vectors might exist, we are currently unaware of it.

4 Our Algorithm and Results

4.1 Private Online-to-Non-Convex Conversion

Our algorithm builds on the Online-to-Nonconvex Conversion (O2NC) by (Cutkosky et al., 2023), which is a general framework that converts any OCO algorithm with $O(\sqrt{T})$ regret into a nonsmooth nonconvex optimization algorithm that finds a (δ, ϵ)-stationary point in $O(\delta^{-1}\epsilon^{-3})$ iterations. The pseudo-code of this framework is presented in Algorithm 3. Specifically, our algorithm chooses projected Online Subgradient Descent (OSD) as the OCO algorithm, which updates Δ_t^k in line 3 of Algorithm 3 following the explicit update rule: $\Delta_1^k \leftarrow 0$ and $\Delta_t^k \leftarrow \Pi_D(\Delta_{t-1}^k - \eta\tilde{\boldsymbol{g}}_{t-1}^k)$ for $t \geq 2$. Here $\Pi_D(\boldsymbol{x}) = \arg\min_{\|\boldsymbol{y}\|\leq D}\|\boldsymbol{x}-\boldsymbol{y}\|$ denotes the projection operator and η is the stepsize tuned by OSD. Regarding notations in Algorithm 3, there are two round indices t and k, and we use the subscript t and superscript k, namely $\boldsymbol{x}_t^k$, to denote a variable $\boldsymbol{x}$ in outer loop k and inner loop t.

The selection of O2NC as our base non-private algorithm is deliberate and influenced by the literature of non-private zeroth-order algorithms in nonsmooth nonconvex optimization. Intuitively, in the realm of privacy, a variance-reduction algorithm is more appealing due to its inherently low sensitivity. However, a straightforward application of variance-reduced SGD paired with zeroth-order estimators only achieves a sub-optimal dimension dependence of $O(d^{3/2}\delta^{-1}\epsilon^{-3})$, as shown in (Chen et al., 2023). A recent advancement by Kornowski & Shamir (2023) has improved the rate to $O(d\delta^{-1}\epsilon^{-3})$. Central to their achievement is the observation that finding a $(2\delta, \epsilon)$-stationary point

Algorithm 3 Online-to-Nonconvex Conversion

Input: OCO algorithm $\mathcal{A}$ with domain D, gradient oracle $\mathcal{O}$, initial state $\boldsymbol{x}_1$.
1: **for** $k = 1, 2, \ldots, K$ **do**
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Receive Δ_t^k from $\mathcal{A}$. (For example, $\Delta_1^k \leftarrow 0$ and $\Delta_t^k \leftarrow \Pi_D(\Delta_{t-1}^k - \eta \tilde{\boldsymbol{g}}_{t-1}^k), t \geq 2$.)
4: Update $\boldsymbol{x}_{t+1}^k \leftarrow \boldsymbol{x}_t^k + \Delta_t^k$ and $\boldsymbol{w}_t^k \leftarrow \boldsymbol{x}_t^k + s_t^k \Delta_t^k$, where $s_t^k \sim \mathrm{Uniform}([0, 1])$.
5: Query gradient estimator $\tilde{\boldsymbol{g}}_t^k$ from $\mathcal{O}$.
6: Send $\ell_t^k(\cdot) = \langle \tilde{\boldsymbol{g}}_t^k, \cdot \rangle$ to $\mathcal{A}$ and $\mathcal{A}$ updates Δ_{t+1}^k.
7: **end for**
8: Compute $\overline{\boldsymbol{w}}^k \leftarrow \frac{1}{T} \sum_{t=1}^T \boldsymbol{w}_t^k$.
9: Reset $\boldsymbol{x}_1^{k+1} \leftarrow \boldsymbol{x}_{T+1}^k$ and restart $\mathcal{A}$.
10: **end for**
11: Output $\overline{\boldsymbol{w}} \sim \mathrm{Uniform}(\{\overline{\boldsymbol{w}}^1, \ldots, \overline{\boldsymbol{w}}^K\})$.

Algorithm 4 Private variance-reduced gradient oracle $\mathcal{O}$

Input: dataset $\mathcal{Z}$, constants B_1, B_2
Initialize: Partition $\mathcal{Z}$ into KT subsets: Z_1^k of size B_1 for $k \in [K]$ and Z_t^k of size B_2 for $t = [2, K], k \in [K]$. The total data size is $|\mathcal{Z}| = M = K(B_1 + B_2(T-1))$.
Upon receiving round index k, t and parameters $\boldsymbol{w}_t^k$:
if $t = 1$ **then**
Query $\boldsymbol{g}_1^k \leftarrow \textsc{Grad}_{f,\delta}(\boldsymbol{w}_1^k, Z_1^k)$.
else
Query $\boldsymbol{d}_t^k \leftarrow \textsc{Diff}_{f,\delta}(\boldsymbol{w}_t^k, \boldsymbol{w}_{t-1}^k, Z_t^k)$ and compute $\boldsymbol{g}_t^k \leftarrow \boldsymbol{g}_{t-1}^k + \boldsymbol{d}_t^k$.
end if
Return $\tilde{\boldsymbol{g}}_t^k \leftarrow \tilde{\boldsymbol{g}}_t^k + \textsc{Tree}(t)$.

of a nonsmooth objective F is equivalent to finding a (δ, ϵ)-stationary point of its uniform smoothing $\hat{F}_\delta$. Previous zeroth-order algorithms aim to find an ϵ-stationary point of $\hat{F}_\delta$ via smooth optimization algorithms such as variance-reduced SGD, resulting in an additional smoothness cost of order $O(\sqrt{d})$. In contrast, the algorithm proposed by Kornowski & Shamir (2023) leverages O2NC to find a (δ, ϵ)-stationary point, achieving a linear dimension dependence.

Informed by the insights of (Kornowski & Shamir, 2023), we adopt O2NC as our foundational non-private algorithm. Specifically, we adapt Algorithm 3 for privacy by substituting the gradient oracle $\mathcal{O}$ in line 5 with a more carefully designed private oracle. A main challenge in this design is to minimize *sensitivity*, which is required to ensure privacy. In the algorithm proposed by Kornowski & Shamir (2023), the gradient oracle is simply the zeroth-order gradient estimator, $\tilde{\boldsymbol{g}}_t^k = \frac{d}{2\delta}(f(\boldsymbol{w}_t^k + \delta \boldsymbol{u}, z_t^k) - f(\boldsymbol{w}_t^k + \delta \boldsymbol{u}, z_t^k))\boldsymbol{u}$, whose sensitivity is $O(dL)$. A straightforward method to ensure privacy for this algorithm is by directly adding Gaussian noise with variance $O(d^2 L^2/\rho^2)$. While this might seem intuitive, it leads to significantly worse data complexity. Specifically, the resulting rate is $O(d^{3/2}\rho^{-1}\delta^{-1}\epsilon^{-3})$, which is worse by a factor of ϵ^{-1} when compared to our proposed algorithm - and importantly *does not* admit any value of ρ for which we obtain the non-private convergence rate. For interested readers, we include the detailed analysis of this naive approach in Appendix E.

In contrast, we designed a more refined private gradient oracle in Algorithm 4. According to Lemma 3.1 and 3.2, the sensitivity of $\boldsymbol{g}_1^k$ is $O(dL/B_1)$, while that of $\boldsymbol{d}_t^k$ is $O(dL\|\boldsymbol{w}_t^k - \boldsymbol{w}_{t-1}^k\|/B_2\delta)$. Setting $B_1 = T, B_2 = 1$, and using Remark 4.1 with $D = \delta/T$, both $\boldsymbol{g}_1^l$ and $\boldsymbol{d}_t^k$ have sensitivity $O(dL/T)$. With these settings, we can apply the tree mechanism, a useful technique to release sums of private algorithms, ensuring that the variance-reduced gradient $\tilde{\boldsymbol{g}}_t^k$ achieves $(\alpha, \alpha\rho^2/2)$-RDP. Specifically, the tree mechanism adds noise of order $\tilde{O}(d^2 L^2/\rho^2 T^2)$ to each $\tilde{\boldsymbol{g}}_t^k$. Compared to the aforementioned naive approach, our refined approach reduces noise by a factor of T^2, thus resulting in the desired data complexity.

Remark 4.1. Note that $\boldsymbol{w}_{t+1}^k = \boldsymbol{x}_{t+1}^k + s_{t+1}^k \Delta_{t+1}^k = \boldsymbol{w}_t^k + (1 - s_t^k)\Delta_t^k + s_{t+1}^k \Delta_{t+1}^k$. Therefore, if the OCO algorithm has domain bounded by D (i.e., $\|\Delta_t^k\| \leq D$ for all t, k), then $\|\boldsymbol{w}_{t+1}^k - \boldsymbol{w}_t^k\| \leq 2D$. In general, for all $t, t' \in [T]$, $\|\boldsymbol{w}_t^k - \boldsymbol{w}_{t'}^k\| \leq 2|t - t'|D$.

Algorithm 5 Tree Mechanism

Input: Index $t \in [T]$, global variables $\sigma_1, \ldots, \sigma_T$.
If $t = 1$, set NOISE $\leftarrow \{\}$.
Sample $\xi_t \sim N(0, \sigma_t^2 I)$ and store ξ_t in NOISE.
Return $\sum_{(\cdot,i)\in \text{NODE}(t)} \xi_i$.

function NODE(t)
  Initialize: Set $k \leftarrow 0$ and $S \leftarrow \{\}$.
  for $i = 0, \ldots, \lceil \log_2(t) \rceil$ while $k < t$ **do**
    Set $k' \leftarrow k + 2^{\lceil \log_2(t) \rceil - i}$. If $k' \leq t$, store $S \leftarrow S \cup \{(k+1), k'\}$ and update $k \leftarrow k'$.
  end for
  Return S.
end function

4.2 PRIVACY GUARANTEE

To ensure the privacy of the Online-to-Nonconvex conversion, we use a variant of the tree mechanism, as defined in Algorithm 5, to privately aggregate the sum of $\boldsymbol{g}_1^k$ and $\boldsymbol{d}_t^k$. The privacy guarantee of the tree mechanism is presented in Theorem 4.3, and the proof is presented in Append C.1. Intuitively, the tree mechanism says if each of the algorithms $\mathcal{M}_1, \ldots, \mathcal{M}_n$ requires σ^2 noise for privacy, then the sum $\sum_{i=1}^n \mathcal{M}_i$ only needs $O(\ln(n)\sigma^2)$ noise for the same level of privacy. With Theorem 4.3, we determine the minimal noise required to ensure privacy for Algorithm 3 in Corollary 4.4, and we evaluate the total noise added by the tree mechanism in Corollary 4.5.

Remark 4.2. To better understand the NODE function in Algorithm 5, consider a binary tree of depth $\lceil \log_2(t) \rceil$. NODE(t) returns the largest node n_i in each layer i, if exists, such that $n_j < n_i \leq t$ starting from the top level. As an example, NODE$(7) = \{(1,4),(5,6),(7,7)\}$ and NODE$(8) = \{(1,8)\}$. In particular, note that $|\text{NODE}(t)| \leq \lceil \log_2(t) \rceil \leq 2\ln(t)$.

Next, we present the main privacy theorem for the tree mechanism as presented in Algorithm 5.

Theorem 4.3. *Let $\mathcal{X}$ be state space and $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(n)}$ be dataset spaces, and denote $\mathcal{Z}^{(1:i)} = \mathcal{Z}^{(1)} \times \cdots \times \mathcal{Z}^{(i)}$. Let $\mathcal{M}^{(i)} : \mathcal{X}^{i-1} \times \mathcal{Z}^{(i)} \to \mathcal{X}$ be a sequence of algorithms for $i \in [n]$, and let $\mathcal{A} : \mathcal{Z}^{(1:n)} \to \mathcal{X}^n$ be the algorithm that, given a dataset $Z_{1:n} \in \mathcal{Z}^{(1:n)}$, sequentially computes $\boldsymbol{x}_i = \sum_{j=1}^i \mathcal{M}^{(j)}(\boldsymbol{x}_{1:j-1}, Z_i) + \text{TREE}(i)$ for $i \in [n]$ and then outputs $\boldsymbol{x}_{1:n}$. Suppose for all $i \in [n]$ and neighboring $Z_{1:n}, Z'_{1:n} \in \mathcal{Z}^{(1:n)}$, $\|\mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z_i) - \mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z'_i)\| \leq s_i$ for all auxiliary inputs $\boldsymbol{x}_{1:i-1} \in \mathcal{X}^{i-1}$. Then for all $\alpha > 1$, $\mathcal{A}$ is $(\alpha, \alpha\rho^2/2)$-RDP where*

$$\rho \leq \sqrt{2\ln n} \cdot \max_{b\in[n], i\leq b} \frac{s_i}{\sigma_b}.$$

In our case, $\mathcal{X}$ is the collection of weights $\boldsymbol{x}$ for $f(\boldsymbol{x}, z)$, $\mathcal{Z}^{(1)}$ is the collection of all possible batches Z_1^k of size B_1 defined in Algorithm 3 and $\mathcal{Z}^{(t)}$ for $t \geq 2$ is the collection of all possible batches Z_t^k of size B_2. Moreover, $\mathcal{M}^{(1)}$ corresponds to GRAD that computes $\boldsymbol{g}_1^k$ and $\mathcal{M}^{(t)}$ corresponds to DIFF that computes $\boldsymbol{d}_t^k$. Upon substituting these specific definition into Theorem 4.3, we have the following privacy guarantees.

Corollary 4.4. *Suppose $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$ and the domain of $\mathcal{A}$ is bounded by $D = \delta/T$. Let B_1, B_2 satisfies $B_1 \geq TB_2/2$. Then for any $\alpha > 1, \rho > 0$, Algorithm 3 is $(\alpha, \alpha\rho^2/2)$-RDP if we set the noises $\sigma_{1:T}$ in the tree mechanism as*

$$\sigma_t = \sigma := \frac{\sqrt{2\ln T}4dL}{B_2 T\rho}.$$

Corollary 4.5. *Following the assumptions and definitions in Corollary 4.4, for all $t \in [T]$,*

$$\mathbb{E}\|\text{TREE}(t)\|^2 \leq 2\ln(t)d\sigma^2 \leq \left(\frac{8\ln(T)d^{3/2}L}{B_2 T\rho}\right)^2.$$

4.3 CONVERGENCE ANALYSIS

To start the convergence analysis, we revisit the convergence result of the non-private O2NC algorithm as presented in Lemma 4.6 whose proof is included in Appendix D for completeness. Building upon the observation in Corollary 2.3, our objective shifts to finding a (δ, ϵ)-stationary point of the uniform smoothing $\hat{F}_\delta$, as opposed to directly optimizing F. This approach leads to the formulation of Corollary 4.7, whose proof is also deferred to the appendix.

Lemma 4.6. *For any function $F : \mathbb{R}^d \to \mathbb{R}$ that is differentiable and $F(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} F(\boldsymbol{x}) \le F^*$, if the domain of the OCO algorithm $\mathcal{A}$ is bounded by $D = \delta/T$, then*

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_\delta \le \frac{F^*}{DKT} + \sum_{k=1}^{K} \frac{\mathbb{E}\mathrm{Reg}_T(\boldsymbol{u}^k)}{DKT} + \sum_{k=1}^{K}\sum_{t=1}^{T} \frac{\mathbb{E}\langle \nabla F(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \boldsymbol{u}^k\rangle}{DKT}.$$

where $\boldsymbol{u}^k = -D\frac{\sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k)}{\|\sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k)\|}$.

Corollary 4.7. *Suppose $F : \mathbb{R}^d \to \mathbb{R}$ is differentiable, L-Lipschitz, and $F(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} F(\boldsymbol{x}) \le F^*$, and suppose the domain of $\mathcal{A}$ is bounded by $D = \delta/T$. Then*

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \le \frac{F^* + 2L\delta}{DKT} + \sum_{k=1}^{K} \frac{\mathbb{E}\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)}{DKT} + \sum_{k=1}^{K}\sum_{t=1}^{T} \frac{\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k\rangle}{DKT}.$$

where $\hat{\boldsymbol{u}}^k = -D\frac{\sum_{t=1}^T \nabla \hat{F}_\delta(\boldsymbol{w}_t^k)}{\|\sum_{t=1}^T \nabla \hat{F}_\delta(\boldsymbol{w}_t^k)\|}$.

Next, we introduce a key result in our convergence analysis. At its core, Lemma 4.8 suggests that the variance of the non-private gradient oracle, defined as $\boldsymbol{g}_t^k$ in Algorithm 4, is approximately $O(dL^2/T)$. This result leads to the non-private term of $O(d\delta^{-1}\epsilon^{-3})$ in the data complexity, matching the optimal rate in non-private contexts. Together with Corollary 4.5, this lemma implies the main result as stated in Theorem 4.9.

Lemma 4.8. *Suppose $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$ and the domain of $\mathcal{A}$ is bounded by $D = \delta/T$, then the variance of the gradient oracle $\mathcal{O}$ (Algorithm 4) is bounded by*

$$\mathbb{E}\|\nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \boldsymbol{g}_t^k\|^2 \le \frac{16dL^2}{B_1} + \frac{64dL^2}{B_2 T}.$$

Proof. First, we recall a martingale concentration bound. Let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be a sequence of random vectors and let $\mathcal{F}_i = \sigma(\boldsymbol{v}_{1:i})$ be the σ-algebra generated by $\boldsymbol{v}_{1:i}$. If $\mathbb{E}[\boldsymbol{v}_i|\mathcal{F}_{i-1}] = 0$, then for all cross terms $i \le j$, $v_i \in \mathcal{F}_{j-1}$ and thus $\mathbb{E}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle = \mathbb{E}_{\mathcal{F}_{j-1}}\langle \boldsymbol{v}_i, \mathbb{E}[\boldsymbol{v}_j|\mathcal{F}_{j-1}]\rangle = 0$. Consequently,

$$\mathbb{E}\|\textstyle\sum_{i=1}^n \boldsymbol{v}_i\|^2 = \sum_{i=1}^n \mathbb{E}\|\boldsymbol{v}_i\|^2.$$

In our case, note that $\nabla \hat{F}(\boldsymbol{w}_t^k) = \nabla \hat{F}(\boldsymbol{w}_1^k) + \sum_{i=2}^t \nabla \hat{F}(\boldsymbol{w}_t^k) - \nabla \hat{F}(\boldsymbol{w}_{t-1}^k)$ and recall that $\boldsymbol{g}_t^k = \boldsymbol{g}_1^k + \sum_{i=2}^t \boldsymbol{d}_i$. Therefore, by the concentration inequality and Lemma 3.1 and 3.2,

$$\begin{aligned}\mathbb{E}\|\nabla \hat{F}(\boldsymbol{w}_t^k) - \boldsymbol{g}_t^k\|^2 &\le \mathbb{E}\|\nabla \hat{F}(\boldsymbol{w}_1^k) - \boldsymbol{g}_1^k\|^2 + \sum_{i=2}^{t} \mathbb{E}\|\nabla \hat{F}(\boldsymbol{w}_i^k) - \nabla \hat{F}(\boldsymbol{w}_{i-1}^k) - \boldsymbol{d}_i^k\|^2 \\ &\le \frac{16dL^2}{B_1} + \sum_{i=2}^{t} \frac{16dL^2}{B_2\delta^2}\mathbb{E}\|\boldsymbol{w}_i^k - \boldsymbol{w}_{i-1}^k\|^2.\end{aligned}$$

We conclude the proof by Remark 4.1 that $\|\boldsymbol{w}_i^k - \boldsymbol{w}_{i-1}^k\| \le 2D \le \frac{2\delta}{T}$. □

Theorem 4.9. *Suppose $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$, $F(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} F(\boldsymbol{x}) \le F^*$. Then for any $\delta, \epsilon > 0$ and $\alpha > 1, \rho > 0$, there exists*

$$M = \tilde{O}\left(d\left(\frac{F^* L^2}{\delta\epsilon^3} + \frac{L^3}{\epsilon^3}\right) + d^{3/2}\left(\frac{F^* L}{\rho\delta\epsilon^2} + \frac{L^2}{\rho\epsilon^2}\right)\right)$$

such that upon running Algorithm 3 with projected OSD as the OCO algorithm, using Algorithm 4 as the private oracle, and setting σ_t as defined in Corollary 4.4 and $B_1 = T + 1, B_2 = 1, D = \frac{\delta}{T}, T = \min\{(\frac{\sqrt{d}L\delta M}{F^* + L\delta})^{2/3}, (\frac{d^{3/2}L\delta M}{(F^* + L\delta)\rho})^{1/2}\}, K = \frac{M}{2T}$, the algorithm outputs an $(\alpha, \alpha\rho^2/2)$-RDP (δ, ϵ)-stationary point with M data points.

Proof. Since $B_1 = T+1, B_2 = 1$ satisfy $B_1 \geq TB_2/2$, Corollary 4.4 implies the privacy guarantee. For the convergence analysis, Corollary 4.7 states that

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \leq \frac{F^* + 2L\delta}{DKT} + \sum_{k=1}^{K} \frac{\mathbb{E}\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)}{DKT} + \sum_{k=1}^{K}\sum_{t=1}^{T} \frac{\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k\rangle}{DKT}.$$

Recall that given a sequence of stochastic vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_T$, the regret of projected OSD is bounded by $\mathbb{E}[\mathrm{Reg}_T(\boldsymbol{u})] \leq D\sqrt{\sum_{t=1}^T \mathbb{E}\|\boldsymbol{v}_t\|^2}$ (Orabona, 2019). In our case, $\boldsymbol{v}_t = \tilde{\boldsymbol{g}}_t^k$. We can bound

$$\begin{aligned}\mathbb{E}\|\tilde{\boldsymbol{g}}_t^k\|^2 &\leq 3\mathbb{E}\|\boldsymbol{g}_t^k - \nabla \hat{F}_\delta(\boldsymbol{w}_t^k)\|^2 + 3\mathbb{E}\|\nabla \hat{F}_\delta(\boldsymbol{w}_t^k)\|^2 + 3\mathbb{E}\|\textsc{Tree}(t)\|^2 \\ &\leq 3 \cdot \frac{80dL^2}{B_2T} + 3L^2 + 3 \cdot \left(\frac{8\ln(T)d^{3/2}L}{B_2T\rho}\right)^2,\end{aligned}$$

where the first term is bounded by Lemma 4.8 with $B_1 \geq TB_2/2$, the second by Lipschitzness of $\hat{F}_\delta$ (Lemma 2.2 (i)), and the third by Corollary 4.5. Consequently, we bound the regret by

$$\mathbb{E}[\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)] \lesssim DL\sqrt{T}\left(\frac{\sqrt{d}}{\sqrt{B_2T}} + 1 + \frac{d^{3/2}}{B_2T\rho}\right). \tag{1}$$

Next, note that $\|\Delta_t^k - \hat{\boldsymbol{u}}^k\| \leq 2D$. Following the same previous bounds, we have

$$\begin{aligned}\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k\rangle &\leq 2D\mathbb{E}[\|\nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \boldsymbol{g}_t^k\| + \|\textsc{Tree}(t)\|] \\ &\lesssim D\left(\frac{\sqrt{d}L}{\sqrt{B_2T}} + \frac{d^{3/2}L}{B_2T\rho}\right).\end{aligned} \tag{2}$$

Upon substituting equation 1 and equation 2 into Corollary 4.7, we have

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \lesssim \frac{F^* + L\delta}{DKT} + \frac{L}{\sqrt{T}}\left(\frac{\sqrt{d}}{\sqrt{B_2T}} + 1 + \frac{d^{3/2}}{B_2T\rho}\right) + \left(\frac{\sqrt{d}L}{\sqrt{B_2T}} + \frac{d^{3/2}L}{B_2T\rho}\right)$$

Upon setting $B_1 = T+1, B_2 = 1, D = \frac{\delta}{T}$, the data size is $M = K(B_1 + B_2(T-1)) = 2KT$ and

$$\lesssim \frac{(F^* + L\delta)T}{\delta M} + \frac{\sqrt{d}L}{\sqrt{T}} + \frac{d^{3/2}L}{T\rho}$$

Upon setting $T = \min\{(\frac{\sqrt{d}L\delta M}{F^*+L\delta})^{2/3}, (\frac{d^{3/2}L\delta M}{(F^*+L\delta)\rho})^{1/2}\}$, we achieve the desired bound

$$\lesssim \left(\frac{F^*dL^2}{\delta M}\right)^{1/3} + \left(\frac{dL^3}{M}\right)^{1/3} + \left(\frac{F^*d^{3/2}L}{\rho\delta M}\right)^{1/2} + \left(\frac{d^{3/2}L^2}{\rho M}\right)^{1/2}.$$

Equivalently, the right hand side is bounded by ϵ if $M = \max\{\frac{F^*dL^2}{\delta\epsilon^3}, \frac{dL^3}{\epsilon^3}, \frac{F^*d^{3/2}L}{\rho\delta\epsilon^2}, \frac{d^{3/2}L^2}{\rho\epsilon^2}\}$. □

5 Conclusion

This paper presents a novel zeroth-order algorithm for private nonsmooth nonconvex optimization. We prove that, given a dataset of size M, our algorithm finds a (δ, ϵ)-stationary point while achieving $(\alpha, \alpha\rho^2/2)$-RDP privacy guarantee so long as $M = \tilde{\Omega}\left(d(\frac{F^*L^2}{\delta\epsilon^3} + \frac{L^3}{\epsilon^3}) + d^{3/2}(\frac{F^*L}{\rho\delta\epsilon^2} + \frac{L^2}{\rho\epsilon^2})\right)$. Notably, this is the first algorithm for the private nonsmooth nonconvex optimization problem.

For future research, there are several intriguing directions. First, we are interested in exploring methods that could eliminate the need to sample d uniform vectors per data point, a limitation we delve into in Section 3. Additionally, the design of efficient first-order private algorithms for nonsmooth nonconvex optimization stands out as an area of potential exploration. Finally, the optimality of the current rate is still uncertain, and finding the lower bounds remains an open problem.

REFERENCES

Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *arXiv preprint arXiv:1912.02365*, 2019.

Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Ayush Sekhari, and Karthik Sridharan. Second-order information in non-convex stochastic optimization: Power and limitations. In *Conference on Learning Theory*, pp. 242–299, 2020.

Raman Arora, Raef Bassily, Tomás González, Cristóbal Guzmán, Michael Menart, and Enayat Ullah. Faster rates of convergence to stationary points in differentially private optimization. 2023.

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pp. 464–473. IEEE, 2014.

Raef Bassily, Vitaly Feldman, Kunal Talwar, and Abhradeep Thakurta. Private stochastic convex optimization with optimal rates. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 11282–11291, 2019.

Raef Bassily, Cristóbal Guzmán, and Michael Menart. Differentially private stochastic optimization: New results in convex and non-convex settings. *Advances in Neural Information Processing Systems*, 34, 2021a.

Raef Bassily, Cristóbal Guzmán, and Anupama Nandi. Non-euclidean differentially private stochastic convex optimization. In *Conference on Learning Theory*, pp. 474–499. PMLR, 2021b.

Dimitri P Bertsekas. Stochastic optimization problems with nondifferentiable cost functionals. *Journal of Optimization Theory and Applications*, 12(2):218–231, 1973.

Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points i. *Mathematical Programming*, pp. 1–50, 2019.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

T-H Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Transactions on Information and System Security (TISSEC)*, 14(3):1–24, 2011.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

Lesi Chen, Jing Xu, and Luo Luo. Faster gradient-free algorithms for nonsmooth nonconvex stochastic optimization, 2023.

Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. In *Advances in Neural Information Processing Systems*, pp. 15210–15219, 2019.

Ashok Cutkosky, Harsh Mehta, and Francesco Orabona. Optimal stochastic non-smooth non-convex optimization through online-to-non-convex conversion. In *International Conference on Machine Learning (ICML)*, 2023.

John C Duchi, Peter L Bartlett, and Martin J Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

John C. Duchi, Michael I. Jordan, Martin J. Wainwright, and Andre Wibisono. Optimal rates for zero-order convex optimization: The power of two function evaluations. *IEEE Transactions on Information Theory*, 61(5):2788–2806, 2015. doi: 10.1109/TIT.2015.2409256.

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

Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pp. 715–724, 2010.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pp. 689–699, 2018.

Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private stochastic convex optimization: optimal rates in linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 439–449, 2020.

Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 385–394, 2005.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

A. A. Goldstein. Optimization of lipschitz continuous functions. *Math. Program.*, 13(1):14–22, dec 1977. ISSN 0025-5610. doi: 10.1007/BF01584320. URL `https://doi.org/10.1007/BF01584320`.

Elad Hazan. Introduction to online convex optimization. *arXiv preprint arXiv:1909.05207*, 2019.

Prateek Jain and Abhradeep Guha Thakurta. (near) dimension independent risk bounds for differentially private learning. In *International Conference on Machine Learning*, pp. 476–484. PMLR, 2014.

Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. In *Conference on Learning Theory*, pp. 24–1. JMLR Workshop and Conference Proceedings, 2012.

Daniel Kifer, Adam Smith, and Abhradeep Thakurta. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, pp. 25–1. JMLR Workshop and Conference Proceedings, 2012.

Guy Kornowski and Ohad Shamir. Oracle complexity in nonsmooth nonconvex optimization. *Journal of Machine Learning Research*, 23(314):1–44, 2022.

Guy Kornowski and Ohad Shamir. An algorithm with optimal dimension-dependence for zero-order nonsmooth nonconvex stochastic optimization, 2023.

Tianyi Lin, Zeyu Zheng, and Michael I. Jordan. Gradient-free methods for deterministic and stochastic nonsmooth nonconvex optimization, 2022.

Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search of static linear policies is competitive for reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 263–275. IEEE, 2017.

Barry L. Nelson. Optimization via simulation over discrete decision variables. In *Risk and Optimization in an Uncertain World*, pp. 193–207. INFORMS, September 2010. doi: 10.1287/educ.1100.0069. URL `https://doi.org/10.1287/educ.1100.0069`.

Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

Ohad Shamir. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *Journal of Machine Learning Research*, 18(52):1–11, 2017.

Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Private empirical risk minimization beyond the worst case: The effect of the constraint set geometry. *arXiv preprint arXiv:1411.5417*, 2014.

Lai Tian, Kaiwen Zhou, and Anthony Man-Cho So. On the finite-time complexity and practical computation of approximate stationarity concepts of lipschitz functions. In *International Conference on Machine Learning*, pp. 21360–21379. PMLR, 2022.

Hoang Tran and Ashok Cutkosky. Momentum aggregation for private non-convex ERM. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=x56v-UN7BjD.

Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2019. doi: 10.1017/9781108627771.

Di Wang, Minwei Ye, and Jinhui Xu. Differentially private empirical risk minimization revisited: faster and more general. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 2719–2728, 2017.

Di Wang, Changyou Chen, and Jinhui Xu. Differentially private empirical risk minimization with non-convex loss functions. In *International Conference on Machine Learning*, pp. 6526–6535. PMLR, 2019a.

Lingxiao Wang, Bargav Jayaraman, David Evans, and Quanquan Gu. Efficient privacy-preserving stochastic nonconvex optimization. *arXiv preprint arXiv:1910.13659*, 2019b.

Farzad Yousefian, Angelia Nedić, and Uday V. Shanbhag. On stochastic gradient and subgradient methods with adaptive steplength sequences. *Automatica*, 48(1):56–67, 2012. ISSN 0005-1098.

Jingzhao Zhang, Hongzhou Lin, Stefanie Jegelka, Ali Jadbabaie, and Suvrit Sra. Complexity of finding stationary points of nonsmooth nonconvex functions. 2020.

Qinzi Zhang, Hoang Tran, and Ashok Cutkosky. Differentially private online-to-batch for smooth losses. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yingxue Zhou, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Arindam Banerjee. Private stochastic non-convex optimization: Adaptive algorithms and tighter generalization bounds. *arXiv preprint arXiv:2006.13501*, 2020.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 928–936, 2003.

A PROOFS IN SECTION 2

A.1 PROOF OF LEMMA 2.2

Lemma 2.2. *Suppose $h : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz. Then (i) $\hat{h}_\delta$ is L-Lipschitz; (ii) $\|\hat{h}_\delta(\boldsymbol{x}) - h(\boldsymbol{x})\| \le L\delta$; (iii) $\hat{h}_\delta$ is differentiable and $\frac{\sqrt{d}L}{\delta}$-smooth; (iv)*

$$\nabla \hat{h}_\delta(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{u}\sim\mathcal{U}_\mathbb{S}}[\tfrac{d}{\delta} h(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u}] = \mathbb{E}_{\boldsymbol{u}\sim\mathcal{U}_\mathbb{S}}[\tfrac{d}{2\delta}(h(\boldsymbol{x}+\delta\boldsymbol{u}) - h(\boldsymbol{x}-\delta\boldsymbol{u}))\boldsymbol{u}];$$

Proof. For simplicity, we drop the subscript δ of $\hat{h}_\delta$. By Jensen's inequality and Lipschitzness,

$$\|\hat{h}(\boldsymbol{x}) - \hat{h}(\boldsymbol{y})\| \le \mathbb{E}_{\boldsymbol{v}}\|h(\boldsymbol{x}+\delta\boldsymbol{v}) - h(\boldsymbol{y}+\delta\boldsymbol{v})\| \le L\|\boldsymbol{x}-\boldsymbol{y}\|, \tag{i}$$

$$\|\hat{h}(\boldsymbol{x}) - h(\boldsymbol{x})\| \le \mathbb{E}_{\boldsymbol{v}}\|h(\boldsymbol{x}+\delta\boldsymbol{v}) - h(\boldsymbol{x})\| \le L\|\delta\boldsymbol{v}\| \le L\delta. \tag{ii}$$

Next, since h is Lipschitz, h is almost every differentiable by Rademacher's theorem and the uniform smoothing $\hat{h}$ is also almost everywhere differentiable by (Bertsekas, 1973, Proposition 2.4). Moreover, it holds that $\nabla\hat{h}(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{v}}[\nabla h(\boldsymbol{x}+\delta\boldsymbol{v})]$. Denote $\triangle$ as the symmetric difference, i.e., $A \triangle B = (A \setminus B) \cup (B \setminus A)$, and let $S = B(\boldsymbol{x},\delta)\triangle B(\boldsymbol{y},\delta)$. Then

$$\begin{aligned}
\|\nabla\hat{h}(\boldsymbol{x}) - \nabla\hat{h}(\boldsymbol{y})\| &= \|\mathbb{E}_{\boldsymbol{v}\sim\mathcal{U}_\mathbb{B}}[\nabla h(\boldsymbol{x}+\delta\boldsymbol{v}) - \nabla h(\boldsymbol{y}+\delta\boldsymbol{v})]\| \\
&= \left\| \int_{B(\boldsymbol{x},\delta)} \frac{\nabla h(\boldsymbol{v})}{\mathrm{vol}(B(\boldsymbol{x},\delta))}\, d\boldsymbol{v} - \int_{B(\boldsymbol{y},\delta)} \frac{\nabla h(\boldsymbol{v})}{\mathrm{vol}(B(\boldsymbol{y},\delta))}\, d\boldsymbol{v} \right\| \\
&= \left\| \int_S \frac{\nabla h(\boldsymbol{v})}{\mathrm{vol}(\delta\mathbb{B})}\, d\boldsymbol{v} \right\| \le \int_S \frac{\|\nabla h(\boldsymbol{v})\|}{\mathrm{vol}(\delta\mathbb{B})}\, d\boldsymbol{v} \le \frac{\mathrm{vol}(S)L}{\mathrm{vol}(\delta\mathbb{B})} \le \frac{\sqrt{d}L}{\delta}\|\boldsymbol{x}-\boldsymbol{y}\|,
\end{aligned}$$

where the second last inequality follows from $\|\nabla h(\boldsymbol{v})\| \le L$, and the last follows from Lemma A.1.

Finally, by definition of expectation,

$$\mathbb{E}_{\boldsymbol{v}\sim\mathcal{U}_\mathbb{B}}[\nabla h(\boldsymbol{x}+\delta\boldsymbol{v})] = \frac{\int_\mathbb{B} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}+\delta\boldsymbol{v})\, d\boldsymbol{v}}{\mathrm{vol}(\mathbb{B})},$$

$$\mathbb{E}_{\boldsymbol{u}\sim\mathcal{U}_\mathbb{S}}[h(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u}] = \frac{\int_\mathbb{S} f(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u}\, d\boldsymbol{u}}{\mathrm{vol}(\mathbb{S})}.$$

By Stokes' theorem,

$$\int_\mathbb{B} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}+\delta\boldsymbol{v})\, d\boldsymbol{v} = \int_\mathbb{S} f(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u}\, d\boldsymbol{u}$$

We then establish the first equality in (iv) by $\mathrm{vol}(\mathbb{B})/\mathrm{vol}(\mathbb{S}) = \delta/d$. Next, let p be a Rademacher random variable and $\boldsymbol{u}' = \boldsymbol{u}/p$. Observe that $\boldsymbol{u}' \sim \mathcal{U}_\mathbb{S}$ as well. Therefore,

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{u}\sim\mathcal{U}_\mathbb{S}}[\tfrac{d}{\delta} h(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u}] &= \mathbb{E}_{p,\boldsymbol{u}'}[\tfrac{d}{\delta} h(\boldsymbol{x}+\delta p\boldsymbol{u}')p\boldsymbol{u}'] \\
&= \tfrac{1}{2}\mathbb{E}_{\boldsymbol{u}'}[\tfrac{d}{\delta} h(\boldsymbol{x}+\delta\boldsymbol{u})\boldsymbol{u} + \tfrac{d}{\delta} h(\boldsymbol{x}-\delta\boldsymbol{u})(-\boldsymbol{u})].
\end{aligned}$$

This proves the second equality in (iv). □

Lemma A.1. *Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ and $S = B(\boldsymbol{x},\delta)\triangle B(\boldsymbol{y},\delta)$, then $\frac{\mathrm{vol}(S)}{\mathrm{vol}(\delta\mathbb{B})} \le \frac{\sqrt{d}}{\delta}\|\boldsymbol{x}-\boldsymbol{y}\|$.*

Proof. Let $D = \|\boldsymbol{x}-\boldsymbol{y}\|$, $V_d(r)$ be the volume of an d-ball in $\mathbb{R}^d$, and $U_d(h,r)$ be the volume of the cap of an d-ball of height h. Observe that for $D \le 2\delta$,

$$\frac{\mathrm{vol}(S)}{\mathrm{vol}(\delta\mathbb{B})} = 2 \cdot \frac{U_d(\delta + \frac{D}{2}, \delta) - U_d(\delta - \frac{D}{2}, \delta)}{V_d(\delta)}.$$

Next, we compute the value of $U_d(h,r)$.

$$U_d(h,r) = \int_0^h V_{d-1}(\sqrt{r^2 - (r-t)^2})\, dt.$$

Recall that the volume of an d-ball is $V_d(r) = \frac{\pi^{d/2}}{\Gamma(d/2+1)} r^n$. Therefore,

$$\begin{aligned}
\frac{\mathrm{vol}(S)}{\mathrm{vol}(\delta\mathbb{B})} &= \frac{\frac{\pi^{(d-1)/2}}{\Gamma((d-1)/2+1)} \int_{\delta-\frac{D}{2}}^{\delta+\frac{D}{2}} (2\delta t - t^2)^{\frac{d-1}{2}}\, dt}{\frac{\pi^{d/2}}{\Gamma(d/2+1)} \delta^d} \\
&= \frac{\Gamma(\frac{d}{2}+1)}{\sqrt{\pi}\Gamma(\frac{d-1}{2}+1)} \int_{\delta-\frac{D}{2}}^{\delta+\frac{D}{2}} \left(\frac{2t}{\delta} - \frac{t^2}{\delta^2}\right)^{\frac{d-1}{2}} \frac{1}{\delta}\, dt \\
&= \frac{\Gamma(\frac{d}{2}+1)}{\sqrt{\pi}\Gamma(\frac{d-1}{2}+1)} \int_{1-\frac{D}{2\delta}}^{1+\frac{D}{2\delta}} (2u - u^2)^{\frac{d-1}{2}}\, du. \qquad (t = \delta u)
\end{aligned}$$

Note that $2u - u^2 \leq 1$ for all $u \in \mathbb{R}$, so the integral is bounded by $\int_{1-\frac{D}{2\delta}}^{1+\frac{D}{2\delta}} 1\, du = \frac{D}{\delta}$. The proof then follows from the fact that $\Gamma(\frac{d}{2}+1)/\Gamma(\frac{d-1}{2}+1) \leq \sqrt{2d}$ (proved in Lemma A.2 for completeness). □

Lemma A.2. *For all $n \in \mathbb{N}$, $\Gamma(\frac{n}{2}+1)/\Gamma(\frac{n-1}{2}+1) \leq \sqrt{2n}$.*

Proof. Let $n!! = n(n-2)(n-4)\cdots = \prod_{k=0}^{\lceil \frac{n}{2} \rceil - 1}(n-2k)$, then by definition of $\Gamma(n+1) = n\gamma(n)$ and $\Gamma(\frac{1}{2}) = \sqrt{\pi}$, we have

$$\Gamma(\tfrac{n}{2}+1) = \begin{cases} \frac{n!!}{2^{n/2}}, & n \text{ is even} \\ \frac{\sqrt{\pi} n!!}{2^{(n+1)/2}}, & n \text{ is odd} \end{cases}$$

Consequently, we have

$$\frac{\Gamma(\frac{n}{2}+1)}{\Gamma(\frac{n-1}{2}+1)} = \begin{cases} \frac{n!!}{\sqrt{\pi}(n-1)!!}, & n \text{ is even} \\ \frac{\sqrt{\pi} n!!}{2(n-1)!!}, & n \text{ is odd} \end{cases} \leq \frac{n!!}{(n-1)!!}.$$

We finish the proof by induction. For $n = 1$, $\frac{1!!}{0!!} = 1 \leq \sqrt{2 \cdot 1}$. For $n = 2$, $\frac{2!!}{1!!} = 2 \leq \sqrt{2 \cdot 2}$. Now assume $\frac{m!!}{(m-1)!!} \leq \sqrt{2m}$ for all $m \leq n$. Then

$$\frac{(n+1)!!}{n!!} = \frac{n+1}{n}\frac{(n-1)!!}{(n-2)!!} \leq \frac{n+1}{n}\sqrt{2(n-1)} = \sqrt{2(n+1)}\frac{\sqrt{(n+1)(n-1)}}{n} \leq \sqrt{2(n+1)}.$$

This concludes the proof by induction. □

A.2 Proof of Corollary 2.3

Corollary 2.3. *Suppose $F : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz. Then for any $\epsilon, \delta > 0$, $\|\nabla \hat{F}_\delta(\boldsymbol{x})\|_\delta \leq \epsilon$ implies that $\|\nabla F(\boldsymbol{x})\|_{2\delta} \leq \epsilon$.*

Proof. Recall that the δ-subdifferential of $\hat{F}_\delta$ is defined as $\partial_\delta \hat{F}_\delta(\boldsymbol{x}) = \mathrm{conv}(\cup_{\boldsymbol{y} \in B(\boldsymbol{x},\delta)} \nabla \hat{F}_\delta(\boldsymbol{y}))$. In other words, for each $\boldsymbol{g} \in \partial_\delta \hat{F}_\delta(\boldsymbol{x})$, $\boldsymbol{g}$ is of form of $\boldsymbol{g} = \sum \lambda_i \nabla \hat{F}_\delta(\boldsymbol{y}_i)$ where λ_i's are convex coefficients and $\boldsymbol{y}_i \in B(\boldsymbol{x}, \delta)$. (Lin et al., 2022, Theorem 10) has proved that $\nabla \hat{F}_\delta(\boldsymbol{y}_i) \in \partial_\delta F(\boldsymbol{y}_i)$. In addition, since $\|\boldsymbol{y}_i - \boldsymbol{x}\| \leq \delta$, we have $\partial_\delta F(\boldsymbol{y}_i) \subset \partial_{2\delta} F(\boldsymbol{x})$. Consequently, we have

$$\nabla \hat{F}_\delta(\boldsymbol{y}_i) \in \partial_\delta F(\boldsymbol{y}_i) \subset \partial_{2\delta} F(\boldsymbol{x})$$

and thus $\boldsymbol{g} \in \partial_{2\delta} F(\boldsymbol{x})$ for all $\boldsymbol{g} \in \partial_\delta \hat{F}_\delta(\boldsymbol{x})$. We conclude the proof by recalling Definition 2.1 that

$$\|\nabla \hat{F}_\delta(\boldsymbol{x})\|_\delta = \inf\{\|\boldsymbol{g}\| : \boldsymbol{g} \in \partial_\delta \hat{F}_\delta(\boldsymbol{x})\}, \quad \|\nabla F(\boldsymbol{x})\|_{2\delta} = \inf\{\|\boldsymbol{g}\| : \boldsymbol{g} \in \partial_{2\delta} F(\boldsymbol{x})\}$$

and the previous result that $\|\partial_\delta \hat{F}_\delta(\boldsymbol{x}) \subset \partial_{2\delta} F(\boldsymbol{x})$. □

B PROOFS IN SECTION 3

Lemma 3.1. *If $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$, then for any $\delta > 0, \boldsymbol{x} \in \mathbb{R}^d$ and neighboring data batches $z_{1:b}, z'_{1:b}$ of size b,*

$$\mathbb{E}[\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b})] = \nabla \hat{F}_\delta(\boldsymbol{x}), \tag{unbiased}$$

$$\mathbb{E}\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2 \le \tfrac{16dL^2}{b}, \tag{variance}$$

$$\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \textsc{Grad}_{f,\delta}(\boldsymbol{x}, z'_{1:b})\| \le \tfrac{2dL}{b}. \tag{sensitivity}$$

Proof. For simplicity, let $\boldsymbol{g}_{ij} = \frac{d}{2\delta}(f(\boldsymbol{x}+\delta \boldsymbol{u}_{ij}, z_i) - f(\boldsymbol{x}-\delta \boldsymbol{u}_{ij}, z_i))\boldsymbol{u}_{ij}$ so that $\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) = \frac{1}{b}\sum_{i=1}^b \frac{1}{d}\sum_{j=1}^d \boldsymbol{g}_{ij}$. Since f is L-Lipschitz, it follows that $\|\boldsymbol{g}_{ij}\| \le dL$.

First, by $F(\boldsymbol{x}) = \mathbb{E}_z[f(\boldsymbol{x}, z)]$ and Lemma 2.2 (iv),

$$\mathbb{E}_{\boldsymbol{u},z}[\boldsymbol{g}_{ij}] = \mathbb{E}_{\boldsymbol{u}}[\tfrac{d}{2\delta}(F(\boldsymbol{x} + \delta \boldsymbol{u}_{ij}) - F(\boldsymbol{x} - \delta \boldsymbol{u}_{ij}))\boldsymbol{u}_{ij}] = \nabla \hat{F}_\delta(\boldsymbol{x}).$$

Taking the average over i, j gives $\mathbb{E}[\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b})] = \nabla \hat{F}_\delta(\boldsymbol{x})$.

Next, let $\overline{\boldsymbol{g}}_i = \frac{1}{d}\sum_{j=1}^d \boldsymbol{g}_{ij}$ and observe that $\mathbb{E}[\overline{\boldsymbol{g}}_i] = \nabla \hat{F}_\delta(\boldsymbol{x})$. Since $\overline{\boldsymbol{g}}_i - \nabla \hat{F}_\delta(\boldsymbol{x})$ are mean-zero and independent for all i, we have

$$\mathbb{E}[\langle \overline{\boldsymbol{g}}_i - \nabla \hat{F}_\delta(\boldsymbol{x}), \overline{\boldsymbol{g}}_{i'} - \nabla \hat{F}_\delta(\boldsymbol{x})\rangle] = \mathbb{E}_{z_{i'}}[\langle \mathbb{E}_{z_i}[\overline{\boldsymbol{g}}_i - \nabla \hat{F}_\delta(\boldsymbol{x})], \overline{\boldsymbol{g}}_{i'} - \nabla \hat{F}_\delta(\boldsymbol{x})\rangle | z_{i'}] = 0.$$

In other words, all cross-terms are 0. Therefore,

$$\begin{aligned}
&\mathbb{E}\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2 \\
&= \tfrac{1}{b^2}\textstyle\sum_{i=1}^b \mathbb{E}\|\overline{\boldsymbol{g}}_i - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2 \\
&\le \tfrac{1}{b^2}\textstyle\sum_{i=1}^b 2\mathbb{E}\|\overline{\boldsymbol{g}}_i - \nabla \hat{f}_\delta(\boldsymbol{x}, z_i)\|^2 + 2\mathbb{E}\|\nabla \hat{f}(\boldsymbol{x}, z_i) - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2.
\end{aligned} \tag{3}$$

We first evaluate the second term. By Lemma 2.2 (i), $\hat{f}_\delta(\cdot, z_i)$ and F_δ are L-Lipschitz, which implies that $\|\nabla \hat{f}_\delta(\boldsymbol{x}, z_i)\| \le L$ and $\|\nabla \hat{F}_\delta(\boldsymbol{x})\| \le L$. Consequently, $\mathbb{E}\|\nabla \hat{f}(\boldsymbol{x}, z_i) - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2 \le (2L)^2$. Next, note that $\boldsymbol{g}_{ij} - \nabla \hat{f}_\delta(\boldsymbol{x}, z_i) | z_i$ are mean-zero and independent for all j. Therefore,

$$\mathbb{E}\|\overline{\boldsymbol{g}}_i - \nabla \hat{f}_\delta(\boldsymbol{x}, z_i)\|^2 = \tfrac{1}{d^2}\textstyle\sum_{j=1}^d \mathbb{E}\|\boldsymbol{g}_{ij} - \nabla \hat{f}_\delta(\boldsymbol{x}, z_i)\|^2 \le \tfrac{(d+1)^2L^2}{d}.$$

The last inequality follows from $\|\boldsymbol{g}_{ij} - \nabla \hat{f}_\delta(\boldsymbol{x}, z_i)\| \le \|\boldsymbol{g}_{ij}\| + \|\nabla \hat{f}_\delta(\boldsymbol{x}, z_i)\| \le (d+1)L$. Upon substituting back into equation 3, we have

$$\mathbb{E}\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \nabla \hat{F}_\delta(\boldsymbol{x})\|^2 \le \tfrac{2}{b}\left(\tfrac{(d+1)^2L^2}{d} + (2L)^2\right) \le \tfrac{16dL^2}{b}.$$

Finally, let $\boldsymbol{g}'_{ij}, \overline{\boldsymbol{g}}'_i$ be defined in the same way as $\boldsymbol{g}_{ij}, \overline{\boldsymbol{g}}_i$ but using $z'_{1:b}$. Since $z_{1:b}, z'_{1:b}$ are neighboring, $\overline{\boldsymbol{g}}_i - \overline{\boldsymbol{g}}'_i = 0$ for all but one i. Let k be the index of the different data point. Then

$$\|\textsc{Grad}_{f,\delta}(\boldsymbol{x}, z_{1:b}) - \textsc{Grad}_{f,\delta}(\boldsymbol{x}, z'_{1:b})\| = \tfrac{1}{b}\|\overline{\boldsymbol{g}}_k - \overline{\boldsymbol{g}}'_k\| \le \tfrac{1}{bd}\textstyle\sum_{j=1}^d \|\boldsymbol{g}_{kj} - \boldsymbol{g}'_{kj}\| \le \tfrac{2dL}{b}. \quad \square$$

Lemma 3.2. *If $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$, then for any $\delta > 0$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ and neighboring data batches $z_{1:b}, z'_{1:b}$ of size b,*

$$\mathbb{E}[\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b})] = \nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y}), \tag{unbiased}$$

$$\mathbb{E}\|\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - [\nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})]\|^2 \le \tfrac{16dL^2}{b\delta^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2, \tag{variance}$$

$$\|\textsc{Diff}_f(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - \textsc{Diff}_f(\boldsymbol{x}, \boldsymbol{y}, z'_{1:b})\| \le \tfrac{2dL}{b\delta}\|\boldsymbol{x} - \boldsymbol{y}\|. \tag{sensitivity}$$

Proof. Let $\boldsymbol{d}_{ij} = \frac{d}{\delta}(f(\boldsymbol{x}+\delta \boldsymbol{u}_{ij}, z_i) - f(\boldsymbol{y}+\delta \boldsymbol{u}_{ij}, z_i))\boldsymbol{u}_{ij}$. Then it follows that $\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) = \frac{1}{b}\sum_{i=1}^b \frac{1}{d}\sum_{j=1}^d \boldsymbol{d}_{ij}$. Since f is L-Lipschitz, it also holds that $\|\boldsymbol{d}_{ij}\| \le \frac{dL}{\delta}\|\boldsymbol{x} - \boldsymbol{y}\|$.

First, by $F(\boldsymbol{x}) = \mathbb{E}_z[f(\boldsymbol{x}, z)]$ and Lemma 2.2 (iv),

$$\mathbb{E}_{\boldsymbol{u},z}[\boldsymbol{d}_{ij}] = \mathbb{E}_{\boldsymbol{u}}[\tfrac{d}{\delta}(F(\boldsymbol{x} + \delta \boldsymbol{u}_{ij}) - F(\boldsymbol{y} + \delta \boldsymbol{u}_{ij})\boldsymbol{u}_{ij}] = \nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y}).$$

Taking the average over i, j gives $\mathbb{E}_{\boldsymbol{u},z}[\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b})] = \nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})$.

Next, let $\overline{\boldsymbol{d}}_i = \frac{1}{k}\sum_{j=1}^k \boldsymbol{d}_{ij}$, $\Delta_F = \nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})$, and $\Delta_f(z_i) = \nabla \hat{f}_\delta(\boldsymbol{x}, z_i) - \nabla \hat{f}_\delta(\boldsymbol{y}, z_i)$. Observe that $\mathbb{E}[\overline{\boldsymbol{d}}_i] = \Delta_F$ and $\mathbb{E}[\boldsymbol{d}_{ij}|z_i] = \Delta_f(z_i)$. Since $\overline{\boldsymbol{d}}_i - \Delta_F$ are independent and mean-zero for all i, we have

$$\begin{aligned}
&\mathbb{E}\|\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - [\nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})]\|^2 \\
&= \tfrac{1}{b^2}\textstyle\sum_{i=1}^b \mathbb{E}\|\overline{\boldsymbol{d}}_i - \Delta_F\|^2 \\
&\le \tfrac{1}{b^2}\textstyle\sum_{i=1}^b 2\mathbb{E}\|\overline{\boldsymbol{d}}_i - \Delta_f(z_i)\|^2 + 2\mathbb{E}\|\Delta_f(z_i) - \Delta_F\|^2. \qquad (4)
\end{aligned}$$

We first evaluate the second term. By Lemma 2.2 (iii), $\hat{f}_\delta, F_\delta$ are $\frac{\sqrt{d}L}{\delta}$-smooth, which implies that $\|\Delta_f(z_i)\|, \|\Delta_F\| \le \frac{\sqrt{d}L}{\delta}\|\boldsymbol{x} - \boldsymbol{y}\|$. Consequently, $\mathbb{E}\|\Delta_f(z_i) - \Delta_F\|^2 \le (\frac{2\sqrt{d}L}{\delta}\|\boldsymbol{x} - \boldsymbol{y}\|)^2$. Next, since $\boldsymbol{d}_{ij} - \Delta_f(z_i)|z_i$ are mean-zero and independent for all j,

$$\mathbb{E}\|\overline{\boldsymbol{d}}_i - \Delta_f(z_i)\|^2 \le \tfrac{1}{d^2}\textstyle\sum_{j=1}^d \mathbb{E}\|\boldsymbol{d}_{ij} - \Delta_f(z_i)\|^2 \le \frac{(2d^2+2d)L^2}{d\delta^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

The last inequality follows from $\|\boldsymbol{d}_{ij}\| \le \frac{dL}{b}\|\boldsymbol{x} - \boldsymbol{y}\|$ and $\|\Delta_f(z_i)\| \le \frac{\sqrt{d}L}{\delta}\|\boldsymbol{x} - \boldsymbol{y}\|$. Therefore, upon substituting back into equation 4, we have

$$\mathbb{E}\|\textsc{Diff}_{f,\delta}(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - [\nabla \hat{F}_\delta(\boldsymbol{x}) - \nabla \hat{F}_\delta(\boldsymbol{y})]\|^2 \le \frac{16dL^2}{b\delta^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

Finally, let $\boldsymbol{d}'_{ij}, \overline{\boldsymbol{d}}'_i$ be defined in the same way as $\boldsymbol{d}_{ij}, \overline{\boldsymbol{d}}_i$ but using $z'_{1:b}$. Since $z_{1:b}, z'_{1:b}$ are neighboring, $\overline{\boldsymbol{d}}_i - \overline{\boldsymbol{d}}'_i = 0$ for all but one i. Let k be the index of the different data point. Then

$$\|\textsc{Diff}_f(\boldsymbol{x}, \boldsymbol{y}, z_{1:b}) - \textsc{Diff}_f(\boldsymbol{x}, \boldsymbol{y}, z'_{1:b})\| = \tfrac{1}{b}\|\overline{\boldsymbol{d}}_k - \overline{\boldsymbol{d}}'_k\| \le \tfrac{2dL}{b\delta}\|\boldsymbol{x} - \boldsymbol{y}\|. \qquad \square$$

C Proofs in Section 4.2

C.1 Proof of Theorem 4.3

Before we prove the theorem, we first prove a more general composition of RDP mechanisms. For two datasets $Z, Z' \in \mathcal{Z}$, we denote $Z \simeq_q Z'$ if Z, Z' are neighbors and they differ at the q-th entry.

Lemma C.1. *For any domain $\mathcal{Z}$, let $\mathcal{R}^{(i)} : \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(i-1)} \times \mathcal{Z} \to \mathcal{S}^{(i)}$ be a sequence of algorithms such that $\mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, \cdot)$ is (α, ϵ_i)-RDP for all auxiliary inputs $\boldsymbol{x}_{1:i-1} \in \mathcal{S}^{(i)} \times \cdots \times \mathcal{S}^{(i-1)}$. Let $f_{\mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, Z)}$ be the distribution of $\mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, Z)$. Suppose each dataset $Z \in \mathcal{Z}$ has m data, and for each $q \in [m]$, let*

$$\textsc{In}(q) := \{i \in [n] : f_{\mathcal{R}^{(i)}(\cdot, Z)} = f_{\mathcal{R}^{(i)}(\cdot, Z')}, \forall Z \simeq_q Z'\}, \quad \textsc{Out}(q) := [n] \setminus \textsc{In}(q).$$

Let $\mathcal{A}_n : \mathcal{Z} \to \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(n)}$ be the algorithm that given a dataset $Z \in \mathcal{Z}$ sequentially computes $\boldsymbol{x}_i = \mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, z_i)$ for $i \in [n]$ and then outputs $\boldsymbol{x}_{1:n}$. Then $\mathcal{A}_n$ is (α, ϵ)-RDP, where

$$\epsilon \le \max_{q \in [m]} \textstyle\sum_{i \in \textsc{Out}(q)} \epsilon_i.$$

Proof. Suppose $Z \simeq_q Z'$ and let P, Q be the distributions of $\mathcal{A}_n(Z), \mathcal{A}_n(Z')$ respectively. Note that $P = P_1 \times \cdots \times P_n$ where P_i is the distribution of $\mathcal{A}_i(Z)$ given $\mathcal{A}_{i-1}(Z)$, i.e., $P_i(\boldsymbol{x}_i|\boldsymbol{x}_{1:i-1}) = f_{\mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, Z)}(\boldsymbol{x}_i)$. Similarly, $Q = Q_1 \times \cdots \times Q_n$ where $Q_i(\boldsymbol{x}_i|\boldsymbol{x}_{1:i-1}) = f_{\mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, Z')}(\boldsymbol{x}_i)$.

For all $i \in \textsc{In}(q)$, $P_i(\cdot|\boldsymbol{x}_{1:i-1}) = Q_i(\cdot|\boldsymbol{x}_{1:i-1})$ for all $\boldsymbol{x}_{1:i-1}$ by definition of $\textsc{In}(q)$; and for all $i \in \textsc{Out}(q)$, since $\mathcal{R}^{(i)}(\boldsymbol{x}_{1:i-1}, \cdot)$ is (α, ϵ_i)-RDP, $D_\alpha(P_i(\cdot|\boldsymbol{x}_{1:i-1})\|Q_i(\cdot|\boldsymbol{x}_{1:i-1}) \le \epsilon_i$. Therefore,

$$\begin{aligned}
&\int P_i(\boldsymbol{x}_i|\boldsymbol{x}_{1:i-1})^\alpha Q_i(\boldsymbol{x}_i|\boldsymbol{x}_{1:i-1})^{1-\alpha}\, d\boldsymbol{x}_i \\
&= \exp((\alpha-1)D_\alpha(P_i(\cdot|\boldsymbol{x}_{1:i-1})\|Q_i(\cdot|\boldsymbol{x}_{1:i-1})) \le \begin{cases} 1, & i \in \textsc{In}(q), \\ e^{(\alpha-1)\epsilon_i}, & i \in \textsc{Out}(q). \end{cases}
\end{aligned}$$

We then conclude the proof by recursively integrating from $d\boldsymbol{x}_n$ to $d\boldsymbol{x}_1$:

$$e^{(\alpha-1)D_\alpha(P\|Q)} = \int \prod P_i(\boldsymbol{x}_i|\boldsymbol{x}_{1:i-1})^\alpha Q_i(\boldsymbol{x}_i|\boldsymbol{x}_{1:i-1})^{1-\alpha}\, d\boldsymbol{x}_n \cdots d\boldsymbol{x}_1 \le \prod_{i \in \textsc{Out}(q)} e^{(\alpha-1)\epsilon_i}. \qquad \square$$

Next, we introduce several tree notations. Let $k \in \mathbb{N}_0$ and consider a complete binary tree with $n = 2^k$ leaves. Leaves are indexed by $i \in [n]$ (or (i, i) interchangeably) in an ascending order from left to right, and every other node is indexed by (i, j) where i is its left-most child leaf and j is its right-most child leaf. Let $\mathbb{T}$ be the set of all nodes, and let $<$ be a total order on $\mathbb{T}$ such that $(a, b) < (c, d)$ if either $b < d$ or $b = d, a < c$. In other words, we compare rightmost child first and break tie with leftmost child. We assume $(\mathbb{T}, <)$ is sorted in ascending order, then it's valid to label a sequence w.r.t. $\mathbb{T}$ (e.g., we can rewrite $a_1, \ldots, a_{|\mathbb{T}|}$ as $a_{(i,j)}$ for $(i, j) \in \mathbb{T}$). We denote $a_{(c,d)<(a,b)}$ as a short-hand notation of $\{a_{(c,d)} : (c, d) < (a, b)\}$. Now we are ready to prove the theorem.

Theorem 4.3. *Let $\mathcal{X}$ be state space and $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(n)}$ be dataset spaces, and denote $\mathcal{Z}^{(1:i)} = \mathcal{Z}^{(1)} \times \cdots \times \mathcal{Z}^{(i)}$. Let $\mathcal{M}^{(i)} : \mathcal{X}^{i-1} \times \mathcal{Z}^{(i)} \to \mathcal{X}$ be a sequence of algorithms for $i \in [n]$, and let $\mathcal{A} : \mathcal{Z}^{(1:n)} \to \mathcal{X}^n$ be the algorithm that, given a dataset $Z_{1:n} \in \mathcal{Z}^{(1:n)}$, sequentially computes $\boldsymbol{x}_i = \sum_{j=1}^{i} \mathcal{M}^{(j)}(\boldsymbol{x}_{1:j-1}, Z_i) + \textsc{Tree}(i)$ for $i \in [n]$ and then outputs $\boldsymbol{x}_{1:n}$. Suppose for all $i \in [n]$ and neighboring $Z_{1:n}, Z'_{1:n} \in \mathcal{Z}^{(1:n)}$, $\|\mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z_i) - \mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z'_i)\| \le s_i$ for all auxiliary inputs $\boldsymbol{x}_{1:i-1} \in \mathcal{X}^{i-1}$. Then for all $\alpha > 1$, $\mathcal{A}$ is $(\alpha, \alpha\rho^2/2)$-RDP where*

$$\rho \le \sqrt{2 \ln n} \cdot \max_{b \in [n], i \le b} \frac{s_i}{\sigma_b}.$$

Proof. For any $n \in \mathbb{N}$, define $\mathbb{T}$ as he tree set with $2^{\lceil \log_2 n \rceil}$ leaves. Let $\mathcal{Z} = \mathcal{Z}^{(1:n)}$ and $S^{(a,b)} = \mathcal{X}$ for all $(a, b) \in \mathbb{T}$. For each $(a, b) \in \mathbb{T}$, define $\mathcal{R}^{(a,b)} : (\prod_{(c,d)<(a,b)} \mathcal{S}^{(c,d)}) \times \mathcal{Z} \to \mathcal{S}^{(a,b)}$ as

$$\mathcal{R}^{(a,b)}(\boldsymbol{y}_{(c,d)<(a,b)}, Z_{1:n}) = \sum_{i=a}^{b} \mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z_i) + \xi_b, \tag{5}$$

where $\boldsymbol{x}_j = \sum_{(c,d) \in \textsc{Node}(j)} \boldsymbol{y}_{(c,d)}$ for each $j \in [i-1]$ and $\xi_b \sim N(0, \sigma_b^2 I)$. We can check that $\boldsymbol{x}_j$ is well-defined: for all $j \le b$ and for all $(c, d) \in \textsc{Node}(j)$, we have $(c, d) < (a, b)$ by definition of $\textsc{Node}$ and thus $\boldsymbol{x}_{1:i-1}$ can be computed from $\boldsymbol{y}_{(i,j)<(a,b)}$ for all $i \in [a, b]$.

Next, since for all neighboring $Z_{1:n}, Z'_{1:n} \in \mathcal{Z}$, $\|\mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z_i) - \mathcal{M}^{(i)}(\boldsymbol{x}_{1:i-1}, Z'_i)\| \le s_i$, $\mathcal{R}^{(a,b)}(\boldsymbol{y}_{(c,d)<(a,b)}, \cdot)$ has sensitivity bounded by $s_{(a,b)} := \max_{i \in [a,b]} s_i$. Therefore, upon adding Gaussian noise $\mathcal{N}(0, \sigma_b^2 I)$, $\mathcal{R}^{(a,b)}(\boldsymbol{y}_{(c,d)<(a,b)}, \cdot)$ is $(\alpha, \epsilon_{(a,b)})$-RDP where $\epsilon_{(a,b)} = \alpha s_{(a,b)}^2 / 2\sigma_b^2$.

Now we are ready to apply Lemma C.1. Let $\mathcal{A}_{\mathcal{R}} : \mathcal{Z} \to \prod \mathcal{S}^{(a,b)}$ be the composition of $\mathcal{R}$, i.e., given dataset $Z_{1:n}$ sequentially computes $\boldsymbol{y}_{(a,b)} = \mathcal{R}^{(a,b)}(\boldsymbol{y}_{(c,d)<(a,b)}, Z_{1:n})$ and outputs $\boldsymbol{y}_{(a,b) \in \mathbb{T}}$. Then by Lemma C.1, $\mathcal{A}_{\mathcal{R}}$ is (α, ϵ)-RDP where

$$\epsilon \le \max_q \sum_{(a,b) \in \textsc{Out}(q)} \epsilon_{(a,b)} = \max_q \sum_{(a,b) \in \textsc{Out}(q)} \max_{i \in [a.b]} \frac{\alpha s_i^2}{2\sigma_b^2}.$$

Observe that node $(a, b) \in \textsc{Out}(q)$ if and only if the q-th data in $Z_{1:n}$ is in Z_i and $i \in [a, b]$. Since there is exactly one such node in each of $\lceil \log_2 n \rceil + 1$ layers in the binary tree associated with $\mathbb{T}$, we have $|\textsc{Out}(q)| = \lceil \log_2 n \rceil + 1 \le 2 \ln n$. Consequently,

$$\epsilon = \frac{\alpha\rho^2}{2} \le \alpha \ln n \cdot \max_{b \in [n], i \le b} \frac{s_i^2}{\sigma_b^2}.$$

Finally, we conclude the proof by claiming that algorithm $\mathcal{A}$ defined in the theorem can be derived from $\mathcal{A}_{\mathcal{R}}$ after post-processing. Observe that $\sum_{(a,b) \in \textsc{Node}(i)} \sum_{j=a}^{b} = \sum_{j=1}^{i}$, so by equation 5,

$$\begin{aligned} \boldsymbol{x}_i = \sum_{(a,b) \in \textsc{Node}(i)} \boldsymbol{y}_{(a,b)} &= \sum_{(a,b) \in \textsc{Node}(i)} \left(\sum_{j=a}^{b} \mathcal{M}^{(j)}(\boldsymbol{x}_{1:j-1}, Z_j) + \xi_b \right) \\ &= \sum_{j=1}^{i} \mathcal{M}^{(j)}(\boldsymbol{x}_{1:j-1}, Z_j) + \textsc{Tree}(i). \end{aligned}$$

Therefore, $\mathcal{A}$ is indeed a post-processing of $\mathcal{A}_{\mathcal{R}}$, which concludes the proof. □

C.2 Proof of Corollary 4.4

Corollary 4.4. *Suppose $f(\boldsymbol{x}, z)$ is differentiable and L-Lipschitz in $\boldsymbol{x}$ and the domain of $\mathcal{A}$ is bounded by $D = \delta/T$. Let B_1, B_2 satisfies $B_1 \geq TB_2/2$. Then for any $\alpha > 1, \rho > 0$, Algorithm 3 is $(\alpha, \alpha\rho^2/2)$-RDP if we set the noises $\sigma_{1:T}$ in the tree mechanism as*

$$\sigma_t = \sigma := \frac{\sqrt{2\ln T}4dL}{B_2 T\rho}.$$

Proof. Since the private oracle defined in Algorithm 4 uses disjoint data in different iteration $k \in [K]$, Algorithm 3 is a post-processing of $\{(\tilde{\boldsymbol{g}}_1^k, \ldots, \tilde{\boldsymbol{g}}_T^k)\}_{k\in[K]}$. Therefore, it suffices to prove that for each $k \in [K]$, the composition $(\tilde{\boldsymbol{g}}_1^k, \ldots, \tilde{\boldsymbol{g}}_T^k)$ is $(\alpha, \alpha\rho^2/2)$-RDP.

The key is to observe that $(\tilde{\boldsymbol{g}}_1^k, \ldots, \tilde{\boldsymbol{g}}_T^k)$ can be written as the adaptive composition in Theorem 4.3. Let $\boldsymbol{g}_1^k = \mathcal{M}^{(1)}(Z_1^k) \triangleq \textsc{Grad}_{f,\delta}(\boldsymbol{w}_1^k, Z_1^k)$ and $\boldsymbol{d}_i^k = \mathcal{M}^{(i)}(\tilde{\boldsymbol{g}}_{1:i-1}^k, Z_i^k) \triangleq \textsc{Diff}_{f,\delta}(\boldsymbol{w}_i^k, \boldsymbol{w}_{i-1}^k, Z_i^k)$. This is well-defined because $\boldsymbol{w}_i^k$ is a post-processing of $\tilde{\boldsymbol{g}}_{1:i-1}^k$ by construction of O2NC. Therefore, $\boldsymbol{g}_t^k = \boldsymbol{g}_1^k + \sum_{i=2}^t \boldsymbol{d}_i^k = \sum_{i=1}^t \mathcal{M}^{(i)}(\tilde{\boldsymbol{g}}_{1:i-1}^k, Z_i^k)$, and Theorem 4.3 can be applied.

By Lemma 3.1, the sensitivity of $\mathcal{M}^{(1)}$ is bounded by $s_1 = \frac{2dL}{B_1}$. By Remark 4.1 and Lemma 3.2, the sensitivity of $\mathcal{M}^{(i)}(\tilde{\boldsymbol{g}}_{1:i-1}^k, \cdot)$ is bounded by $s_i = \frac{2dL}{B_2\delta}\|\boldsymbol{w}_i^k - \boldsymbol{w}_{i-1}^k\| \leq \frac{4dL}{B_2 T}$ for all $i \geq 2$. Since we assume $B_1 \geq TB_2/2$, it holds that $s_1 \leq s_t$ for all $t \geq 2$. Therefore, upon setting $\sigma_t = \sqrt{2\ln T}s_2/\rho$ for all $t \in [T]$, Theorem 4.3 implies that $(\tilde{\boldsymbol{g}}_1^k, \ldots \tilde{\boldsymbol{g}}_T^k)$ is $(\alpha, \alpha\rho'^2/2)$-RDP where

$$\rho' \leq \sqrt{2\ln T} \cdot \max_{b\in[n], i\leq n} \frac{s_i}{\sigma_b} \leq \sqrt{2\ln T} \cdot \frac{s_2}{\sigma_2} = \rho. \qquad \square$$

C.3 Proof of Corollary 4.5

Corollary 4.5. *Following the assumptions and definitions in Corollary 4.4, for all $t \in [T]$,*

$$\mathbb{E}\|\textsc{Tree}(t)\|^2 \leq 2\ln(t)d\sigma^2 \leq \left(\frac{8\ln(T)d^{3/2}L}{B_2 T\rho}\right)^2.$$

Proof. Recall that $\textsc{Tree}(t) = \sum_{(\cdot,i)\in\textsc{Node}(t)} \xi_i$. Since ξ_i's are independent and $\xi_i \sim \mathcal{N}(0, \sigma_i^2 I)$,

$$\mathbb{E}\|\textsc{Tree}(t)\|^2 = \sum_{(\cdot,i)\in\textsc{Node}(t)} \mathbb{E}\|\xi_i\|^2 = \sum_{(\cdot,i)\in\textsc{Node}(t)} d\sigma_i^2 \leq 2\ln(t)d\sigma^2.$$

The last inequality follows from Remark 4.2 that $|\textsc{Node}(t)| \leq 2\ln(t)$. We then conclude the proof by substituting the value of σ defined in Corollary 4.4. $\square$

D Missing Proofs in Section 4.3

Lemma 4.6. *For any function $F : \mathbb{R}^d \to \mathbb{R}$ that is differentiable and $F(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} F(\boldsymbol{x}) \leq F^*$, if the domain of the OCO algorithm $\mathcal{A}$ is bounded by $D = \delta/T$, then*

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_\delta \leq \frac{F^*}{DKT} + \sum_{k=1}^K \frac{\mathbb{E}\mathrm{Reg}_T(\boldsymbol{u}^k)}{DKT} + \sum_{k=1}^K\sum_{t=1}^T \frac{\mathbb{E}\langle \nabla F(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \boldsymbol{u}^k\rangle}{DKT}.$$

where $\boldsymbol{u}^k = -D\frac{\sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k)}{\|\sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k)\|}$.

Proof. First, by fundamental theorem of calculus, we have

$$\begin{aligned} F(\boldsymbol{x}_{t+1}^k) - F(\boldsymbol{x}_t^k) &= \int_0^1 \langle \nabla F(\boldsymbol{x}_t^k + t(\boldsymbol{x}_{t+1}^k - \boldsymbol{x}_t^k)), \boldsymbol{x}_{t+1}^k - \boldsymbol{x}_t^k\rangle\, dt \\ &= \int_0^1 \langle \nabla F(\boldsymbol{x}_t^k + t\Delta_t^k), \Delta_t^k\rangle\, dt = \mathbb{E}_{s_t^k}\langle \nabla F(\boldsymbol{w}_t^k), \Delta_t^k\rangle. \end{aligned}$$

Next, upon taking the telescopic sum (and recall $x_1^{k+1} = x_{T+1}^k$), we have

$$\mathbb{E}F(\boldsymbol{x}_{T+1}^K) - F(\boldsymbol{x}_1^1) = \sum_{k=1}^K \sum_{t=1}^T \mathbb{E}[F(\boldsymbol{x}_{t+1}^k) - F(\boldsymbol{x}_t^k)] = \sum_{k=1}^K \sum_{t=1}^T \mathbb{E}\langle \nabla F(\boldsymbol{w}_t^k), \Delta_t^k \rangle.$$

Note that $\langle \nabla F(\boldsymbol{w}_t^k), \Delta_t^k \rangle = \langle \nabla F(\boldsymbol{w}_t^k), \boldsymbol{u}^k \rangle + \langle \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \boldsymbol{u}^k \rangle + \langle \nabla F(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \boldsymbol{u}^k \rangle$. Therefore, by definition of $\boldsymbol{u}^k$ and $\mathrm{Reg}_T(\boldsymbol{u}^k)$,

$$\sum_{t=1}^T \langle \nabla F(\boldsymbol{w}_t^k), \Delta_t^k \rangle = -D \left\| \sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k) \right\| + \mathrm{Reg}_T(\boldsymbol{u}^k) + \sum_{t=1}^T \langle \nabla F(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \boldsymbol{u}^k \rangle.$$

Upon rearranging the inequality and applying $F(\boldsymbol{x}_{T+1}^K) - F(\boldsymbol{x}_1^1) \leq F^*$, we have

$$\sum_{k=1}^K D\mathbb{E} \left\| \sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k) \right\| \leq F^* + \sum_{k=1}^K \mathbb{E}\mathrm{Reg}_T(\boldsymbol{u}^k) + \mathbb{E}\sum_{k=1}^K \sum_{t=1}^T \langle \nabla F(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \boldsymbol{u}_t^k \rangle.$$

Finally, note that $\|\boldsymbol{w}_t^k - \overline{w}^k\| \leq DT = \delta$, so $\|\nabla F(\overline{w}^k)\|_\delta \leq \|\frac{1}{T}\sum_{t=1}^T \nabla F(\boldsymbol{w}_t^k)\|$ by definition of $\|\cdot\|_\delta$. Moreover, $\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_\delta = \mathbb{E}[\frac{1}{K}\sum_{i=1}^K \|\nabla F(\overline{w}^k)\|_\delta]$. Therefore, dividing both sides of the inequality by DKT concludes the proof. □

Corollary 4.7. *Suppose $F : \mathbb{R}^d \to \mathbb{R}$ is differentiable, L-Lipschitz, and $F(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} F(\boldsymbol{x}) \leq F^*$, and suppose the domain of $\mathcal{A}$ is bounded by $D = \delta/T$. Then*

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \leq \frac{F^* + 2L\delta}{DKT} + \sum_{k=1}^K \frac{\mathbb{E}\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)}{DKT} + \sum_{k=1}^K \sum_{t=1}^T \frac{\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k \rangle}{DKT}.$$

where $\hat{\boldsymbol{u}}^k = -D\frac{\sum_{t=1}^T \nabla \hat{F}_\delta(\boldsymbol{w}_t^k)}{\|\sum_{t=1}^T \nabla \hat{F}_\delta(\boldsymbol{w}_t^k)\|}$.

Proof. By Lemma 2.2, $\hat{F}_\delta$ is differentiable. Next, since $\|\hat{F}_\delta - F\| \leq L\delta$,

$$\hat{F}_\delta(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} \hat{F}_\delta(\boldsymbol{x}) \leq F(\boldsymbol{x}_1) - \inf_{\boldsymbol{x}} F(\boldsymbol{x}) + 2L\delta = F^* + 2L\delta.$$

Hence, upon applying Lemma 4.6 on the uniform smoothing $\hat{F}_\delta$, we have

$$\mathbb{E}\|\hat{F}_\delta(\overline{\boldsymbol{w}})\|_\delta \leq \frac{F^* + 2L\delta}{DKT} + \sum_{k=1}^K \frac{\mathbb{E}\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)}{DKT} + \sum_{k=1}^K \sum_{t=1}^T \frac{\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k \rangle}{DKT}.$$

Finally, by Corollary 2.3, $\mathbb{E}\|F(\overline{\boldsymbol{w}})\|_{2\delta}$ is also bounded by RHS, which concludes the proof. □

E Comparison with naive approach

To concrete the discussion from Section 4.1, we'll delve into the analysis of the naive private algorithm built on O2NC. Specifically, this algorithm replaces the gradient oracle (as seen in line 5 of Algorithm 3) with a direct zeroth-order estimator, as defined in Algorithm 6. As a remark, $\boldsymbol{g}_t^k$ is an unbiased estimator and $\mathbb{E}\|\boldsymbol{g}_t^k\|^2 \lesssim L^2(\frac{d}{B}+1)$, according to (Kornowski & Shamir, 2023) Remark 6. Moreover, following the same argument as the sensitivity bound in Lemma 3.1, the sensitivity of $\boldsymbol{g}_t^k$ is bounded by $O(dL/B)$. Consequently, if we set $\sigma = dL/B\rho$, each $\tilde{\boldsymbol{g}}_t^k$ achieves $(\alpha, \alpha\rho^2/2)$-RDP.

Although the algorithm appears more straightforward, its convergence analysis is less favorable. Following the proof idea of Theorem 4.9, we first recall the result in Corollary 4.7:

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \leq \frac{F^* + 2L\delta}{DKT} + \sum_{k=1}^K \frac{\mathbb{E}\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)}{DKT} + \sum_{k=1}^K \sum_{t=1}^T \frac{\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k \rangle}{DKT}.$$

Algorithm 6 Naive private gradient oracle

Input: dataset $\mathcal{Z}$, constant B, noise σ
Initialize: Partition $\mathcal{Z}$ into KT subsets Z_t^k of size B. The data size is $|\mathcal{Z}| = M = BKT$.
1: Upon receiving round index k, t and parameter $\boldsymbol{w}_t^k$:
2: Sample $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_B \sim \mathrm{Uniform}(\mathbb{S})$ i.i.d.
3: $\boldsymbol{g}_t^k \leftarrow \frac{1}{B}\sum_{i=1}^{B} \frac{d}{2\delta}(f(\boldsymbol{w}_t^k + \delta \boldsymbol{u}_i, z_i) - f(\boldsymbol{w}_t^k - \delta \boldsymbol{u}_i, z_i))\boldsymbol{u}_i$ where z_i is the i-th data in Z_t^k.
4: Return $\tilde{\boldsymbol{g}}_t^k \leftarrow \boldsymbol{g}_t^k + \xi_t^k$ where $\xi_t^k \sim \mathcal{N}(0, \sigma^2 I)$ i.i.d.

Recall that $\mathbb{E}[\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)] \le D\sqrt{\sum_{t=1}^T \mathbb{E}\|\tilde{\boldsymbol{g}}_t^k\|^2}$. Therefore, we can bound the regret of OSD by

$$\begin{aligned}
\mathbb{E}[\mathrm{Reg}_T(\hat{\boldsymbol{u}}^k)] &\le D\sqrt{\textstyle\sum_{t=1}^T \mathbb{E}\|\boldsymbol{g}_t^k\|^2 + \mathbb{E}\|\xi_t^k\|^2} \\
&\lesssim D\sqrt{\textstyle\sum_{t=1}^T L^2(\frac{d}{B}+1) + \frac{d^3L^2}{B^2\rho^2}} \\
&\lesssim DL\sqrt{T}\left(\tfrac{\sqrt{d}}{\sqrt{B}} + 1 + \tfrac{d^{3/2}}{B\rho}\right).
\end{aligned}$$

Next, we bound the sum of inner product. It's important to note that $\mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k\rangle = 0$ because Δ_t^k is independent of Z_t^k. Therefore, using the martingale concentration bound, we have

$$\begin{aligned}
\textstyle\sum_{t=1}^T \mathbb{E}\langle \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, \Delta_t^k - \hat{\boldsymbol{u}}^k\rangle &= \mathbb{E}\langle \textstyle\sum_{t=1}^T \nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \tilde{\boldsymbol{g}}_t^k, -\hat{\boldsymbol{u}}^k\rangle \\
&\le D\sqrt{\textstyle\sum_{t=1}^T \mathbb{E}\|\nabla \hat{F}_\delta(\boldsymbol{w}_t^k) - \boldsymbol{g}_t^k\|^2 + \mathbb{E}\|\xi_t^k\|^2} \\
&\lesssim DL\sqrt{T}\left(\tfrac{\sqrt{d}}{\sqrt{B}} + 1 + \tfrac{d^{3/2}}{B\rho}\right).
\end{aligned}$$

Upon substituting back into Corollary 4.7, we have

$$\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \lesssim \frac{F^* + L\delta}{DKT} + \frac{L}{\sqrt{T}}\left(\frac{\sqrt{d}}{\sqrt{B}} + 1 + \frac{d^{3/2}}{B\rho}\right)$$

Upon replacing $D = \delta/T$ and $M = KBT$, we have

$$\lesssim \frac{(F^* + L\delta)BT}{\delta M} + \frac{\sqrt{d}L}{\sqrt{BT}} + \frac{L}{\sqrt{T}} + \frac{d^{3/2}L}{B\sqrt{T}\rho}$$

To achieve the optimal non-private rate $O(d\delta^{-1}\epsilon^{-3})$, we need to set $B = 1$. Therefore, upon setting $T = \min\{(\frac{\sqrt{d}L\delta M}{F^*+L\delta})^{2/3}, (\frac{d^{3/2}L\delta M}{(F^*+L\delta)\rho})^{2/3}\}$, we have

$$\lesssim \left(\frac{(F^* + L\delta)dL^2}{\delta M}\right)^{1/3} + \left(\frac{(F^* + L\delta)d^{3/2}L^2}{\rho\delta M}\right)^{1/3}.$$

Equivalently, $\mathbb{E}\|\nabla F(\overline{\boldsymbol{w}})\|_{2\delta} \le \epsilon$ if $M = \Omega\left(d(\frac{F^*L^2}{\delta\epsilon^3} + \frac{L^3}{\epsilon^3}) + d^{3/2}(\frac{F^*L^2}{\rho\delta\epsilon^3} + \frac{L^3}{\rho\epsilon^3})\right)$, whose dominating term is $\Omega(d^{3/2}\rho^{-1}\delta^{-1}\epsilon^{-3})$. In contrast, our carefully designed algorithm only requires $\tilde{\Omega}(d\delta^{-1}\epsilon^{-3} + d^{3/2}\rho^{-1}\delta^{-1}\epsilon^{-2})$ samples, as proved in Theorem 4.9. Notably, the naive algorithm suffers an additional cost of ϵ^{-1} for privacy.