ReSQueing Parallel and Private Stochastic Convex Optimization

Yair Carmon
Tel Aviv University
ycarmon@tauex.tau.ac.il

Arun Jambulapati
University of Washington
jmblpati@uw.edu

Yujia Jin
Stanford University
yujiajin@stanford.edu

Yin Tat Lee
Microsoft Research
yintatlee@microsoft.com

Daogao Liu
University of Washington
dgliu@uw.edu

Aaron Sidford
Stanford University
sidford@stanford.edu

Kevin Tian
Microsoft Research
tiankevin@microsoft.com

***Abstract*—We introduce a new tool for stochastic convex optimization (SCO): a Reweighted Stochastic Query (ReSQue) estimator for the gradient of a function convolved with a (Gaussian) probability density. Combining ReSQue with recent advances in *ball oracle acceleration* [CJJ+20], [ACJ+21], we develop algorithms achieving state-of-the-art complexities for SCO in parallel and private settings. For a SCO objective constrained to the unit ball in $\mathbb{R}^d$, we obtain the following results (up to polylogarithmic factors).**

1) **We give a parallel algorithm obtaining optimization error ϵ_{opt} with $d^{1/3}\epsilon_{\text{opt}}^{-2/3}$ gradient oracle query depth and $d^{1/3}\epsilon_{\text{opt}}^{-2/3} + \epsilon_{\text{opt}}^{-2}$ gradient queries in total, assuming access to a bounded-variance stochastic gradient estimator. For $\epsilon_{\text{opt}} \in [d^{-1}, d^{-1/4}]$, our algorithm matches the state-of-the-art oracle depth of [BJL+19] while maintaining the optimal total work of stochastic gradient descent.**
2) **Given n samples of Lipschitz loss functions, prior works [BFTT19], [BFGT20], [AFKT21], [KLL21] established that if $n \gtrsim d\epsilon_{\text{dp}}^{-2}$, $(\epsilon_{\text{dp}}, \delta)$-differential privacy is attained at no asymptotic cost to the SCO utility. However, these prior works all required a superlinear number of gradient queries. We close this gap for sufficiently large $n \gtrsim d^2\epsilon_{\text{dp}}^{-3}$, by using ReSQue to design an algorithm with near-linear gradient query complexity in this regime.**

***Index Terms*—stochastic optimization, parallel computation, differential privacy**

I. Introduction

Stochastic convex optimization (SCO) is a foundational problem in optimization theory, machine learning, theoretical computer science, and modern data science. Variants of the problem underpin a wide variety of applications in machine learning, statistical inference, operations research, signal processing, and control and systems engineering [Sha07], [SB14]. Moreover, SCO provides a fertile ground for the design and analysis of scalable optimization algorithms such as the celebrated stochastic gradient descent (SGD), which is ubiquitous in machine learning practice [Bot12].

SGD approximately minimizes a function $f : \mathbb{R}^d \to \mathbb{R}$ by iterating $x_{t+1} \leftarrow x_t - \eta g(x_t)$, where $g(x_t)$ is an unbiased estimator to a (sub)gradient of f at iterate x_t. When f is convex, $\mathbb{E}\left\|g(x)\right\|^2 \le 1$ for all x and f is minimized at $x^\star$ in the unit ball, SGD finds an ϵ_{opt}-optimal point (i.e. x satisfying $\mathbb{E}f(x) \le f(x^\star) + \epsilon_{\text{opt}}$) using $O(\epsilon_{\text{opt}}^{-2})$ stochastic gradient evaluations [Bub15]. This complexity is unimprovable without further assumptions [Duc18]; for sufficiently large d, this complexity is optimal even if g is an exact subgradient of f [DG19].

Although SGD is widely-used and theoretically optimal in this simple setting, the algorithm in its basic form has natural limitations. For example, when parallel computational resources are given (i.e., multiple stochastic gradients can be queried in batch), SGD has suboptimal sequential depth in certain regimes [DBW12], [BJL+19]. Furthermore, standard SGD is not differentially private, and existing private[1] SCO algorithms are not as efficient as SGD in terms of gradient evaluation complexity [BST14], [BFTT19], [FKT20], [BFGT20], [AFKT21], [KLL21]. Despite substantial advances in both the parallel and private settings, the optimal complexity of each SCO problem remains open (see Sections I-A and I-B for more precise definitions of problem settings and the state-of-the-art rates, and Section I-C for a broader discussion of related work).

Though seemingly disparate at first glance, in spirit parallelism and privacy impose similar constraints on effective algorithms. Parallel algorithms must find a way to query the oracle multiple times (possibly at multiple points) without using the oracle's output at these points to determine where they were queried. In other words, they cannot be too reliant on a particular outcome to adaptively choose the next query. Likewise, private algorithms must make optimization progress without over-relying on any individual sample to determine the optimization trajectory. In both cases, oracle queries must be suitably robust to preceding oracle outputs.

In this paper, we provide a new stochastic gradient estimation tool which we call *Reweighted Stochastic Query (ReSQue) estimators* (defined more precisely in Section I-D). ReSQue is essentially an efficient parallel method for computing an unbiased estimate of the gradient of a convolution of f with a continuous (e.g. Gaussian) kernel. These estimators are

[1] Throughout this paper, when we use the description "private" without further description we always refer to differential privacy [DR14]. For formal definitions of differential privacy, see Section IV-A.

particularly well-suited for optimizing a convolved function over small Euclidean balls, as they enjoy improved stability properties over these regions. In particular, these local stability properties facilitate tighter control over the stability of SGD-like procedures. We show that careful applications of ReSQue in conjunction with recent advances in accelerated ball-constrained optimization [CJJ+20], [ACJ+21] yield complexity improvements for both parallel and private SCO.

a) Paper organization.: In Sections I-A and I-B respectively, we formally describe the problems of parallel and private SCO we study, stating our results and contextualizing them in the prior literature. We then cover additional related work in Section I-C and, in Section I-D, give an overview of our approach to obtaining these results. In Section I-E, we describe the notation we use throughout.

In Section II-A we introduce our ReSQue estimator and prove some of its fundamental properties. In Section II-B we describe our adaptation of the ball acceleration frameworks of [ACJ+21], [CH22], reducing SCO to minimizing the objective over small Euclidean balls, subproblems which are suitable for ReSQue-based stochastic gradient methods. Finally, in Sections III and IV, we prove our main results for parallel and private SCO (deferring problem statements to Problem 1 and Problem 2), respectively, by providing suitable implementations of our ReSQue ball acceleration framework.

A. Parallelism

In Section III we consider the following formulation of the SCO problem, simplified for the purposes of the introduction. We assume there is a convex function $f : \mathbb{R}^d \to \mathbb{R}$ which can be queried through a *stochastic gradient oracle* g, satisfying $\mathbb{E}g \in \partial f$ and $\mathbb{E}\left\|g\right\|^2 \le 1$. We wish to minimize the restriction of f to the unit Euclidean ball to expected additive error ϵ_{opt}. In the standard sequential setting, SGD achieves this goal using roughly $\epsilon_{\text{opt}}^{-2}$ queries to g; as previously mentioned, this complexity is optimal. A generalization of this formulation is restated in Problem 1 with a variance bound L^2 and a radius bound R, which are both set to 1 here.

In settings where multiple machines can be queried simultaneously, the parallel complexity of an SCO algorithm is a further important measure for consideration. In [Nem94], this problem was formalized in the setting of oracle-based convex optimization, where the goal is to develop iterative methods with a number of parallel query batches to g. In each batch, the algorithm can submit polynomially many queries to g in parallel, and then perform computations based on the outputs of g. The *query depth* of a parallel algorithm in the [Nem94] model is the number of parallel rounds used to query g, and was later considered in stochastic algorithms [DBW12]. Ideally, a parallel SCO algorithm will also have bounded *total queries* (the number of overall queries to g), and bounded *computational depth*, e.g., the parallel depth used by the algorithm treating the depth of each oracle query as $O(1)$. We discuss these three complexity measures more formally in Section III-A.

Method	g query depth	computational depth	# g queries
[Nes18]	ϵ^{-2}	ϵ^{-2}	ϵ^{-2}
[DBW12]	$d^{\frac{1}{4}}\epsilon^{-1}$	$d^{\frac{1}{4}}\epsilon^{-1}$	$d^{\frac{1}{4}}\epsilon^{-1} + \epsilon^{-2}$
[BJL+19]	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}}$	$d^{\frac{4}{3}}\epsilon^{-\frac{8}{3}}$	$d^{\frac{4}{3}}\epsilon^{-\frac{8}{3}}$
[KTE88]	d	d	d
Theorem 1	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}}$	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}} + \epsilon^{-2}$	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}} + \epsilon^{-2}$
Theorem 2	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}}$	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}} + d^{\frac{1}{4}}\epsilon^{-1}$	$d^{\frac{1}{3}}\epsilon^{-\frac{2}{3}} + \epsilon^{-2}$

Table I

COMPARISON OF PARALLEL SCO RESULTS. THE COMPLEXITY OF FINDING A POINT WITH EXPECTED ERROR $\epsilon := \epsilon_{\text{opt}}$ IN PROBLEM 1, WHERE $L = R = 1$. WE HIDE POLYLOGARITHMIC FACTORS IN d AND ϵ^{-1}.

In the low-accuracy regime $\epsilon_{\text{opt}} \ge d^{-1/4}$, recent work [BJL+19] showed that SGD indeed achieves the optimal oracle query depth among parallel algorithms.[2] Moreover, in the high-accuracy regime $\epsilon_{\text{opt}} \le d^{-1}$, cutting plane methods (CPMs) by e.g. [KTE88] (see [JLSW20] for an updated overview) achieve the state-of-the-art oracle query depth of d, up to logarithmic factors in d, ϵ_{opt}.

In the intermediate regime $\epsilon_{\text{opt}} \in [d^{-1}, d^{-1/4}]$, [DBW12], [BJL+19] designed algorithms with oracle query depths that improved upon SGD, as summarized in Table I. In particular, [BJL+19] obtained an algorithm with query depth $\widetilde{O}(d^{1/3}\epsilon_{\text{opt}}^{-2/3})$, which they conjectured is optimal for intermediate ϵ_{opt}. However, the total oracle query complexity of [BJL+19] is $\widetilde{O}(d^{4/3}\epsilon_{\text{opt}}^{-8/3})$, a (fairly large) polynomial factor worse than SGD.

a) Our results.: The main result of Section III is a pair of improved parallel algorithms in the setting of Problem 1. Both of our algorithms achieve the "best of both worlds" between the [BJL+19] parallel algorithm and SGD, in that their oracle query depth is bounded by $\widetilde{O}(d^{1/3}\epsilon_{\text{opt}}^{-2/3})$ (as in [BJL+19]), but their total query complexity matches SGD's in the regime $\epsilon_{\text{opt}} \le d^{-1/4}$. We note that $\epsilon_{\text{opt}} \le d^{-1/4}$ is the regime where a depth of $\widetilde{O}(d^{1/3}\epsilon_{\text{opt}}^{-2/3})$ improves upon [DBW12] and SGD. Our guarantees are formally stated in Theorems 1 and 2, and summarized in Table I.

Our first algorithm (Theorem 1) is based on a batched SGD using our ReSQue estimators, within the "ball acceleration" framework of [ACJ+21] (see Section I-D). By replacing SGD with an accelerated counterpart [GL12], we obtain a further improved *computational depth* in Theorem 2. Theorem 2 simultaneously achieves the query depth of [BJL+19], the computational depth of [DBW12], and the total query complexity of SGD in the intermediate regime $\epsilon_{\text{opt}} \in [d^{-1}, d^{-1/4}]$.

B. Differential privacy

Differential privacy (DP) is a mathematical quantification for privacy risks in algorithms involving data. When performing stochastic convex optimization with respect to a sampled

[2] We omit logarithmic factors when discussing parameter regimes throughout the introduction.

dataset from a population, privacy is frequently a natural practical desideratum [BST14], [EPK14], [Abo16], [App17]. For example, the practitioner may want to privately learn a linear classifier or estimate a regression model or a statistical parameter from measurements.

In this paper, we obtain improved rates for private SCO in the following model, which is standard in the literature and restated in Problem 2 in full generality. Symmetrically to the previous section, in the introduction, we only discuss the specialization of Problem 2 with $L = R = 1$, where L is a Lipschitz parameter and R is a domain size bound. We assume there is a distribution $\mathcal{P}$ over a population $\mathcal{S}$, and we obtain independent samples $\{s_i\}_{i\in[n]} \sim \mathcal{P}$. Every element $s \in \mathcal{S}$ induces a 1-Lipschitz convex function $f(\cdot; s)$, and the goal of SCO is to approximately optimize the population loss $f^{\text{pop}} := \mathbb{E}_{s\sim\mathcal{P}}[f(\cdot; s)]$. The setting of Problem 2 can be viewed as a specialization of Problem 1 which is more compatible with the notion of DP, discussed in more detail in Section IV-A.

The cost of achieving approximate DP with privacy loss parameter ϵ_{dp} (see Section IV-A for definitions) has been studied by a long line of work, starting with [BST14]. The optimal error (i.e., excess population loss) given n samples scales as (omitting logarithmic factors)

$$\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\epsilon_{\text{dp}}}, \tag{1}$$

with matching lower and upper bounds given by [BST14] and [BFTT19], respectively. The $n^{-1/2}$ term is achieved (without privacy considerations) by simple one-pass SGD, i.e., treating sample gradients as unbiased for the population loss, and discarding samples after we query their gradients. Hence, the term $\sqrt{d} \cdot (n\epsilon_{\text{dp}})^{-1}$ can be viewed as the "cost of privacy" in SCO. Assuming that we have access to $n \geq d\epsilon_{\text{dp}}^{-2}$ samples is then natural, as this is the setting where privacy comes at no asymptotic cost from the perspective of the bound (1). Moreover, many real-world problems in data analysis have low intrinsic dimension, meaning that the effective number of degrees of freedom in the optimization problem is much smaller than the ambient dimension [SSTT21], [LLH+22], which can be captured via a dimension-reducing preprocessing step. For these reasons, we primarily focus on the regime when the number of samples n is sufficiently large compared to d.

An unfortunate property of private SCO algorithms achieving error (1) is they all query substantially more than n sample gradients without additional smoothness assumptions [BST14], [BFTT19], [FKT20], [BFGT20], [AFKT21], [KLL21], which can be viewed as a statistical-computational gap. For example, analyses of simple perturbed SGD variants result in query bounds of $\approx n^2$ [BFGT20]. In fact, [BFGT20] conjectured this quadratic complexity was necessary, which was disproven by [AFKT21], [KLL21]. The problem of obtaining the optimal error (1) using n gradient queries has been repeatedly highlighted as an important open problem by the private optimization community, as discussed in [BFGT20], [AFKT21], [KLL21], [ACJ+21] as well as the recent research overview [Tal22].

Qualitatively, optimality of the bound (1) shows that there is no statistical cost of privacy when the number of samples n is large enough, as the solver relies less on any specific sample. A natural first step towards developing optimal private SCO algorithms is to ask a similar qualitative question regarding their computational guarantees. Concretely, given enough samples n, can we develop statistically-optimal SCO algorithms which only query $\approx n$ sample gradients?

a) Our results.: In Section IV, we develop the first private SCO algorithm with this aforementioned computational guarantee. Our algorithm achieves the error bound (1) up to logarithmic factors, as well as a new gradient query complexity. Our result is formally stated in Theorem 4 and summarized in Table II and Figure 1. Up to logarithmic factors, our gradient query complexity is

$$\min\left(n, \frac{n^2\epsilon_{\text{dp}}^2}{d}\right) + \min\left(\frac{(nd)^{\frac{2}{3}}}{\epsilon_{\text{dp}}}, n^{\frac{4}{3}}\epsilon_{\text{dp}}^{\frac{1}{3}}\right).$$

Theorem 4 improves upon the prior state-of-the-art gradient query complexity by polynomial factors whenever $d \ll n^{4/3}$ (omitting ϵ_{dp} dependencies for simplicity). As with prior recent SCO advancements, our result has the appealing property that it achieves the optimal $n^{-1/2}$ error for SCO when $n \gtrsim d\epsilon_{\text{dp}}^{-2}$. Moreover, given $n \gtrsim d^2\epsilon_{\text{dp}}^{-3}$ samples, the gradient query complexity of Theorem 4 improves to $\widetilde{O}(n)$, the first near-linear query complexity for a statistically-optimal private SCO algorithm in any regime. In Table II and Figure 1, we compare our bounds with the prior art.

While there remains a gap between the sample complexity at which our algorithm is statistically optimal, and that at which it is computationally (nearly)-optimal, we find it promising that our result comes within logarithmic factors of achieving the best-of-both-worlds for sufficiently large n. This is a key step towards optimal algorithms for the fundamental problem of private SCO. It is an interesting open question to refine current algorithmic techniques for private SCO to remove this gap, and we are optimistic that the tools developed in this paper will be fruitful in this endeavor.

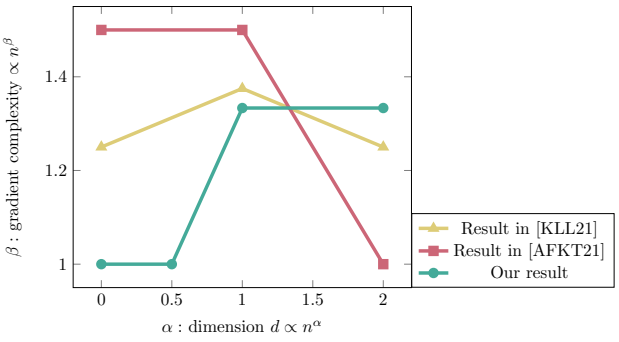

Figure 1. Comparison among our gradient complexity and previous results in [AFKT21], [KLL21] for the non-trivial regime $d \leq n^2$. We omit dependencies on ϵ_{dp} (treated as $\Theta(1)$ in this figure) and logarithmic terms for simplicity.

Method	excess f^{pop} loss	# sample gradient queries
[BST14]	$\frac{\sqrt[4]{d}\log\frac{n}{\delta}}{\sqrt{n}} + \frac{\sqrt{d}\log^2\frac{n}{\delta}}{n\epsilon}$	n^2
[BFTT19]	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}}{n\epsilon}$	$n^{\frac{9}{2}}$
[FKT20]	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}}{n\epsilon}$	n^2
[BFGT20]	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}}{n\epsilon}$	n^2
[AFKT21]	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}}{n\epsilon}$	$n^{\frac{3}{2}} \wedge \frac{n^2\epsilon}{\sqrt{d}}$
[KLL21]	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}}{n\epsilon}$	$n^{\frac{5}{4}}d^{\frac{1}{8}}\sqrt{\epsilon} \wedge \frac{n^{\frac{3}{2}}\epsilon}{d^{\frac{1}{8}}}$
Theorem 4	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}\log^{2.5}\frac{n}{\delta}}{n\epsilon}$	$n + \frac{(nd)^{\frac{2}{3}}}{\epsilon} \wedge \frac{n^2\epsilon^2}{d} + n^{\frac{4}{3}}\epsilon^{\frac{1}{3}}$

Table II
COMPARISON OF PRIVATE SCO RESULTS. THE EXCESS LOSS AND GRADIENT COMPLEXITY OF $(\epsilon := \epsilon_{\text{dp}}, \delta)$-DP IN PROBLEM 2, WHERE $L = R = 1$. WE HIDE POLYLOGARITHMIC FACTORS IN $d, n, \delta^{-1}, \epsilon^{-1}$ IN THE THIRD COLUMN. THE OPTIMAL LOSS [BST14], [SU15] IS ACHIEVED BY ROWS 2-6.

C. Related work

a) Stochastic convex optimization.: Convex optimization is a fundamental task with numerous applications in computer science, operations research, and statistics [BV14], [Bub15], [Nes18], and has been the focus of extensive research over the past several decades. This paper's primary setting of interest is non-smooth (Lipschitz) stochastic convex optimization in private and parallel computational models. Previously, [Gol64] gave a gradient method that used $O(\epsilon^{-2})$ gradient queries to compute a point achieving ϵ error for Lipschitz convex minimization. This rate was shown to be optimal in an information-theoretic sense in [NY83]. The stochastic gradient descent method extends [Gol64] to tolerate randomized, unbiased gradient oracles with bounded second moment: this yields algorithms for Problem 1 and Problem 2 (when privacy is not a consideration).

b) Acceleration.: Since the first proposal of accelerated (momentum-based) methods [Pol64], [Nes83], [Nes03], acceleration has become a central topic in optimization. This work builds on the seminal Monteiro-Svaiter acceleration technique [MS13] and its higher-order variants [GDG+19], [BJL+19]. More specifically, our work follows recent developments in accelerated ball optimization [CJJ+20], [CJJS21], [ACJ+21], which can be viewed as a limiting case of high-order methods. Our algorithms directly leverage error-robust variants of this framework developed by [ACJ+21], [CH22].

c) Parallel SCO.: Recently, parallel optimization has received increasing interest in the context of large-scale machine learning. Speeding up SGD by averaging stochastic gradients across mini-batches is extremely common in practice, and optimal in certain distributed optimization settings; see e.g. [DGBSX12], [DRY18], [WBSS21]. Related to the setting we study are the distributed optimization methods proposed in [SBB+18], which also leverage convolution-based randomized smoothing and apply to both stochastic and deterministic gradient-based methods (but do not focus on parallel depth in the sense of [Nem94]). Finally, lower bounds against the oracle query depth of parallel SCO algorithms in the setting we consider have been an active area of study, e.g. [Nem94], [BS18], [DG19], [BJL+19].

d) Private SCO.: Both the private stochastic convex optimization problem (DP-SCO) and the private empirical risk minimization problem (DP-ERM) are well-studied by the DP community [CM08], [RBHT12], [CMS11], [JT14], [BST14], [KJ16], [FTS17], [ZZMW17], [Wan18], [INS+19], [BFTT19], [FKT20]. In particular, [BST14] shows that the exponential mechanism and noisy stochastic gradient descent achieve the optimal loss for DP-ERM for $(\epsilon_{\text{dp}}, 0)$-DP and $(\epsilon_{\text{dp}}, \delta)$-DP. In follow-up works, [BFTT19], [FKT20] show that one can achieve the optimal loss for DP-SCO as well, by a suitable modification of noisy stochastic gradient descent. However, these algorithms suffer from large (at least quadratic in n) gradient complexities. Under an additional assumption that the loss functions are sufficiently smooth (i.e., have Lipschitz gradient), [FKT20] remedies this issue by obtaining optimal loss and optimal *gradient complexity* under differential privacy. In a different modification of Problem 2's setting (where sample function access is modeled through value oracle queries instead of subgradients), [GLL22] designs an exponential mechanism-based method that uses the optimal value oracle complexity to obtain the optimal SCO loss for non-smooth functions.

Most directly related to our approach are the recent works [KLL21] and [ACJ+21]. Both propose methods improving upon the quadratic gradient complexity achieved by noisy SGD, by using variants of smoothing via Gaussian convolution. The former proposes an algorithm that uses noisy accelerated gradient descent for private SCO with subquadratic gradient complexity. The latter suggests a ball acceleration framework to solve private SCO with linear gradient queries, under a hypothetical algorithm to estimate subproblem solutions. Our work can be viewed as a formalization of the connection between ball acceleration strategies and private SCO as suggested in [ACJ+21], by way of ReSQue estimators, which we use to obtain improved query complexities.

D. Our approach

Here we give an overview of our approach towards obtaining the results outlined in Section I-A and Section I-B. To illustrate and situate our approach, we first briefly discuss prior approaches, their insights that we leverage, and obstacles that we overcome. Then we discuss a common framework based on a new stochastic gradient estimation tool we introduce and call *Reweighted Stochastic Query (ReSQue) estimators* which enables our results on parallel and private SCO. Our new tool is naturally compatible with ball-constrained optimization frameworks, where an optimization problem is localized to a sequence of constrained subproblems (solved to sufficient accuracy), whose solutions are then stitched together. We exploit this synergy, as well as the local stability properties of our ReSQue estimators, to design our SCO algorithms.

We discuss the different instantiations of our framework for parallel and private SCO at the end of this section.

a) Convolutions and prior approaches.: All new results on parallel and private SCO in this paper use the convolution of a function of interest $f : \mathbb{R}^d \to \mathbb{R}$ with a Gaussian density γ_ρ (with covariance $\rho^2 \mathbf{I}_d$), which we denote by $\widehat{f}_\rho$. Such *Gaussian convolutions* have a longer history of facilitating algorithmic advances for SCO. All previous advances on parallel SCO and Lipschitz convex function minimization used Gaussian convolutions, i.e., [DBW12], [BJL+19], as did a state-of-the-art (in some regimes) private SCO algorithm [KLL21]. Each of [DBW12], [KLL21] leverage that $\widehat{f}_\rho$ is a smooth, additive approximation to f, and [BJL+19] further used that the higher derivatives of $\widehat{f}_\rho$ are bounded, as well as the fact that its gradients can be well-approximated within small balls.

As one of our motivating problems, we seek to move beyond the reliance on (high-order) smoothness properties of $\widehat{f}_\rho$, and achieve total work bounds improving upon [BJL+19]. Unfortunately, doing so while following the strategy of [BJL+19] poses an immediate challenge. Though [BJL+19] achieves improved parallel depth bounds for Lipschitz convex optimization, it comes at a cost. Their approach, which relies on the p^{th}-order Lipschitzness of $\widehat{f}_\rho$, would naively involve computing p^{th} derivatives of the objective, and their approach to gradient approximation involves estimating the gradient everywhere inside a ball of sufficient radius. Naïvely, either of these approaches would involve making $\Omega(d)$ queries per parallel step. Removing this cost is one of our main contributions to parallel SCO, and our corresponding development is key to enabling our private SCO results.

b) ReSQue estimators and ball acceleration.: To overcome this bottleneck to prior approaches, we introduce a new tool that capitalizes upon a different property of Gaussian convolutions: the fact that the Gaussian density is locally stable in a small ball around its center. This property is arguably closely related to how [BJL+19] are able to prove that they can approximate the gradients of $\widehat{f}_\rho$ inside a ball. However, rather than building such a complete model of $\widehat{f}_\rho$, we instead use only use this property to suitably implement independent stochastic gradient queries to $\widehat{f}_\rho$.

Given a reference point $\bar{x}$ and a query point x, our proposed estimator for $\nabla \widehat{f}_\rho(x)$ is

$$\begin{aligned} &\text{draw } \xi \sim \mathcal{N}(0, \rho^2 \mathbf{I}_d), \\ &\text{and output estimate } \frac{\gamma_\rho(x - \bar{x} - \xi)}{\gamma_\rho(\xi)} g(\bar{x} + \xi), \end{aligned} \tag{2}$$

where $g(z)$ is an unbiased estimate for a subgradient of f, i.e., $\mathbb{E} g(z) \in \partial f(z)$. That is, to estimate the gradient of $\widehat{f}_\rho$, we simply reweight (stochastic) gradients of f that were queried at random perturbations of reference point $\bar{x}$. This reweighted stochastic query (ReSQue) estimator is unbiased for $\nabla \widehat{f}_\rho(x)$, regardless of $\bar{x}$. However, when $\|x - \bar{x}\| \ll \rho$, i.e., x is contained in a small ball around $\bar{x}$, the reweighting factor $\frac{\gamma_\rho(x-\bar{x}-\xi)}{\gamma_\rho(\xi)}$ is likely to be close to 1. As a result, when g is bounded and x is near $\bar{x}$, the estimator (2) enjoys regularity properties such as moment bounds. Crucially, the stochastic gradient queries performed by ReSQue (at points of the form $\bar{x} + \xi$) *do not depend* on the point x at which we eventually estimate the gradient.

We develop this theory in Section II, but mention one additional property here, which can be thought of as a "relative smoothness" property. We show that when $\|x - x'\|$ is sufficiently smaller than ρ, the *difference* of estimators of the form (2) has many bounded moments, where bounds scale as a function of $\|x - x'\|$. When we couple a sequence of stochastic gradient updates by the randomness used in defining (2), we can use this property to bound how far sequences drift apart. In particular, initially nearby points are likely to stay close. We exploit this property when analyzing the stability of private stochastic gradient descent algorithms later in the paper.

To effectively use these local stability properties of (2), we combine them with an optimization framework called *ball-constrained optimization* [CJJ+20]. It is motivated by the question: given parameters $0 < r < R$, and an oracle which minimizes $f : \mathbb{R}^d$ in a ball of radius r around an input point, how many oracles must we query to optimize f in a ball of larger radius R? It is not hard to show that simply iterating calls to the oracle gives a good solution in roughly $\frac{R}{r}$ queries. In recent work, [CJJ+20] demonstrated that the optimal number of calls scales (up to logarithmic factors) as $(\frac{R}{r})^{2/3}$, and [ACJ+21] gave an approximation-tolerant variant of the [CJJ+20] algorithm. We refer to these algorithms as *ball acceleration*. Roughly, [ACJ+21] shows that running stochastic gradient methods on $\approx (\frac{R}{r})^{2/3}$ subproblems constrained to balls of radius r obtains total gradient query complexity comparable to directly running SGD on the global function of domain radius R.

Importantly, in many structured cases, we have dramatically more freedom in solving these subproblems, compared to the original optimization problem, since we are only required to optimize over a small radius. One natural form of complexity gain from ball acceleration is when there is a much cheaper gradient estimator, which is only locally defined, compared to a global estimator. This was the original motivation for combining ball acceleration with stochastic gradient methods in [CJJS21], which exploited local smoothness of the softmax function; the form of our ReSQue estimator (2) is motivated by the [CJJS21] estimator. In this work, we show that using ReSQue with reference point $\bar{x}$ significantly improves the parallel and private complexity of minimizing the convolution $\widehat{f}_\rho$ inside a ball of radius $r \approx \rho$ centered at $\bar{x}$.

c) Parallel subproblem solvers.: A key property of the ReSQue estimator (2) is that its estimate of $\nabla \widehat{f}_\rho(x)$ is a scalar reweighting of $g(\bar{x} + \xi)$, where $\xi \sim N(0, \rho^2 \mathbf{I}_d)$ and $\bar{x}$ is a fixed reference point. Hence, in each ball subproblem (assuming $r = \rho$), we can make *all* the stochastic gradient queries in parallel, and use the resulting pool of vectors to perform standard (ball-constrained) stochastic optimization using ReSQue. Thus, we solve each ball subproblem with a single parallel stochastic gradient query, and — using ball

acceleration — minimize $\widehat{f}_\rho$ with query depth of roughly $\rho^{-2/3}$. To ensure that $\widehat{f}_\rho$ is a uniform ϵ_{opt}-approximation of the original f, we must set ρ to be roughly $\epsilon_{\text{opt}}/\sqrt{d}$, leading to the claimed $d^{1/3}\epsilon_{\text{opt}}^{-2/3}$ depth bound. Furthermore, the ball acceleration framework guarantees that we require no more than roughly $\rho^{-2/3} + \epsilon_{\text{opt}}^{-2}$ stochastic gradient computations throughout the optimization, yielding the claimed total query bound. However, the computational depth of the algorithm described thus far is roughly $\epsilon_{\text{opt}}^{-2}$, which is no better than SGD. In Section III we combine our approach with the randomized smoothing algorithm of [DBW12] by using an accelerated mini-batched method [GL12] for the ball-constrained stochastic optimization, leading to improved computational depth as summarized in Table I. Our parallel SCO results use the ReSQue/ball acceleration technique in a simpler manner than our private SCO results described next and in Section IV, so we chose to present them first.

d) Private subproblem solvers.: To motivate our improved private SCO solvers, we make the following connection. First, it is straightforward to show that the convolved function $\widehat{f}_\rho$ is $\frac{1}{\rho}$-smooth whenever the underlying function f is Lipschitz. Further, recently [FKT20] obtained a linear gradient query complexity for SCO, under the stronger assumption that each sample function (see Problem 2) is $\lesssim \sqrt{n}$-smooth (for $L = R = 1$ in Problem 2). This bound is satisfied by the result of Gaussian convolution with radius $\frac{1}{\sqrt{n}}$; however, two difficulties arise. First, to preserve the function value approximately up to ϵ_{opt}, we must take a Gaussian convolution of radius $\rho \approx \frac{\epsilon_{\text{opt}}}{\sqrt{d}}$. For ϵ_{opt} in (1), this is much smaller than $\frac{1}{\sqrt{n}}$ in many regimes. Second, we cannot access the exact gradients of the convolved sampled functions. Hence, it is natural to ask: is there a way to simulate the smoothness of the convolved function, under stochastic query access?

Taking a step back, the primary way in which [FKT20] used the smoothness assumption was through the fact that gradient steps on a sufficiently smooth function are *contractive*. This observation is formalized as follows: if $x' \leftarrow x - \eta\nabla f(x)$ and $y' \leftarrow y - \eta\nabla f(y)$, when f is $O(\frac{1}{\eta})$-smooth, then $\|x' - y'\| \le \|x - y\|$. As alluded to earlier, we show that ReSQue estimators (2) allow us to simulate this contractivity up to polylogarithmic factors. We show that by coupling the randomness ξ in the estimator (2), the drift growth in two-point sequences updated with (2) is predictable. We give a careful potential-based argument (see Lemma 5) to bound higher moments of our drift after a sequence of updates using ReSQue estimators, when they are used in an SGD subroutine over a ball of radius $\ll \rho$. This allows for the use of "iterative localization" strategies introduced by [FKT20], based on iterate perturbation via the Gaussian mechanism.

We have not yet dealt with the fact that while this "smoothness simulation" strategy allows us to privately solve *one* constrained ball subproblem, we still need to solve $K \approx (\frac{1}{r})^{2/3}$ ball subproblems to optimize our original function, where $r \ll \rho$ is the radius of each subproblem. Here we rely on arguments based on amplification by subsampling, a common strategy in the private SCO literature [ACG+16], [BBG18]. We set our privacy budget for each ball subproblem to be approximately $(\epsilon_{\text{dp}}, \delta)$ (our final overall budget), before subsampling. We then use solvers by suitably combining the [FKT20] framework and our estimator (2) to solve these ball subproblems using $\approx n \cdot K^{-1/2}$ gradient queries each. Finally, our algorithm obtains the desired

$$\begin{aligned} \text{query complexity:} \quad &\approx \underbrace{\frac{n}{\sqrt{K}}}_{\substack{\text{gradient queries}\\ \text{per subproblem}}} \cdot \underbrace{K}_{\substack{\text{number of}\\ \text{subproblems}}} = n\sqrt{K}, \text{ and} \\ \text{privacy:} \quad &\approx \underbrace{\epsilon_{\text{dp}}}_{\substack{\text{privacy budget}\\ \text{per subproblem}}} \cdot \underbrace{\frac{1}{\sqrt{K}}}_{\substack{\text{subsampling}}} \cdot \underbrace{\sqrt{K}}_{\substack{\text{advanced}\\ \text{composition}}} = \epsilon_{\text{dp}}. \end{aligned} \tag{3}$$

Here we used the standard technique of advanced composition (see e.g. Section 3.5.2, [DR14]) to bound the privacy loss over K consecutive ball subproblems.

Let us briefly derive the resulting complexity bound and explain the bottleneck for improving it further. First, the ball radius r must be set to $\approx \rho$ (the smoothing parameter) for our ReSQue estimators to be well-behaved. Moreover, we have to set $\rho \approx \frac{\epsilon_{\text{opt}}}{\sqrt{d}}$, otherwise the effect of the convolution begins to dominate the optimization error. For $\epsilon_{\text{opt}} \approx \frac{1}{\sqrt{n}} + \sqrt{d}(n\epsilon_{\text{dp}})^{-1}$ (see (1)), this results in $\frac{1}{r} \approx \min(\sqrt{nd}, n\epsilon_{\text{dp}})$. Next, $K \approx (\frac{1}{r})^{2/3}$ is known to be essentially tight for ball acceleration with $R = 1$ [CJJ+20]. For the subproblem accuracies required by the [ACJ+21] ball acceleration framework,[3] known lower bounds on private empirical risk minimization imply that $\approx \frac{n}{\sqrt{K}}$ gradients are necessary for each subproblem to preserve a privacy budget of ϵ_{dp} [BST14]. As subsampling requires the privacy loss before amplification to already be small (see discussion in [Smi09], [BBG18]), all of these parameter choices are optimized, leading to a gradient complexity of $n\sqrt{K}$. For our lower bound on $\frac{1}{r}$, this scales as $\approx \min(n^{4/3}, (nd)^{2/3})$ as we derive in Theorem 4.[4] To go beyond the strategies we employ, it is natural to look towards other privacy amplification arguments (for aggregating ball subproblems) beyond subsampling, which we defer to future work.

Our final algorithm is analyzed through the machinery of Rényi differential privacy (RDP) [Mir17], which allows for more fine-grained control of the effects of composition and subsampling. We modify the standard RDP machinery in two main ways. We define an approximate relaxation and control the failure probability of our relaxation using high moment

[3]These subproblem accuracy requirements cannot be lowered in general, because combined they recover the optimal gradient complexities of SGD over the entire problem domain.

[4]In the low-dimensional regime $d \le n\epsilon_{\text{dp}}^2$, the gradient queries used per subproblem improves to $\frac{\sqrt{nd}}{\epsilon_{\text{dp}}\sqrt{K}}$.

bounds on our drift (see Section IV-B). We also provide an analysis of amplification under subsampling with replacement by modifying the truncated CDP (concentrated DP) tools introduced by [BDRS18], who analyzed subsampling without replacement. Sampling with replacement is crucial in order to guarantee that our ReSQue estimators are unbiased for the empirical risks we minimize when employing a known reduction [FKT20], [KLL21] from private SCO to private regularized empirical risk minimization.

E. Notation

Throughout $\widetilde{O}$ hides polylogarithmic factors in problem parameters. For $n \in \mathbb{N}$, we let $[n] := \{i \mid 1 \le i \le n\}$. For $x \in \mathbb{R}^d$ we let $\|x\|$ denote the Euclidean norm of x, and let $\mathbb{B}_x(r) := \{x' \in \mathbb{R}^d \mid \|x' - x\| \le r\}$ denote a Euclidean ball of radius r centered at x; when x is unspecified we take it to be the origin, i.e., $\mathbb{B}(r) := \{x' \in \mathbb{R}^d \mid \|x'\| \le r\}$. We let $\mathcal{N}(\mu, \mathbf{\Sigma})$ denote a multivariate Gaussian distribution with mean $\mu \in \mathbb{R}^d$ and covariance $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$, and $\mathbf{I}_d$ is the identity matrix in $\mathbb{R}^{d \times d}$. For $\mathcal{K} \subseteq \mathbb{R}^d$, we define the Euclidean projection onto $\mathcal{K}$ by $\Pi_{\mathcal{K}}(x) := \operatorname{argmin}_{x' \in \mathcal{K}} \|x - x'\|$. For $p \in [0, 1]$, we let $\mathrm{Geom}(p)$ denote the geometric distribution with parameter p.

a) Optimization.: We say a function $f : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz if for all $x, x' \in \mathbb{R}^d$ we have $|f(x) - f(x')| \le L \|x - x'\|$. We say f is λ-strongly convex if for all $x, x' \in \mathbb{R}^d$ and $t \in [0, 1]$ we have

$$f(tx + (1-t)y) \le tf(x) + (1-t)f(y) - \frac{\lambda t(1-t)}{2} \|x - x'\|^2.$$

We denote the subdifferential (i.e., set of all subgradients) of a convex function $f : \mathbb{R}^d \to \mathbb{R}$ at $x \in \mathbb{R}^d$ by $\partial f(x)$. Overloading notation, when clear from the context we will write $\partial f(x)$ to denote an arbitrary subgradient.

b) Probability.: Let μ, ν be two probability densities μ, ν on the same probability space Ω. We let $D_{\mathrm{TV}}(\mu, \nu) := \frac{1}{2} \int |\mu(\omega) - \nu(\omega)| \mathrm{d}\omega$ denote the total variation distance. The following fact is straightforward to see and will be frequently used.

Fact 1. *Let $\mathcal{E}$ be any event that occurs with probability at least $1 - \delta$ under the density μ. Then $D_{\mathrm{TV}}(\mu, \mu \mid \mathcal{E}) \le \delta$, where $\mu \mid \mathcal{E}$ denotes the conditional distribution of μ under $\mathcal{E}$.*

For two densities μ, ν, we say that a joint distribution $\Gamma(\mu, \nu)$ over the product space of outcomes is a coupling of μ, ν if for $(x, x') \sim \Gamma(\mu, \nu)$, the marginals of x and x' are μ and ν, respectively. When μ is absolutely continuous with respect to ν, and $\alpha > 1$, we define the α-Rényi divergence by

$$D_\alpha(\mu \| \nu) := \frac{1}{\alpha - 1} \log \left(\int \left(\frac{\mu(\omega)}{\nu(\omega)} \right)^\alpha \mathrm{d}\nu(\omega) \right). \quad (4)$$

D_α is quasiconvex in its arguments, i.e. if $\mu = \mathbb{E}_\xi \mu_\xi$ and $\nu = \mathbb{E}_\xi \nu_\xi$ (where ξ is a random variable, and μ_ξ, ν_ξ are distribution families indexed by ξ), then $D_\alpha(\mu \| \nu) \le \max_\xi D_\alpha(\mu_\xi \| \nu_\xi)$.

II. Framework

We now outline our primary technical innovation, a new gradient estimator for stochastic convex optimization (ReSQue). We define this estimator in Section II-A and prove that it satisfies several local stability properties in a small ball around a "centerpoint" used for its definition. In Section II-B, we then give preliminaries on a "ball acceleration" framework developed in [CJJ+20], [ACJ+21]. This framework aggregates solutions to proximal subproblems defined on small (Euclidean) balls, and uses these subproblem solutions to efficiently solve an optimization problem on a larger domain. Our algorithms in Sections III and IV instantiate the framework of Section II-B with new subproblem solvers enjoying improved parallelism or privacy, based on our new ReSQue estimator.

A. ReSQue estimators

Throughout we use $\gamma_\rho : \mathbb{R}^d \to \mathbb{R}_{\ge 0}$ to denote the probability density function of $\mathcal{N}(0, \rho^2 \mathbf{I}_d)$, i.e., $\gamma_\rho(x) = (2\pi\rho)^{-\frac{d}{2}} \exp(-\frac{1}{2\rho^2} \|x\|^2)$. We first define the Gaussian convolution operation.

Definition 1 (Gaussian convolution)**.** *For a function $f : \mathbb{R}^d \to \mathbb{R}$ we denote its convolution with a Gaussian of covariance $\rho^2 \mathbf{I}_d$ by $\widehat{f}_\rho := f * \gamma_\rho$, i.e.,*

$$\widehat{f}_\rho(x) := \mathbb{E}_{y \sim \mathcal{N}(0, \rho^2 \mathbf{I}_d)} f(x + y) = \int_{y \in \mathbb{R}^n} f(x - y) \gamma_\rho(y) \mathrm{d}y. \quad (5)$$

Three well-known properties of $\widehat{f}_\rho$ are that it is differentiable, that if f is L-Lipschitz, so is $\widehat{f}_\rho$ for any ρ, and that $|\widehat{f}_\rho - f| \le L\rho\sqrt{d}$ pointwise (Lemma 8, [BJL+19]). Next, given a centerpoint $\bar{x}$ and a smoothing radius ρ, we define the associated reweighted stochastic query (ReSQue) estimator.

Definition 2 (ReSQue estimator)**.** *Let $\bar{x} \in \mathbb{R}^d$ and let $f : \mathbb{R}^d \to \mathbb{R}$ be convex. Suppose we have a gradient estimator $g : \mathbb{R}^d \to \mathbb{R}^d$ satisfying $\mathbb{E} g \in \partial f$. We define the* ReSQue estimator of radius ρ *as the random vector*

$$\widetilde{\nabla}^g_{\bar{x}} \widehat{f}_\rho(x) := \frac{\gamma_\rho(x - \bar{x} - \xi)}{\gamma_\rho(\xi)} g(\bar{x} + \xi) \text{ where } \xi \sim \mathcal{N}(0, \rho^2 \mathbf{I}_d),$$

where we first sample ξ, and then independently query g at $\bar{x} + \xi$. When g is deterministically an element of ∂f, we drop the superscript and denote the estimator by $\widetilde{\nabla}_{\bar{x}} \widehat{f}_\rho$.

When g is unbiased for ∂f and enjoys a variance bound, the corresponding ReSQue estimator is unbiased for the convolved function, and inherits a similar variance bound.

Lemma 1. *The estimator in Definition 2 satisfies the following properties, where expectations are taken over both the randomness in ξ and the randomness in g.*

1) *Unbiased:* $\mathbb{E} \widetilde{\nabla}^g_{\bar{x}} \widehat{f}_\rho(x) = \nabla \widehat{f}_\rho(x)$.
2) *Bounded variance: If $\mathbb{E} \|g\|^2 \le L^2$ everywhere, and $x \in \mathbb{B}_{\bar{x}}(\rho)$, then $\mathbb{E} \|\widetilde{\nabla}^g_{\bar{x}} \widehat{f}_\rho(x)\|^2 \le 3L^2$.*

Proof. The first statement follows by expanding the expectation over ξ and g:

$$\begin{aligned}\mathbb{E}_g \int \frac{\gamma_\rho(x-\bar{x}-\xi)}{\gamma_\rho(\xi)} g(\bar{x}+\xi)\gamma_\rho(\xi)\mathrm{d}\xi \\ = \int \frac{\gamma_\rho(x-\bar{x}-\xi)}{\gamma_\rho(\xi)} \partial f(\bar{x}+\xi)\gamma_\rho(\xi)\mathrm{d}\xi \\ = \int \partial f(\bar{x}+\xi)\gamma_\rho(x-\bar{x}-\xi)\mathrm{d}\xi = \nabla \widehat{f}_\rho(x).\end{aligned}$$

The last equality used that the integral is a subgradient of $\widehat{f}_\rho$, and $\widehat{f}_\rho$ is differentiable.

For the second statement, denote $v := x - \bar{x}$ for simplicity. Since f is L-Lipschitz,

$$\begin{aligned}\mathbb{E}\|\widetilde{\nabla}^g_{\bar{x}} \widehat{f}_\rho(x)\|^2 = \mathbb{E}_g \int \frac{(\gamma_\rho(v-\xi))^2}{\gamma_\rho(\xi)} \|g(\bar{x}+\xi)\|^2 \mathrm{d}\xi \\ \le L^2(2\pi\rho)^{-\frac{d}{2}} \\ \cdot \int \exp\left(-\frac{\|v-\xi\|^2}{\rho^2} + \frac{\|\xi\|^2}{2\rho^2}\right)\mathrm{d}\xi.\end{aligned}$$

Next, a standard calculation for Gaussian integrals shows

$$\begin{aligned}\int \exp\left(\frac{2\langle v,\xi\rangle - \|\xi\|^2}{2\rho^2}\right)\mathrm{d}\xi \\ = \exp\left(\frac{\|v\|^2}{2\rho^2}\right)\int \exp\left(-\frac{\|\xi-v\|^2}{2\rho^2}\right)\mathrm{d}\xi \\ = \exp\left(\frac{\|v\|^2}{2\rho^2}\right)(2\pi\rho)^{\frac{d}{2}}.\end{aligned} \tag{6}$$

The statement then follows from (6), which yields

$$\begin{aligned}\int \exp\left(-\frac{\|v-\xi\|^2}{\rho^2} + \frac{\|\xi\|^2}{2\rho^2}\right)\mathrm{d}\xi \\ = \exp\left(-\frac{\|v\|^2}{\rho^2}\right)\int \exp\left(\frac{4\langle v,\xi\rangle - \|\xi\|^2}{2\rho^2}\right)\mathrm{d}\xi \\ = (2\pi\rho)^{\frac{d}{2}} \exp\left(\frac{2\|v\|^2}{\rho^2}\right) \le 3\cdot(2\pi\rho)^{\frac{d}{2}}\end{aligned} \tag{7}$$

and completes the proof of the second statement. □

When the gradient estimator g is deterministically a subgradient of a Lipschitz function, we can show additional properties about ReSQue. The following lemma will be used in Section IV both to obtain higher moment bounds on ReSQue, as well as higher moment bounds on the difference of ReSQue estimators at nearby points, where the bound scales with the distance between the points.

Lemma 2. *If $x, x' \in \mathbb{B}_{\bar{x}}(\frac{\rho}{p})$ for $p \ge 2$ then*

$$\begin{aligned}\mathbb{E}_{\xi\sim\mathcal{N}(0,\rho^2\mathbf{I}_d)}\left[\left(\frac{\gamma_\rho(x-\bar{x}-\xi)}{\gamma_\rho(\xi)}\right)^p\right] \le 2, \\ \mathbb{E}_{\xi\sim\mathcal{N}(0,\rho^2\mathbf{I}_d)}\left[\left|\frac{\gamma_\rho(x-\bar{x}-\xi)-\gamma_\rho(x'-\bar{x}-\xi)}{\gamma_\rho(\xi)}\right|^p\right] \\ \le \left(\frac{24p\,\|x-x'\|}{\rho}\right)^p.\end{aligned}$$

We defer a proof to Appendix A, where a helper calculation (Fact 3) is used to obtain the result.

B. Ball acceleration

We summarize the guarantees of a recent "ball acceleration" framework originally proposed by [CJJ+20]. For specified parameters $0 < r < R$, this framework efficiently aggregates (approximate) solutions to constrained optimization problems over Euclidean balls of radius r to optimize a function over a ball of radius R. Here we give an approximation-tolerant variant of the [CJJ+20] algorithm in Proposition 1, which was developed by [ACJ+21]. Before stating the guarantee, we require the definitions of three types of oracles. In each of the following definitions, for some function $F : \mathbb{R}^d \to \mathbb{R}$, scalars λ, r, and point $\bar{x} \in \mathbb{R}^d$ which are clear from context, we will denote

$$x^\star_{\bar{x},\lambda} := \mathrm{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)}\left\{F(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2\right\}. \tag{8}$$

We mention that in the non-private settings of prior work [ACJ+21], [CH22] (and under slightly different oracle access assumptions), it was shown that the implementation of line search oracles (Definition 3) and stochastic proximal oracles (Definition 5) can be reduced to ball optimization oracles (Definition 4). Indeed, such a result is summarized in Proposition 2 and used in Section III to obtain our parallel SCO algorithms. To tightly quantify the privacy loss of each oracle for developing our SCO algorithms in Section IV (and to implement these oracles under only the function access afforded by Problem 2), we separate out the requirements of each oracle definition separately.

Definition 3 (Line search oracle)**.** *We say $\mathcal{O}_{\mathrm{ls}}$ is a (Δ,λ)-*line search oracle *for $F : \mathbb{R}^d \to \mathbb{R}$ if given $\bar{x} \in \mathbb{R}^d$, $\mathcal{O}_{\mathrm{ls}}$ returns $x \in \mathbb{R}^d$ with*

$$\left\|x - x^\star_{\bar{x},\lambda}\right\| \le \Delta.$$

Definition 4 (Ball optimization oracle)**.** *We say $\mathcal{O}_{\mathrm{bo}}$ is a (ϕ,λ)-*ball optimization oracle *for $F : \mathbb{R}^d \to \mathbb{R}$ if given $\bar{x} \in \mathbb{R}^d$, $\mathcal{O}_{\mathrm{bo}}$ returns $x \in \mathbb{R}^d$ with*

$$\mathbb{E}\left[F(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2\right] \le F(x^\star_{\bar{x},\lambda}) + \frac{\lambda}{2}\left\|x^\star_{\bar{x},\lambda} - \bar{x}\right\|^2 + \phi.$$

Definition 5 (Stochastic proximal oracle)**.** *We say $\mathcal{O}_{\mathrm{sp}}$ is a (Δ,σ,λ)-*stochastic proximal oracle *for $F : \mathbb{R}^d \to \mathbb{R}$ if given $\bar{x} \in \mathbb{R}^d$, $\mathcal{O}_{\mathrm{sp}}$ returns $x \in \mathbb{R}^d$ with*

$$\left\|\mathbb{E}x - x^\star_{\bar{x},\lambda}\right\| \le \frac{\Delta}{\lambda},\ \mathbb{E}\left\|x - x^\star_{\bar{x},\lambda}\right\|^2 \le \frac{\sigma^2}{\lambda^2}.$$

Leveraging Definitions 3, 4, and 5, we state a variant of the main result of [ACJ+21]. Roughly speaking, Proposition 1 states that to optimize a function F over a ball of radius R, it suffices to query $\approx (\frac{R}{r})^{\frac{2}{3}}$ oracles which approximately optimize a sufficiently regularized variant of F over a ball of radius r. We quantify the types of approximate optimization of such regularized functions in Proposition 1, and defer a detailed discussion of how to derive this statement from [ACJ+21] in Appendix A, as it is stated slightly differently in the original work.[5]

Proposition 1. *Let $F : \mathbb{R}^d \to \mathbb{R}$ be L-Lipschitz and convex, and let $x^\star \in \mathbb{B}(R)$. There is an algorithm* BallAccel *(Algorithm 4, [ACJ+21]) taking parameters $r \in [0, R]$ and $\epsilon_{\text{opt}} \in (0, LR]$ with the following guarantee. Define*

$$\kappa := \frac{LR}{\epsilon_{\text{opt}}}, \; K := \left(\frac{R}{r}\right)^{\frac{2}{3}}, \; \lambda_\star := \frac{\epsilon_{\text{opt}} K^2}{R^2} \log^2 \kappa.$$

For a universal constant $C_{\text{ba}} > 0$, BallAccel *runs in at most $C_{\text{ba}} K \log \kappa$ iterations and produces $x \in \mathbb{R}^d$ such that*

$$\mathbb{E} F(x) \le F(x^\star) + \epsilon_{\text{opt}}.$$

Moreover, in each iteration BallAccel *requires the following oracle calls (all for F).*

1) *At most $C_{\text{ba}} \log(\frac{R\kappa}{r})$ calls to a $(\frac{r}{C_{\text{ba}}}, \lambda)$-line search oracle with values of $\lambda \in [\frac{\lambda_\star}{C_{\text{ba}}}, \frac{C_{\text{ba}} L}{\epsilon_{\text{opt}}}]$.*
2) *A single call to $(\frac{\lambda r^2}{C_{\text{ba}} \log^3 \kappa}, \lambda)$-ball optimization oracle with $\lambda \in [\frac{\lambda_\star}{C_{\text{ba}}}, \frac{C_{\text{ba}} L}{\epsilon_{\text{opt}}}]$.*
3) *A single call to $(\frac{\epsilon_{\text{opt}}}{C_{\text{ba}} R}, \frac{\epsilon_{\text{opt}} \sqrt{K}}{C_{\text{ba}} R}, \lambda)$-stochastic proximal oracle with $\lambda \in [\frac{\lambda_\star}{C_{\text{ba}}}, \frac{C_{\text{ba}} L}{\epsilon_{\text{opt}}}]$.*

The optimization framework in Proposition 1 is naturally compatible with our ReSQue estimators, whose stability properties are local in the sense that they hold in balls of radius $\approx \rho$ around the centerpoint $\bar{x}$ (see Lemma 2). Conveniently, BallAccel reduces an optimization problem over a domain of size R to a sequence of approximate optimization problems on potentially much smaller domains of radius r. In Sections III and IV, by instantiating Proposition 1 with $r \approx \rho$, we demonstrate how to use the local stability properties of ReSQue estimators (on smaller balls) to solve constrained subproblems, and consequently design improved parallel and private algorithms.

Finally, as mentioned previously, in settings where privacy is not a consideration, Proposition 1 of [CH22] gives a direct implementation of all the line search and stochastic proximal oracles required by Proposition 1 by reducing them to ball optimization oracles. The statement in [CH22] also assumes access to *function evaluations* in addition to gradient (estimator) queries; however, it is straightforward to use geometric aggregation techniques (see Lemma 11) to bypass this requirement. We give a slight rephrasing of Proposition 1 in [CH22] without the use of function evaluation oracles, and defer further discussion to Appendix 0d where we prove the following.

[5] In particular, we use an error tolerance for the ball optimization oracles, which is slightly larger than in [ACJ+21], following a tighter error analysis given in Proposition 1 of [CH22].

Proposition 2. *Let $F : \mathbb{R}^d \to \mathbb{R}$ be L-Lipschitz and convex, and let $x^\star \in \mathbb{B}(R)$. There is an implementation of* BallAccel *(see Proposition 1) taking parameters $r \in [0, R]$ and $\epsilon_{\text{opt}} \in (0, LR]$ with the following guarantee, where we define $\kappa, K, \lambda_\star$ as in Proposition 1. For a universal constant $C_{\text{ba}} > 0$,* BallAccel *runs in at most $C_{\text{ba}} K \log \kappa$ iterations and produces $x \in \mathbb{R}^d$ such that $\mathbb{E} F(x) \le F(x^\star) + \epsilon_{\text{opt}}$.*

1) *Each iteration makes at most $C_{\text{ba}} \log^2(\frac{R\kappa}{r})$ calls to $(\frac{\lambda r^2}{C_{\text{ba}}}, \lambda)$-ball optimization oracle with values of $\lambda \in [\frac{\lambda_\star}{C_{\text{ba}}}, \frac{C_{\text{ba}} L}{\epsilon_{\text{opt}}}]$.*
2) *For each $j \in [\lceil \log_2 K + C_{\text{ba}} \rceil]$, at most $C_{\text{ba}}^2 \cdot 2^{-j} K \log(\frac{R\kappa}{r})$ iterations query a $(\frac{\lambda r^2}{C_{\text{ba}} 2^j} \cdot \log^{-2}(\frac{R\kappa}{r}), \lambda)$-ball optimization oracle for some $\lambda \in [\frac{\lambda_\star}{C_{\text{ba}}}, \frac{C_{\text{ba}} L}{\epsilon_{\text{opt}}}]$.*

III. Parallel stochastic convex optimization

In this section, we present our main results on parallel convex optimization with improved computational depth and total work. We present our main results below in Theorems 1 and 2, after formally stating our notation and the SCO problem we study in this section.

A. Preliminaries

In this section, we study the following SCO problem, which models access to an objective only through the stochastic gradient oracle.

Problem 1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be convex. We assume there exists a* stochastic gradient oracle *$g : \mathbb{R}^d \to \mathbb{R}^d$ satisfying for all $x \in \mathbb{R}^d$, $\mathbb{E} g(x) \in \partial f(x)$, $\mathbb{E} \|g(x)\|^2 \le L^2$. Our goal is to produce $x \in \mathbb{R}^d$ such that $\mathbb{E} f(x) \le \min_{x^\star \in \mathbb{B}(R)} f(x^\star) + \epsilon_{\text{opt}}$. We define parameter*

$$\kappa := \frac{LR}{\epsilon_{\text{opt}}}. \tag{9}$$

When discussing a parallel algorithm which queries a stochastic gradient oracle, in the sense of Problem 1, we separate its complexity into four parameters. The *query depth* is the maximum number of sequential rounds of interaction with the oracle, where queries are submitted in batch. The *total number of queries* is the total number of oracle queries used by the algorithm. The *computational depth and work* are the sequential depth and total amount of computational work, treating each oracle query as requiring $O(1)$ depth and work. For simplicity we assume that all d-dimensional vector operations have a cost of d when discussing computation.

B. Proofs of Theorems 1 and 2

Theorem 1 (Parallel EpochSGD-based solver)**.** BallAccel *(Proposition 2) using parallel* EpochSGD *(Algorithm 1) as a*

ball optimization oracle solves Problem 1 with expected error ϵ_{opt}*, with*

$$O\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}\log^3(d\kappa)\right) \text{ query depth}$$

$$\text{and } O\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}\log^3(d\kappa)+\kappa^2\log^4(d\kappa)\right) \text{ total queries,}$$

and an additional computational cost of

$$O\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}\log^3(d\kappa)+\kappa^2\log^4(d\kappa)\right) \text{ depth}$$

$$\text{and } O\left(\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}\log^3(d\kappa)+\kappa^2\log^4(d\kappa)\right)\cdot d\right) \text{ work.}$$

Theorem 2 (Parallel AC-SA-based solver)**.** BallAccel *(Proposition 2) using parallel* AC-SA *(Algorithm 2) as a ball optimization oracle solves Problem 1 with expected error* ϵ_{opt}*, with*

$$O\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}\log\kappa\right) \text{ query depth}$$

$$\text{and } O\left(\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}+d^{\frac{1}{4}}\kappa+\kappa^2\right)\log^4(d\kappa)\right) \text{ total queries,}$$

and an additional computational cost of

$$O\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}\log^3(d\kappa)+d^{\frac{1}{4}}\kappa\log^4(d\kappa)\right) \text{ depth}$$

$$\text{and } O\left(\left(d^{\frac{1}{3}}\kappa^{\frac{2}{3}}+d^{\frac{1}{4}}\kappa+\kappa^2\right)\cdot d\log^4(d\kappa)\right) \text{ work.}$$

The query depth, total number of queries, and total work for both of our results are the same (up to logarithmic factors). The main difference is that AC-SA attains an improved computational depth for solving SCO, compared to using EpochSGD. Our results build upon the BallAccel framework in Section II-B, combined with careful parallel implementations of the required ball optimization oracles to achieve improved complexities.

We begin by developing our parallel ball optimization oracles using our ReSQue estimator machinery from Section II-A. First, Proposition 2 reduces Problem 1 to implementation of a ball optimization oracle. Recall that a ball optimization oracle (Definition 4) requires an approximate solution x of a regularized subproblem. In particular, for some accuracy parameter ϕ, and defining $x^\star_{\bar{x},\lambda}$ as in (8), we wish to compute a random $x \in \mathbb{B}_{\bar{x}}(r)$ such that

$$\mathbb{E}\left[\widehat{f}_\rho(x)+\frac{\lambda}{2}\|x-\bar{x}\|^2\right] \le \widehat{f}_\rho(x^\star_{\bar{x},\lambda})+\frac{\lambda}{2}\|x^\star_{\bar{x},\lambda}-\bar{x}\|^2+\phi.$$

Note that such a ball optimization oracle can satisfy the requirements of Proposition 2 with $F \leftarrow \widehat{f}_\rho$, $r \leftarrow \rho$. In particular, Lemma 1 gives a gradient estimator variance bound under the setting $r = \rho$.

a) EpochSGD*.:* We implement EpochSGD [HK14], [ACJ+21], a variant of standard stochastic gradient descent on regularized objective functions, in parallel using the stochastic ReSQue estimator constructed in Definition 2. Our main observation is that the gradient queries in Definition 2 can be implemented in parallel at the beginning of the algorithm. We provide the pseudocode of our parallel implementation of EpochSGD in Algorithm 1 and state its guarantees in Proposition 3.

Algorithm 1: EpochSGD$(f, g, \bar{x}, r, \rho, \lambda, \phi)$

Input: $f : \mathbb{R}^d \to \mathbb{R}$ and $g : \mathbb{R}^d \to \mathbb{R}$ satisfying the assumptions of Problem 1, $\bar{x} \in \mathbb{R}^d$, $r, \rho, \lambda, \phi > 0$
$\eta_1 \leftarrow \frac{1}{4\lambda}$, $T_1 \leftarrow 16$, $T \leftarrow \lceil \frac{48L^2}{\lambda\phi} \rceil$
Sample $\xi_i \sim \mathcal{N}(0, \rho^2\mathbf{I}_d)$, $i \in [2T]$ independently
Query $g(\bar{x}+\xi_i)$ for all $i \in [2T]$ (in parallel)
$x_1^0 \leftarrow \bar{x}$, $k \leftarrow 1$
while $\sum_{j\in[k]} T_j \le T$ **do**
  $x_k^1 \leftarrow \operatorname{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)} \left\{\frac{\eta_k\lambda}{2}\|x-\bar{x}\|^2+\frac{1}{2}\|x-x_k^0\|^2\right\}$
  for $t \in [T_k - 1]$ **do**
    $i \leftarrow \sum_{j\in[k-1]} T_j + t$
    $g_t \leftarrow \frac{\gamma_\rho(x_k^t-\bar{x}-\xi_i)}{\gamma_\rho(\xi_i)} g(\bar{x}+\xi_i)$
    $x_k^{t+1} \leftarrow \operatorname{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)} \langle \eta_k g_t, x\rangle + \frac{\lambda\eta_k}{2}\|x-\bar{x}\|^2+\frac{1}{2}\|x-x_k^t\|^2$
  end
  $x_{k+1}^0 \leftarrow \frac{1}{T_k}\sum_{t\in[T_k]} x_k^t$, $T_{k+1} \leftarrow 2T_k$, $\eta_{k+1} \leftarrow \frac{\eta_k}{2}$, $k \leftarrow k+1$
end
return x_k^0

Proposition 3 (Proposition 3, [ACJ+21])**.** *Let* f, g *satisfy the assumptions of Problem 1. When* $\rho = r$*, Algorithm 1 is a* (ϕ, λ)*-ball optimization oracle for* $\widehat{f}_\rho$ *which makes* $O(\frac{L^2}{\phi\lambda})$ *total queries to* g *with constant query depth, and an additional computational cost of* $O(\frac{L^2}{\phi\lambda})$ *depth and work.*

b) AC-SA*.:* We can also implement AC-SA [GL12], a variant of accelerated gradient descent under stochastic gradient queries, in parallel using stochastic ReSQue estimators. We provide the pseudocode of our parallel implementation of AC-SA in Algorithm 2 and state its guarantees in Lemma 4.

Proposition 4 (Special case of Theorem 1, [GL12])**.** *Let* f, g *satisfy the assumptions of Problem 1. When* $\rho = r$*, Algorithm 2 is a* (ϕ, λ)*-ball optimization oracle for* $\widehat{f}_\rho$ *which makes*

$$O\left(\sqrt{1+\frac{L}{\rho\lambda}}\log\left(\frac{\lambda r^2}{\phi}\right)+\frac{L^2}{\lambda\phi}\right) \text{ total queries}$$

with constant query depth, and an additional computational cost of

$$O\left(\sqrt{1+\frac{L}{\rho\lambda}}\log\left(\frac{\lambda r^2}{\phi}\right)\right) \text{ depth}$$

$$\text{and } O\left(\sqrt{1+\frac{L}{\rho\lambda}}\log\left(\frac{\lambda r^2}{\phi}\right)+\frac{L^2}{\lambda\phi}\right) \text{ work.}$$

Because the statement of Proposition 4 follows from specific parameter choices in the main result in [GL12], we defer a more thorough discussion of how to obtain this result to Appendix 0f.

Algorithm 2: AC-SA$(f, \bar{x}, r, \rho, \lambda, \phi)$

Input: $f : \mathbb{R}^d \to \mathbb{R}$, $g : \mathbb{R}^d \to \mathbb{R}$ satisfying the assumptions of Problem 1, $\bar{x} \in \mathbb{R}^d$, $r, \rho, \lambda, \phi > 0$

$K \leftarrow \lceil \log_2(\frac{\lambda r^2}{\phi}) \rceil$, $T \leftarrow \lceil 4\sqrt{\frac{L}{\rho\lambda}} + 1 \rceil$, $N_k \leftarrow \left\lceil 48 \cdot 2^k \cdot \frac{L^2}{\lambda^2 r^2 T} \right\rceil$ for $k \in [K]$

Sample $\xi_i \sim \mathcal{N}(0, \rho^2 \mathbf{I}_d)$, $i \in [N]$ independently, for $N = T \cdot (\sum_{k \in [K]} N_k)$

Query $g(\bar{x} + \xi_i)$ for all $i \in [N]$ (in parallel)

$x_0^{\mathsf{ag}} \leftarrow \bar{x}$, $x_0 \leftarrow \bar{x}$

for $k \in [K]$ **do**

  for $t \in [T]$ **do**

    $\alpha_t \leftarrow \frac{2}{t+1}$, $\gamma_t \leftarrow \frac{4(\frac{L}{\rho} + \lambda)}{t(t+1)}$

    $x_t^{\mathsf{md}} \leftarrow \frac{(1-\alpha_t)(\lambda+\gamma_t)}{\gamma_t + (1-\alpha_t^2)\lambda} x_{t-1}^{\mathsf{ag}} + \frac{\alpha_t(1-\alpha_t)(\lambda+\gamma_t)}{\gamma_t + (1-\alpha_t^2)\lambda} x_{t-1}$

    $N_{T,[k-1]} \leftarrow T \cdot \sum_{k' \in [k-1]} N_{k'}$

    $\widehat{\nabla} f(x_t^{\mathsf{md}}) \leftarrow \frac{1}{N_k} \sum_{n \in [N_k]} \frac{\gamma_\rho(x_t^{\mathsf{md}} - \bar{x} - \xi_{N_{T,[k-1]}+n})}{\gamma_\rho(\xi_{N_{T,[k-1]}+n})} g(\bar{x} + \xi_{N_{T,[k-1]}+n})$

    $x_t \leftarrow \operatorname{argmin}_{x \in \mathbb{B}_{\bar{x}}(r)} \Psi_t(x)$, where $\Psi_t(x) := \langle \alpha_t \widehat{\nabla} f(x_t^{\mathsf{md}}) + \lambda(x_t^{\mathsf{md}} - \bar{x}), x - x_t \rangle + \frac{\gamma_t + \lambda(1-\alpha_t)}{2} \|x - x_{t-1}\|^2 + \frac{\lambda \alpha_t}{2} \|x - x_t^{\mathsf{md}}\|^2$

    $x_t^{\mathsf{ag}} \leftarrow \alpha_t x_t + (1 - \alpha_t) x_{t-1}^{\mathsf{ag}}$

  end

  $x_0^{\mathsf{ag}} \leftarrow x_T^{\mathsf{ag}}$, $x_0 \leftarrow x_T^{\mathsf{ag}}$

end

Return: x_T^{ag}

c) Main results.: We now use our parallel ball optimization oracles to prove Theorems 1 and 2.

Proofs of Theorems 1 and 2. We use Proposition 2 with $r = \rho = \frac{\epsilon_{\text{opt}}}{\sqrt{dL}}$ on $F \leftarrow \widehat{f}_\rho$, which approximates f to additive ϵ_{opt}, and $x^\star := \arg\min_{x \in \mathbb{B}(R)} f(x)$. Rescaling ϵ_{opt} by a constant from the guarantee of Proposition 2 gives the error claim. For the oracle query depths, note that each ball optimization oracle (whether implemented using Algorithm 1 or Algorithm 2) has constant query depth, and at most $O(\log^2(d\kappa))$ ball optimization oracles are queried per iteration on average. Note that (see Proposition 1)

$$\kappa = \frac{LR}{\epsilon_{\text{opt}}}, \; K = \left(\frac{R}{r}\right)^{\frac{2}{3}} = d^{\frac{1}{3}} \kappa^{\frac{2}{3}},$$
$$\lambda_\star = \frac{\epsilon_{\text{opt}} K^2}{R^2} \log^2 \kappa = \frac{\epsilon_{\text{opt}} d^{\frac{2}{3}} \kappa^{\frac{4}{3}}}{R^2} \log^2 \kappa.$$

For the total oracle queries, computational depth, and work, when implementing each ball optimization oracle with EpochSGD, we have that for $j_{\max} := \lceil \log_2 K + C_{\text{ba}} \rceil$, these are all

$$K \log(d\kappa) \cdot O\left(\sum_{j \in [j_{\max}]} \frac{1}{2^j} \left(\frac{L^2 \cdot 2^j \log^2(d\kappa)}{\lambda_\star^2 r^2} \right) + \left(\frac{L^2}{\lambda_\star^2 r^2} \right) \log^2(d\kappa) \right)$$
$$= O\left(K \log^4(d\kappa) \cdot \frac{L^2}{\lambda_\star^2 r^2} \right) = O\left(\kappa^2 \log^4(d\kappa) \right)$$

due to Proposition 3. The additional terms in the theorem statement are due to the number of ball oracles needed. For the computational depth when implementing each ball optimization oracle with AC-SA we have that (due to Proposition 4), it is bounded by

$$O\left(K \log^3(d\kappa) \cdot \sqrt{\frac{L}{r\lambda_\star}} \log(d\kappa) \right)$$
$$= O\left(K \log^4(d\kappa) \cdot \frac{\sqrt{\kappa}}{K^{\frac{1}{4}}} \right) = O\left(d^{\frac{1}{4}} \kappa \log^4(d\kappa) \right).$$

Finally, for the total oracle queries and work bounds, the bound due to the $\frac{L^2}{\lambda\phi}$ term is as was computed for Theorem 1, and the bound due to the other term is the same as the above display. □

IV. Private stochastic convex optimization

We now develop our main result on an improved gradient complexity for private SCO. First, in Section IV-A, we introduce several variants of differential privacy including a relaxation of Rényi differential privacy [Mir17], which tolerates a small amount of total variation error. Next, in Sections IV-B, IV-C, and IV-D, we build several private stochastic optimization subroutines which will be used in the ball acceleration framework of Proposition 1. Specifically, these subroutines will be called as the oracles in Definitions 3, 4, and 5 with the parameters required by Proposition 1 in the proof of our main result (see (32), (33), and (34)). Finally, in Sections IV-E and IV-F, we give our main results on private ERM and SCO respectively, by leveraging the subroutines we develop. Finally, in Sections IV-E and IV-F, we give our main results on private ERM and SCO respectively, by leveraging the subroutines we develop.

A. Preliminaries

In this section, we study the following specialization of Problem 1 naturally compatible with preserving privacy with respect to samples, through the formalism of DP (to be defined shortly).

Problem 2. *Let $\mathcal{P}$ be a distribution over $\mathcal{S}$, and suppose there is a family of functions indexed by $s \in \mathcal{S}$, such that $f(\cdot; s) : \mathbb{R}^d \to \mathbb{R}$ is convex for all $s \in \mathcal{S}$. Let $\mathcal{D} := \{s_i\}_{i \in [n]}$ consist of n i.i.d. draws from $\mathcal{P}$, and define the* empirical risk *and* population risk *by*

$$f^{\text{erm}}(x) := \frac{1}{n} \sum_{i \in [n]} f(x; s_i) \text{ and } f^{\text{pop}}(x) := \mathbb{E}_{s \sim \mathcal{P}} f(x; s).$$

We denote $f^i := f(\cdot; s_i)$ for all $i \in [n]$, and assume that for all $s \in \mathcal{S}$, $f(\cdot; s)$ is L-Lipschitz. We are given $\mathcal{D}$, and can query subgradients of the "sampled functions" f^i. Our goal is to produce $x \in \mathbb{R}^d$ such that $\mathbb{E} f^{\text{pop}}(x) \le \min_{x^\star \in \mathbb{B}(R)} f^{\text{pop}}(x^\star) + \epsilon_{\text{opt}}$. We again define $\kappa = \frac{LR}{\epsilon_{\text{opt}}}$ as in (9).

In the "one-pass" setting where we only query each ∂f^i a single time, we can treat each ∂f^i as a bounded stochastic gradient of the underlying population risk f^{pop}. We note the related problem of *empirical risk minimization*, i.e., optimizing f^{erm} (in the setting of Problem 2), can also be viewed as a case of Problem 1 where we construct g by querying ∂f^i for $i \sim_{\text{unif.}} [n]$. We design $(\epsilon_{\text{dp}}, \delta)$-DP algorithms for solving Problem 2 which obtain small optimization error for f^{erm} and f^{pop}. To disambiguate, we will always use ϵ_{opt} to denote an optimization error parameter, and ϵ_{dp} to denote a privacy parameter. Our private SCO algorithm will require querying ∂f^i multiple times for some $i \in [n]$, and hence incur bias for the population risk gradient. Throughout the rest of the section, following the notation of Problem 2, we will fix a dataset $\mathcal{D} \in \mathcal{S}^n$ and define the empirical risk f^{erm} and population risk f^{pop} accordingly. We now move on to our privacy definitions.

We say that two datasets $\mathcal{D} = \{s_i\}_{i \in [n]} \in \mathcal{S}^n$ and $\mathcal{D}' = \{s'_i\}_{i \in [n]} \in \mathcal{S}^n$ are *neighboring* if $|\{i \mid s_i \ne s'_i\}| = 1$. We say a mechanism (i.e., a randomized algorithm) $\mathcal{M}$ satisfies $(\epsilon_{\text{dp}}, \delta)$-differential privacy (DP) if, for its output space Ω and all neighboring $\mathcal{D}$, $\mathcal{D}'$, we have for all $S \subseteq \Omega$,

$$\Pr[\mathcal{M}(\mathcal{D}) \in S] \le \exp(\epsilon_{\text{dp}}) \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta. \tag{10}$$

We extensively use the notion of Rényi differential privacy (RDP) due to its compatibility with the subsampling arguments we will use, as well as an approximate relaxation of its definition which we introduce. While it is likely that our results can be recovered (possibly up to logarithmic terms) by accounting for privacy losses via approximate differential privacy, we present our privacy accounting via RDP to simplify calculations.

We say that a mechanism $\mathcal{M}$ satisfies (α, ϵ)-Rényi differential privacy if for all neighboring $\mathcal{D}, \mathcal{D}' \in \mathcal{S}^n$, the α-Rényi divergence (4) satisfies

$$D_\alpha(\mathcal{M}(\mathcal{D}) \| \mathcal{M}(\mathcal{D}')) \le \epsilon. \tag{11}$$

RDP has several useful properties which we now summarize.

Proposition 5 (Propositions 1, 3, and 7, [Mir17])**.** *RDP has the following properties.*

1) *(Composition): Let $\mathcal{M}_1 : \mathcal{S}^n \to \Omega$ satisfy (α, ϵ_1)-RDP and $\mathcal{M}_2 : \mathcal{S}^n \times \Omega \to \Omega'$ satisfy (α, ϵ_2)-RDP for any input in Ω. Then the composition of $\mathcal{M}_2$ and $\mathcal{M}_1$, defined as $\mathcal{M}_2(\mathcal{D}, \mathcal{M}_1(\mathcal{D}))$ satisfies $(\alpha, \epsilon_1 + \epsilon_2)$-RDP.*
2) *(Gaussian mechanism): For $\mu, \mu' \in \mathbb{R}^d$, $D_\alpha(\mathcal{N}(\mu, \sigma^2 \mathbf{I}_d) \| \mathcal{N}(\mu', \sigma^2 \mathbf{I}_d)) \le \frac{\alpha}{2\sigma^2} \|\mu - \mu'\|^2$.*
3) *(Standard DP): If $\mathcal{M}$ satisfies (α, ϵ)-RDP, then for all $\delta \in (0, 1)$, $\mathcal{M}$ satisfies $(\epsilon + \frac{1}{\alpha - 1} \log \frac{1}{\delta}, \delta)$-DP.*

We also use the following definition of approximate Rényi divergence:

$$D_{\alpha,\delta}(\mu \| \nu) := \min_{D_{\text{TV}}(\mu', \mu) \le \delta, D_{\text{TV}}(\nu', \nu) \le \delta} D_\alpha(\mu' \| \nu'). \tag{12}$$

We relax the definition (11) and say that $\mathcal{M}$ satisfies $(\alpha, \epsilon, \delta)$-RDP if for all neighboring $\mathcal{D}$, $\mathcal{D}' \in \mathcal{S}^n$, recalling definition (12),

$$D_{\alpha,\delta}(\mathcal{M}(\mathcal{D}) \| \mathcal{M}(\mathcal{D}')) \le \epsilon.$$

The following is then immediate from Proposition 5, and our definition of approximate RDP, by coupling the output distributions with the distributions realizing the minimum (12).

Corollary 1. *If $\mathcal{M}$ satisfies $(\alpha, \epsilon, \delta)$-RDP, then for all $\delta' \in (0, 1)$, $\mathcal{M}$ satisfies $(\epsilon_{\text{dp}}, \delta' + (1 + \exp(\epsilon_{\text{dp}}))\delta)$-DP for $\epsilon_{\text{dp}} := \epsilon + \frac{1}{\alpha - 1} \log \frac{1}{\delta'}$.*

Proof. Let μ, ν be within total variation δ of $\mathcal{M}(\mathcal{D})$ and $\mathcal{M}(\mathcal{D}')$, such that $D_\alpha(\mu \| \nu) \le \epsilon$ and hence for any event S,

$$\Pr_{\omega \sim \mu}[\omega \in S] \le \exp(\epsilon_{\text{dp}}) \Pr_{\omega \sim \nu}[\omega \in S] + \delta'.$$

Combining the above with

$$\Pr_{\omega \sim \mathcal{M}(\mathcal{D})}[\omega \in S] - \delta \le \Pr_{\omega \sim \mu}[\omega \in S],$$
$$\Pr_{\omega \sim \nu}[\omega \in S] \le \Pr_{\omega \sim \mathcal{M}(\mathcal{D}')}[\omega \in S] + \delta,$$

we have

$$\begin{aligned}\Pr_{\omega \sim \mathcal{M}(\mathcal{D})}[\omega \in S] &\le \exp(\epsilon_{\text{dp}}) \Pr_{\omega \sim \nu}[\omega \in S] + \delta' + \delta \\ &\le \exp(\epsilon_{\text{dp}}) \Pr_{\omega \sim \mathcal{M}(\mathcal{D}')}[\omega \in S] \\ &\quad + \delta' + (1 + \exp(\epsilon_{\text{dp}}))\delta.\end{aligned}$$

□

Finally, our approximate RDP notion enjoys a composition property similar to standard RDP.

Lemma 3. *Let $\mathcal{M}_1 : \mathcal{S}^n \to \Omega$ satisfy $(\alpha, \epsilon_1, \delta_1)$-RDP and $\mathcal{M}_2 : \mathcal{S}^n \times \Omega \to \Omega'$ satisfy $(\alpha, \epsilon_2, \delta_2)$-RDP for any input in Ω. Then the composition of $\mathcal{M}_2$ and $\mathcal{M}_1$, defined as $\mathcal{M}_2(\mathcal{D}, \mathcal{M}_1(\mathcal{D}))$ satisfies $(\alpha, \epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-RDP.*

Proof. Let $\mathcal{D}$, $\mathcal{D}'$ be neighboring datasets, and let μ, μ' be distributions within total variation δ_1 of $\mathcal{M}_1(\mathcal{D})$, $\mathcal{M}_1(\mathcal{D}')$ realizing the bound $D_\alpha(\mu \| \mu') \le \epsilon_1$. For any $\omega \in \Omega$, similarly let ν_ω, ν'_ω be the distributions within total variation δ_2 of $\mathcal{M}_2(\mathcal{D}, \omega)$ and $\mathcal{M}_2(\mathcal{D}', \omega)$ realizing the bound $D_\alpha(\nu_\omega \| \nu'_\omega) \le \epsilon_2$. Finally, let P_1 be the distribution of $\omega \in \Omega$ according to $\mathcal{M}_1(\mathcal{D})$, and Q_1 to be the distribution of $\mathcal{M}_1(\mathcal{D}')$; similarly, let $P_{2,\omega}$, $Q_{2,\omega}$ be the distributions of $\omega' \in \Omega'$ according to

Algorithm 3: Subsampled ReSQued ERM solver, convex case

Input: $\bar{x} \in \mathbb{R}^d$, ball radius, convolution radius, and privacy parameter $r, \rho, \beta > 0$, dataset $\mathcal{D} \in \mathcal{S}^n$, iteration count $T \in \mathbb{N}$

$\widehat{T} \leftarrow 2^{\lfloor \log_2 T \rfloor}$, $k \leftarrow \log_2 \widehat{T}$, $\eta \leftarrow \frac{r}{L}\min(\frac{1}{\sqrt{T}}, \frac{\beta}{\sqrt{d}})$, $x_0 \leftarrow \bar{x}$

for $i \in [k]$ **do**

  $T_i \leftarrow 2^{-i}\widehat{T}$, $\eta_i \leftarrow 4^{-i}\eta$, $\sigma_i \leftarrow \frac{L\eta_i}{\beta}$

  $y_0 \leftarrow x_{i-1}$

  for $j \in [T_i]$ **do**

    $z_{i,j} \sim_{\text{unif.}} [n]$

    $y_j \leftarrow \Pi_{\mathbb{B}_{\bar{x}}(r)}(y_{j-1} - \eta_i \widetilde{\nabla}_{\bar{x}} \widehat{f}_\rho^{z_{i,j}}(y_{j-1}))$;

    ▷ PSGD step using ReSQue (See Definition 2) for a subsampled function. Lemma 5 denotes the random Gaussian sample by $\xi_{i,j}$.

  end

  $\bar{y}_i \leftarrow \frac{1}{T_i}\sum_{j\in[T_i]} y_j$

  $x_i \leftarrow \bar{y}_i + \zeta_i$, for $\zeta_i \sim \mathcal{N}(0, \sigma_i^2 \mathbf{I}_d)$

end

return x_k

$\mathcal{M}_2(\mathcal{D}, \omega)$ and $\mathcal{M}_2(\mathcal{D}', \omega)$. We first note that by a union bound,

$$D_{\text{TV}}\left(\int \nu_\omega(\omega')\mu(\omega)d\omega d\omega', \int P_1(\omega)P_{2,\omega}(\omega')d\omega d\omega'\right) \le \delta_1 + \delta_2,$$

$$D_{\text{TV}}\left(\int \nu'_\omega(\omega')\mu'(\omega)d\omega d\omega', \int Q_1(\omega)Q_{2,\omega}(\omega')d\omega d\omega'\right) \le \delta_1 + \delta_2.$$

Finally, by Proposition 1 of [Mir17], we have

$$D_\alpha\left(\int \nu_\omega(\omega')\mu(\omega)d\omega d\omega' \middle\| \int \nu'_\omega(\omega')\mu'(\omega)d\omega d\omega'\right) \le \epsilon_1 + \epsilon_2.$$

Combining the above two displays yields the claim. □

B. Subsampled smoothed ERM solver: the convex case

We give an ERM algorithm that takes as input a dataset $\mathcal{D} \in \mathcal{S}^n$, parameters $T \in \mathbb{N}$ and $r, \rho, \beta > 0$, and a center point $\bar{x} \in \mathbb{R}^d$. Our algorithm is based on a localization approach introduced by [FKT20] which repeatedly decreases a domain size to bound the error due to adding noise for privacy. In particular we will obtain an error bound on $\widehat{f_\rho^{\text{erm}}}$ with respect to the set $\mathbb{B}_{\bar{x}}(r)$, using at most T calls to the ReSQue estimator in Definition 2 with a deterministic subgradient oracle. Here we recall that f^{erm} is defined as in Problem 2, and $\widehat{f_\rho^{\text{erm}}}$ is correspondingly defined as in Definition 1. Importantly, our ERM algorithm developed in this section attains RDP bounds improving with the subsampling parameter $\frac{T}{n}$ when $T \ll n$, due to only querying T random samples in our dataset.

We summarize our optimization and privacy guarantees on Algorithm 3 in the following. The proof follows by combining Lemma 4 (the utility bound) and Lemma 7 (the privacy bound).

Proposition 6. *Let $x_{\bar{x}}^\star \in \operatorname{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)} \widehat{f_\rho^{\text{erm}}}(x)$. Algorithm 3 uses at most T gradients and produces $x \in \mathbb{B}_{\bar{x}}(r)$ such that, for a universal constant C_{cvx},*

$$\mathbb{E}\left[\widehat{f_\rho^{\text{erm}}}(x)\right] - \widehat{f_\rho^{\text{erm}}}(x_{\bar{x}}^\star) \le C_{\text{cvx}} Lr\left(\frac{\sqrt{d}}{\beta T} + \frac{1}{\sqrt{T}}\right).$$

Moreover, there is a universal constant $C_{\text{priv}} \ge 1$, such that if $\frac{T}{n} \le \frac{1}{C_{\text{priv}}}$, $\beta^2 \log^2(\frac{1}{\delta}) \le \frac{1}{C_{\text{priv}}}$, $\delta \in (0, \frac{1}{6})$, and $\frac{\rho}{r} \ge C_{\text{priv}} \log^2(\frac{\log T}{\delta})$, Algorithm 3 satisfies $(\alpha, \alpha\tau, \delta)$-RDP for

$$\tau := C_{\text{priv}}\left(\beta \log\left(\frac{1}{\delta}\right)\cdot\frac{T}{n}\right)^2 \text{ and } \alpha \in \left(1, \frac{1}{C_{\text{priv}}\beta^2\log^2(\frac{1}{\delta})}\right).$$

a) Utility analysis.: We begin by proving a utility guarantee for Algorithm 3, following [FKT20].

Lemma 4. *Let $x_{\bar{x}}^\star := \operatorname{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)} \widehat{f_\rho^{\text{erm}}}(x)$. We have, for a universal constant C_{cvx},*

$$\mathbb{E}\left[\widehat{f_\rho^{\text{erm}}}(x_k)\right] - \widehat{f_\rho^{\text{erm}}}(x_{\bar{x}}^\star) \le C_{\text{cvx}} Lr\left(\frac{\sqrt{d}}{\beta T} + \frac{1}{\sqrt{T}}\right).$$

Proof. Denote $F := \widehat{f_\rho^{\text{erm}}}$, $\bar{y}_0 := x_{\bar{x}}^\star$, and $\zeta_0 := \bar{x} - x_{\bar{x}}^\star$, where by assumption $\|\zeta_0\| \le r$. We begin by observing that in each run of Line 8, by combining the first property in Lemma 1 with the definition of f^{erm}, we have that $\mathbb{E}\left[\widetilde{\nabla}_{\bar{x}} \widehat{f}_\rho^{z_{i,j}}(y_{j-1}) \mid y_{j-1}\right] \in \partial F(y_{j-1})$. Moreover, by the second property in Lemma 1 and the fact that $f^{z_{i,j}}$ is L-Lipschitz,

$$\mathbb{E}\left\|\widetilde{\nabla}_{\bar{x}} \widehat{f}_\rho^{z_{i,j}}(y_{j-1})\right\|^2 \le 3L^2.$$

We thus have

$$\begin{aligned}
\mathbb{E}\left[F(x_k)\right] - F(x_{\bar{x}}^\star) &= \sum_{i\in[k]} \mathbb{E}[F(\bar{y}_i) - F(\bar{y}_{i-1})] \\
&\quad + \mathbb{E}\left[F(x_k) - F(\bar{y}_k)\right] \\
&\le \sum_{i\in[k]}\left(\frac{\mathbb{E}\left[\|x_{i-1} - \bar{y}_{i-1}\|^2\right]}{2\eta_i T_i} + \frac{3\eta_i L^2}{2}\right) \\
&\quad + L\mathbb{E}\left[\|x_k - \bar{y}_k\|\right] \\
&\le \frac{8r^2}{\eta T} + 4\sum_{i\in[k-1]}\frac{\sigma_i^2 d}{\eta_i T_i} \\
&\quad + \sum_{i\in[k]}\frac{3\eta_i L^2}{2} + L\sigma_k\sqrt{d}.
\end{aligned} \tag{13}$$

In the third line, we used standard regret guarantees on projected stochastic gradient descent, e.g. Lemma 7 of [HK14], where we used that all $\bar{y}_i \in \mathbb{B}_{\bar{x}}(r)$; in the fifth line, we used

$$\mathbb{E}[\|x_k - \bar{y}_k\|] \le \sqrt{\mathbb{E}\left[\|x_k - \bar{y}_k\|^2\right]} = \sqrt{\mathbb{E}\left[\|\zeta_k\|^2\right]} = \sigma_k\sqrt{d}$$

by Jensen's inequality. Continuing, we have by our choice of parameters that $\frac{\sigma_i^2}{\eta_i T_i} \le 2^{-i}\frac{L^2\eta}{2\beta^2\widehat{T}}$, hence

$$\begin{aligned}\mathbb{E}\left[F(x_k)\right] - F(x_{\bar{x}}^\star) &\le \frac{8r^2}{\eta T} + \frac{4L^2\eta d}{\beta^2\widehat{T}} + \frac{3\eta L^2}{2} \\ &\quad + \frac{L^2\eta\sqrt{d}}{\beta}\cdot\frac{1}{\widehat{T}^2} \\ &\le \left(\frac{8Lr}{\sqrt{T}} + \frac{8Lr\sqrt{d}}{\beta T}\right) + \frac{8Lr\sqrt{d}}{\beta T} \\ &\quad + \frac{3Lr}{2\sqrt{T}} + \frac{Lr}{\sqrt{T}}.\end{aligned}$$

Here we used that $2\widehat{T} \ge T$ and $\widehat{T}^2 \ge \sqrt{T}$, for all $T \in \mathbb{N}$. □

b) Privacy analysis.: We now show that our algorithm satisfies a strong (approximate) RDP guarantee. Let $\mathcal{D}' = \{s_i'\}_{i\in[n]} \in \mathcal{S}^n$ be such that $\mathcal{D} = \{s_i\}_{i\in[n]}$ and $\mathcal{D}'$ are neighboring, and without loss of generality assume $s_1' \ne s_1$. Define the multiset

$$\mathcal{I} := \{z_{i,j} \mid i \in [k], j \in [T_i]\} \tag{14}$$

to contain all sampled indices in $[n]$ throughout Algorithm 3. We begin by giving an (approximate) RDP guarantee conditioned on the number of times "1" appears in $\mathcal{I}$. The proof of Lemma 5 is primarily based on providing a potential-based proof of a "drift bound," i.e., how far away iterates produced by two neighboring datasets drift apart (coupling all other randomness used). To carry out this potential proof, we rely on the local stability properties afforded by Lemma 2.

Lemma 5. *Define $\mathcal{I}$ as in (14) in one call to Algorithm 3. Let $\mathcal{I}$ be deterministic (i.e., this statement is conditioned on the realization of $\mathcal{I}$). Let b be the number of times the index 1 appears in $\mathcal{I}$. Let μ be the distribution of the output of Algorithm 3 run on $\mathcal{D}$, and μ' be the distribution when run on $\mathcal{D}'$, such that $\mathcal{D}$ and $\mathcal{D}'$ are neighboring and differ in the first entry, and the only randomness is in the Gaussian samples used to define ReSQue estimators and on Line 11. Suppose $\frac{\rho}{r} \ge 1728\log^2(\frac{\log T}{\delta})$. Then we have for any $\alpha > 1$,*

$$D_{\alpha,\delta}(\mu\|\mu') \le 1500\alpha\beta^2 b^2.$$

Proof. Throughout this proof we treat $\mathcal{I}$ as fixed with b occurrences of the index 1. Let b_i be the number of times 1 appears in $\mathcal{I}_i := \{z_{i,j} \mid j \in [T_i]\}$, such that $\sum_{i\in[k]} b_i = b$. We first analyze the privacy guarantee of one loop, and then analyze the privacy of the whole algorithm.

We begin by fixing some $i \in [k]$, and analyzing the RDP of the i^{th} outer loop in Algorithm 3, conditioned on the starting point y_0. Consider a particular realization of the T_i Gaussian samples used in implementing Line 8, $\Xi_i := \{\xi_{i,j}\}_{j\in[T_i]}$, where we let $\xi_{i,j} \sim \mathcal{N}(0, \rho^2\mathbf{I}_d)$ denote the Gaussian sample used to define the update to y_{j-1}. Conditioned on the values of $\mathcal{I}_i$, Ξ_i, the i^{th} outer loop in Algorithm 3 (before adding ζ_i in Line 11) is a deterministic map. For a given realization of $\mathcal{I}_i$ and Ξ_i, we abuse notation and denote $\{y_j\}_{j\in[T_i]}$ to be the iterates of the i^{th} outer loop in Algorithm 3 using the dataset $\mathcal{D}$ starting at y_0, and $\{y_j'\}_{j\in[T_i]}$ similarly using $\mathcal{D}'$. Finally, define

$$\Phi_j := \left\|y_j - y_j'\right\|^2,\ p := \left\lceil 5\log\left(\frac{\log T}{\delta}\right)\right\rceil.$$

In the following parts of the proof, we will bound for this p the quantity $\mathbb{E}\Phi_{T_i}^p$, to show that with high probability it remains small at the end of the loop, regardless of the location of the 1 indices.

Potential growth: iterates with $z_{i,j} \ne 1$. We first bound the potential growth in any iteration $j \in [T_i]$ where $z_{i,j} \ne 1$. Fix y_0, y_0' and $\{\xi_{i,t}\}_{t\in[j-1]}$, so that Φ_{j-1} is deterministic. We have (taking expectations over only $\xi_{i,j}$),

$$\mathbb{E}_{\xi_{i,j}}\Phi_j^p \le \mathbb{E}\left(\Phi_{j-1} + A_j + B_j\right)^p, \tag{15}$$

where

$$\begin{aligned}A_j &:= -2\eta_i Z_j\left\langle \partial f^{z_{i,j}}(\bar{x}+\xi_{i,j}), y_{j-1} - y_{j-1}'\right\rangle, \\ B_j &:= \eta_i^2 Z_j^2\left\|\partial f^{z_{i,j}}(\bar{x}+\xi_{i,j})\right\|^2, \quad \text{and} \\ Z_j &:= \frac{\gamma_\rho(y_{j-1}-\bar{x}-\xi_{i,j}) - \gamma_\rho(y_{j-1}'-\bar{x}-\xi_{i,j})}{\gamma_\rho(\xi_{i,j})}.\end{aligned}$$

The inequality in (15) follows from expanding the definition of the update to Φ_j before projection, and then using the fact that Euclidean projections onto a convex set only decrease distances. By the second part of Lemma 2, for all $q \in [2,p]$, if $\sqrt{\Phi_{j-1}} \le \frac{\rho}{p}$ (which is always satisfied as $\sqrt{\Phi_{j-1}} \le r$),

$$\mathbb{E}_{\xi_{i,j}} Z_j^q \le \left(\frac{24q\sqrt{\Phi_{j-1}}}{\rho}\right)^q.$$

By Lipschitzness of $f^{z_{i,j}}$ and Cauchy-Schwarz (on A_j), we thus have

$$\begin{aligned}\mathbb{E}_{\xi_{i,j}}|A_j|^q &\le \left(\frac{48\eta_i L q\Phi_{j-1}}{\rho}\right)^q \text{ for all } q \in [2,p], \\ \mathbb{E}_{\xi_{i,j}} B_j^q &\le \left(\frac{48\eta_i L q}{\rho}\right)^{2q}\Phi_{j-1}^q \text{ for all } q \in [1,p].\end{aligned} \tag{16}$$

Next, we perform a Taylor expansion of (15), which yields

$$\begin{aligned}\mathbb{E}_{\xi_{i,j}}\Phi_j^p \le \Phi_{j-1}^p + p\Phi_{j-1}^{p-1}\mathbb{E}_{\xi_{i,j}}\left[A_j + B_j\right] \\ +p(p-1) \\ \cdot\int_0^1 (1-t)\mathbb{E}_{\xi_{i,j}}\left[\left(\Phi_{j-1} + t(A_j+B_j)\right)^{p-2}\left(A_j+B_j\right)^2\right]\mathrm{d}t.\end{aligned} \tag{17}$$

By monotonicity of convex gradients and the first part of Lemma 1, we have

$$\mathbb{E}_{\xi_{i,j}}\left[A_j\right] \le 0. \tag{18}$$

By applying (16), we have

$$p\Phi_{j-1}^{p-1}\mathbb{E}_{\xi_{i,j}} B_j \le p\left(\frac{48\eta_i L}{\rho}\right)^2\Phi_{j-1}^p. \tag{19}$$

Next we bound the second-order terms. For any $t \in [0,1]$ we have denoting $C_j := A_j + B_j$,

$$\begin{aligned}
&\mathbb{E}_{\xi_{i,j}}\left[(\Phi_{j-1}+tC_j)^{p-2}C_j^2\right] \\
&= \sum_{q=0}^{p-2}\binom{p-2}{q}\Phi_{j-1}^{p-2-q}\mathbb{E}_{\xi_{i,j}}\left[t^{2+q}C_j^{2+q}\right] \\
&\le 4\sum_{q=0}^{p-2}2^q\binom{p-2}{q}\Phi_{j-1}^{p-2-q}\mathbb{E}_{\xi_{i,j}}\left[|A_j|^{2+q}\right] \\
&\quad +4\sum_{q=0}^{p-2}2^q\binom{p-2}{q}\Phi_{j-1}^{p-2-q}\mathbb{E}_{\xi_{i,j}}\left[B_j^{2+q}\right] \\
&\le 4\Phi_{j-1}^p\left(\frac{48\eta_i Lp}{\rho}\right)^2\sum_{q=0}^{p-2}2^q\binom{p-2}{q}\left(\frac{48\eta_i Lq}{\rho}\right)^q \\
&+4\Phi_{j-1}^p\left(\frac{48\eta_i Lp}{\rho}\right)^2\sum_{q=0}^{p-2}2^q\binom{p-2}{q}\left(\frac{48\eta_i L(2+q)}{\rho}\right)^{2q+2} \\
&\le 8\Phi_{j-1}^p\left(\frac{48\eta_i Lp}{\rho}\right)^2\left(1+\frac{96\eta_i Lp}{\rho}\right)^{p-2} \\
&\le 16\Phi_{j-1}^p\left(\frac{48\eta_i Lp}{\rho}\right)^2.
\end{aligned} \tag{20}$$

The first inequality used $(a+b)^p \le 2^p(a^p+b^p)$ for any nonnegative a, b and $0 \le t \le 1$, the second inequality used (16), and the third and fourth inequalities used

$$\frac{48\eta_i L(2+q)}{\rho} \le \frac{1}{2p}$$

for our choices of $\eta_i L \le \frac{r}{4}$ and ρ. Finally, plugging (18), (19), and (20) into (17),

$$\begin{aligned}
\mathbb{E}_{\xi_{i,j}}\Phi_j^p &\le \Phi_{j-1}^p\left(1+16p^2\left(\frac{48\eta_i Lp}{\rho}\right)^2\right) \\
&\le \Phi_{j-1}^p\left(1+16p\left(\frac{48\eta_i Lp}{\rho}\right)^2\right)^p.
\end{aligned}$$

Finally, using $(\eta_i L)^2 \le \frac{r^2}{16T} \le \frac{r^2}{16T_i}$ and our assumed bound on $\frac{r}{\rho}$, which implies $\frac{16p}{\rho^2}(48\eta_i Lp)^2 \le \frac{1}{T_i}$, taking expectations over $\{\xi_t\}_{t\in[j-1]}$ yields

$$\mathbb{E}\Phi_j^p \le \mathbb{E}\Phi_{j-1}^p\left(1+\frac{1}{T_i}\right)^p \quad \text{when } z_{i,j} \ne 1. \tag{21}$$

Potential growth: iterates with $z_{i,j} = 1$. Next, we handle the case where $z_{i,j} = 1$. We have that conditional on fixed values of $\{\xi_{i,t}\}_{t\in[j-1]}$, y_0 and y_0',

$$\begin{aligned}
\mathbb{E}_{\xi_{i,j}}\Phi_j^p &\le \mathbb{E}_{\xi_{i,j}}(\Phi_{j-1}+D_j+E_j)^p \\
&\le \mathbb{E}_{\xi_{i,j}}\left(\left(1+\frac{1}{b_i}\right)\Phi_{j-1}+2b_iE_j\right)^p,
\end{aligned} \tag{22}$$

where overloading $f \leftarrow f(\cdot; s_1)$, $h \leftarrow f(\cdot; s_1')$,

$$\begin{aligned}
D_j &:= -2\eta_i\left\langle\widetilde{\nabla}_{\bar{x}}\widehat{f}_\rho(y_{j-1}) - \widetilde{\nabla}_{\bar{x}}\widehat{h}_\rho(y'_{j-1}), y_{j-1}-y'_{j-1}\right\rangle, \\
E_j &:= \eta_i^2\left\|\widetilde{\nabla}_{\bar{x}}\widehat{f}_\rho(y_{j-1}) - \widetilde{\nabla}_{\bar{x}}\widehat{h}_\rho(y'_{j-1})\right\|^2,
\end{aligned}$$

and we use $D_j \le \frac{1}{b_i}\Phi_{j-1} + b_iE_j$ by Cauchy-Schwarz and Young's inequality. Next, convexity of $\|\cdot\|^{2q}$ implies that

$$E_j^q \le \eta_i^{2q}2^{2q-1}\left(\left\|\widetilde{\nabla}_{\bar{x}}\widehat{f}_\rho(y_{j-1})\right\|^{2q} + \left\|\widetilde{\nabla}_{\bar{x}}\widehat{h}_\rho(y'_{j-1})\right\|^{2q}\right).$$

Next, we note that since f is Lipschitz, the first part of Lemma 2 implies for all $q \le p$,

$$\begin{aligned}
\mathbb{E}\left\|\widetilde{\nabla}_{\bar{x}}\widehat{f}_\rho(y_{j-1})\right\|^{2q} &\le L^{2q}\mathbb{E}\left[\left(\frac{\gamma_\rho(y_{j-1}-\bar{x}-\xi)}{\gamma_\rho(\xi)}\right)^{2q}\right] \\
&\le 2(L)^{2q},
\end{aligned}$$

and a similar calculation holds for h. Here we used our assumed bound on $\frac{r}{\rho}$ to check the requirement in Lemma 2 is satisfied. By linearity of expectation, we thus have

$$\mathbb{E}_{\xi_{i,j}}E_j^q \le (9\eta_i L)^{2q}. \tag{23}$$

Finally, expanding (22) and plugging in the moment bound (23),

$$\begin{aligned}
\mathbb{E}_{\xi_{i,j}}\Phi_j^p &\le \sum_{q=0}^{p}\binom{p}{q}\left(1+\frac{1}{b_i}\right)^q\Phi_{j-1}^q(2b_i)^{p-q}\mathbb{E}_{\xi_{i,j}}\left[E_j^{p-q}\right] \\
&\le \sum_{q=0}^{p}\binom{p}{q}\left(1+\frac{1}{b_i}\right)^q\Phi_{j-1}^q(2b_i)^{p-q}(9\eta_i L)^{2(p-q)} \\
&= \left(\left(1+\frac{1}{b_i}\right)\Phi_{j-1}+2b_i(9\eta_i L)^2\right)^p.
\end{aligned}$$

Taking expectations over $\{\xi_{i,t}\}_{t\in[j-1]}$, and using Fact 4 with $Z \leftarrow (1+\frac{1}{b_i})\Phi_{j-1}$ and $C \leftarrow 2b_i(9\eta_i L)^2$, when $z_{i,j} = 1$,

$$\mathbb{E}\Phi_j^p \le \left(\left(1+\frac{1}{b_i}\right)\mathbb{E}\left[\Phi_{j-1}^p\right]^{\frac{1}{p}} + 2b_i(9\eta_i L)^2\right)^p. \tag{24}$$

One loop privacy. We begin by obtaining a high-probability bound on Φ_{T_i}. Define

$$W_j := \mathbb{E}[\Phi_j^p]^{\frac{1}{p}}.$$

By using (21) and (24), we observe

$$W_j \le \begin{cases}\left(1+\frac{1}{T_i}\right)W_{j-1} & z_{i,j} \ne 1 \\ \left(1+\frac{1}{b_i}\right)W_{j-1} + 2b_i(9\eta_i L)^2 & z_{i,j} = 1\end{cases}.$$

Hence, regardless of the b_i locations of the 1 indices in $\mathcal{I}_i$, we have

$$W_{T_i} \le \left(1+\frac{1}{T_i}\right)^{T_i}\left(1+\frac{1}{b_i}\right)^{b_i}\left(2b_i^2(9\eta_i L)^2\right) \le 1200b_i^2(\eta_i L)^2.$$

Thus, by Markov's inequality, with probability at least $1-\frac{\delta}{\log T}$ over the randomness of $\Xi_i = \{\xi_{i,j}\}_{j\in[T_i]}$, we have using our choice of p,

$$\left\|y_{T_i} - y'_{T_i}\right\|^2 \le 1200b_i^2(\eta_i L)^2 \cdot \left(\frac{\log T}{\delta}\right)^{\frac{1}{p}} \le 1500b_i^2(\eta_i L)^2. \tag{25}$$

In the last inequality, we used our choice of p. Call $\mathcal{E}_i$ the event that the sampled Ξ_i admits a deterministic map which yields the bound in (25). By the second part of Proposition 5,

the conditional distribution of the output of the i^{th} outer loop under $\mathcal{E}_i$ satisfies $(\alpha, 1500\beta^2 b_i^2)$-RDP, where we use the value of σ_i in Line 4 of Algorithm 3. We conclude via Fact 1 with $\mathcal{E} \leftarrow \mathcal{E}_i$ that the i^{th} outer loop of Algorithm 3 satisfies

$$\left(\alpha, 1500\alpha\beta^2 b_i^2, \frac{\delta}{\log T}\right)\text{-RDP.}$$

All loops privacy. By applying composition of RDP (the third part of Proposition 5), for a given realization of $\mathcal{I} = \cup_{i\in[k]}\mathcal{I}_i$ with b occurrences of 1, applying composition over the $\log T$ outer iterations (Lemma 3), Algorithm 3 satisfies

$$\left(\alpha, 1500\alpha\beta^2 b^2, \delta\right)\text{-RDP.}$$

Here, we used $\sum_{i\in[k]} b_i^2 \le b^2$. This is the desired conclusion. □

We next apply amplification by subsampling to boost the guarantee of Lemma 5. To do so, we use the following key Proposition 7, which was proven in [BDRS18]. The use case in [BDRS18] involved subsampling with replacement and was used in a framework they introduced termed truncated CDP, but we will not need the framework except through the following powerful fact.

Proposition 7 (Theorem 12, [BDRS18])**.** *Let $\tau \le \frac{1}{3}$, $s \in (0, \frac{1}{40})$. Let P, Q, R be three distributions over the same probability space, such that for each pair $P_1, P_2 \in \{P, Q, R\}$, we have $D_\alpha(P_1\|P_2) \le \alpha\tau$ for all $\alpha > 1$. Then for all $\alpha \in (1, \frac{3}{\tau})$,*

$$D_\alpha(sP + (1-s)R\|sQ + (1-s)R) \le 13s^2\alpha\tau.$$

We also require a straightforward technical fact about binomial distributions.

Lemma 6. *Let $m, n \in \mathbb{N}$ satisfy $\frac{m}{n} \le \frac{1}{60}$. Consider the following partition of the elements $\mathcal{I} \in [n]^m$ with at most b copies of 1:*

$$S_0 := \{\mathcal{I} \in [n]^m \mid \mathcal{I}_i \ne 1 \text{ for all } i \in [m]\},$$
$$S_1 := \{\mathcal{I} \in [n]^m \mid \mathcal{I}_i = 1 \text{ for } k \text{ many } i \in [m]\},\ k \in [1, b].$$

Let π_0 and π_1 be the uniform distributions on S_0 and S_1 respectively. Then there exists a coupling $\Gamma(\pi_0, \pi_1)$ such that for all $(\mathcal{I}, \mathcal{I}')$ in the support of Γ,

$$|\{i \mid \mathcal{I}_i \ne \mathcal{I}'_i\}| \le b.$$

Proof. Define a probability distribution p on elements of $[b]$ such that

$$p_a := \frac{\binom{m}{a}(n-1)^{m-a}}{\sum_{a\in[b]}\binom{m}{a}(n-1)^{m-a}} \text{ for all } a \in [b].$$

Clearly, $\sum_{a\in[b]} p_a = 1$. Our coupling $\Gamma := \Gamma(\pi_0, \pi_1)$ is defined as follows.

1) Draw $\mathcal{I} \sim \pi_0$ and $a \sim p$ independently.
2) Let $\mathcal{I}'$ be $\mathcal{I}$ with a uniformly random subset of a indices replaced with 1. Return $(\mathcal{I}, \mathcal{I}')$.

This coupling satisfies the requirement, so it suffices to verify it has the correct marginals. This is immediate for S_0 by definition. For $\mathcal{I}' \in S_1$, suppose $\mathcal{I}'$ has a occurrences of the index 1. The total probability $\mathcal{I}'$ is drawn from Γ is then indeed

$$\frac{(n-1)^a}{(n-1)^m} \cdot \frac{p_a}{\binom{m}{a}} = \frac{1}{\sum_{a\in[b]}\binom{m}{a}(n-1)^{m-a}} = \frac{1}{|S_1|}.$$

The first equality follows as the probability we draw $\mathcal{I} \sim \pi_0$ which agrees with $\mathcal{I}'$ on all the non-1 locations is $(n-1)^{a-m}$, and the probability $\mathcal{I}'$ is drawn given that we selected $\mathcal{I}$ is $p_a \cdot \binom{m}{a}^{-1}$. □

Finally, we are ready to state our main privacy guarantee for Algorithm 3.

Lemma 7. *There is a universal constant $C_{\text{priv}} \in [1, \infty)$, such that if $\frac{T}{n} \le \frac{1}{C_{\text{priv}}}$, $\beta^2 \log^2(\frac{1}{\delta}) \le \frac{1}{C_{\text{priv}}}$, $\delta \in (0, \frac{1}{6})$, and $\frac{\rho}{r} \ge C_{\text{priv}} \log^2(\frac{\log T}{\delta})$, Algorithm 3 satisfies $(\alpha, \alpha\tau, \delta)$-RDP for*

$$\tau := C_{\text{priv}}\left(\beta \log\left(\frac{1}{\delta}\right) \cdot \frac{T}{n}\right)^2,\ \alpha \in \left(1, \frac{1}{C_{\text{priv}}\beta^2 \log^2(\frac{1}{\delta})}\right).$$

Proof. Let $\mathcal{D}$, $\mathcal{D}'$ be neighboring, and without loss of generality, suppose they differ in the first entry. Let $C_{\text{priv}} \ge 60$, and let $\mathcal{I}$ be defined as in (14). Let $\mathcal{E}$ be the event that $\mathcal{I}$ contains at most b copies of the index 1, where

$$b := 2\log\left(\frac{2}{\delta}\right).$$

By a Chernoff bound, $\mathcal{E}$ occurs with probability at least $1 - \frac{\delta}{2}$ over the randomness of $\mathcal{I}$. We define P to be the distribution of the output of Algorithm 3 when run on $\mathcal{D}$, conditioned on $\mathcal{E}$ and $\mathcal{I}$ containing at least one copy of the index 1 (call this total conditioning event $\mathcal{E}_1$, i.e., there are between 1 and b copies of the index 1). Similarly, we define Q to be the distribution when run on $\mathcal{D}'$ conditioned on $\mathcal{E}_1$, and R to be the distribution conditioned on $\mathcal{E} \cap \mathcal{E}_1^c$ (when run on either $\mathcal{D}$ or $\mathcal{D}'$). We claim that for all $P_1, P_2 \in \{P, Q, R\}$, we have

$$D_{\alpha, \frac{\delta}{2}}(P_1\|P_2) \le 1500\alpha\beta^2 b^2, \text{ for all } \alpha > 1. \tag{26}$$

To see (26) for $P_1 = P$ and $P_2 = Q$ (or vice versa), we can view P, Q as mixtures of outcomes conditioned on the realization $\mathcal{I}$. Then, applying quasiconvexity of Rènyi divergence (over this mixture), and applying Lemma 5 (with $\delta \leftarrow \frac{\delta}{2}$), we have the desired claim. To see (26) for the remaining cases, we first couple the conditional distributions under $\mathcal{E}_1$ and $\mathcal{E} \cap \mathcal{E}_1^c$ by their index sets, according to the coupling in Lemma 6. Then applying quasiconvexity of Rényi divergence (over this coupling) again yields the claim, where we set $m \leftarrow \widehat{T} - 1 \le T$. Finally, let

$$s := \Pr[\mathcal{E}_1 \mid \mathcal{E}] = 1 - \frac{\left(1 - \frac{1}{n}\right)^{\widehat{T}-1}}{\Pr[\mathcal{E}]}$$
$$\le 1 - \frac{1 - \frac{1.1T}{n}}{1 - \frac{\delta}{2}} \le \frac{1.2T}{n}.$$

Note that conditional on $\mathcal{E}$ and the failure event in Lemma 5 not occurring, the distributions of Algorithm 3 using $\mathcal{D}$ and $\mathcal{D}'$ respectively are $sP + (1-s)R$ and $sQ + (1-s)R$. Hence,

union bounding with $\mathcal{E}^c$ (see Fact 1), the claim follows from Proposition 7 with $\tau \leftarrow 6000\beta^2 \log^2(\frac{2}{\delta})$. □

c) Regularized extension.: We give a slight extension to Algorithm 3 which handles regularization, and enjoys similar utility and privacy guarantees as stated in Proposition 6. Let

$$x^\star_{\bar{x},\lambda} := \text{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)} \left\{ \widehat{f^{\text{erm}}_\rho}(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2 \right\}. \quad (27)$$

Our extension Algorithm 4 is identical to Algorithm 3, except it requires a regularization parameter λ, allows for an arbitrary starting point with an expected distance bound (adjusting the step size accordingly), and takes composite projected steps incorporating the regularization.

Algorithm 4: Subsampled ReSQued ERM solver, regularized case, convex rate

Input: $\bar{x} \in \mathbb{R}^d$, ball radius, convolution radius, privacy parameter, and regularization parameter $r, \rho, \beta, \lambda > 0$, dataset $\mathcal{D} \in \mathcal{S}^n$, iteration count $T \in \mathbb{N}$, distance bound $r' \in [0, 2r]$, initial point $x_0 \in \mathbb{B}_{\bar{x}}(r)$ satisfying $\mathbb{E}\|x_0 - x^\star_{\bar{x},\lambda}\|^2 \le (r')^2$
$\widehat{T} \leftarrow 2^{\lfloor \log_2 T \rfloor}$, $k \leftarrow \log_2 \widehat{T}$, $\eta \leftarrow \frac{r'}{L}\min(\frac{1}{\sqrt{T}}, \frac{\beta}{\sqrt{d}})$
for $i \in [k]$ **do**
  $T_i \leftarrow 2^{-i}\widehat{T}$, $\eta_i \leftarrow 4^{-i}\eta$, $\sigma_i \leftarrow \frac{L\eta_i}{\beta}$
  $y_0 \leftarrow x_{i-1}$
  for $j \in [T_i]$ **do**
    $z_{i,j} \sim_{\text{unif.}} [n]$
    $y_j \leftarrow \text{argmin}_{y\in\mathbb{B}_{\bar{x}}(r)}\{\langle \eta_i \widetilde{\nabla}_{\bar{x}} \widehat{f}^{z_{i,j}}_\rho(y_{j-1}), y\rangle + \frac{1}{2}\|y - y_{j-1}\|^2 + \frac{\eta_i\lambda}{2}\|y-\bar{x}\|^2\}$
  end
  $\bar{y}_i \leftarrow \frac{1}{T_i}\sum_{j\in[T_i]} y_j$
  $x_i \leftarrow \bar{y}_i + \zeta_i$, for $\zeta_i \sim \mathcal{N}(0, \sigma_i^2 \mathbf{I}_d)$
end
return x_k

Corollary 2. *Let $x^\star_{\bar{x},\lambda}$ be defined as in (27). Algorithm 4 uses at most T gradients and produces $x \in \mathbb{B}_{\bar{x}}(r)$ such that, for a universal constant C_{cvx},*

$$\mathbb{E}\left[\widehat{f^{\text{erm}}_\rho}(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2\right] - \left(\widehat{f^{\text{erm}}_\rho}(x^\star_{\bar{x},\lambda}) + \frac{\lambda}{2}\|x^\star_{\bar{x},\lambda}-\bar{x}\|^2\right) \le C_{\text{cvx}} L r' \left(\frac{\sqrt{d}}{\beta T} + \frac{1}{\sqrt{T}}\right).$$

Moreover, there is a universal constant $C_{\text{priv}} \ge 1$, such that if $\frac{T}{n} \le \frac{1}{C_{\text{priv}}}$, $\beta^2 \log^2(\frac{1}{\delta}) \le \frac{1}{C_{\text{priv}}}$, $\delta \in (0, \frac{1}{6})$, and $\frac{\rho}{r} \ge C_{\text{priv}} \log^2(\frac{\log T}{\delta})$, Algorithm 4 satisfies $(\alpha, \alpha\tau, \delta)$-RDP for

$$\tau := C_{\text{priv}}\left(\beta \log\left(\frac{1}{\delta}\right)\cdot\frac{T}{n}\right)^2, \quad \alpha \in \left(1, \frac{1}{C_{\text{priv}}\beta^2\log^2(\frac{1}{\delta})}\right).$$

Proof. The proof is almost identical to Proposition 6, so we only discuss the differences. Throughout this proof, for notational convenience, we define

$$F^\lambda(x) := \widehat{f^{\text{erm}}_\rho}(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2.$$

Utility. Standard results on composite stochastic mirror descent (e.g. Lemma 12 of [CJST19]) show the utility bound in (13) still holds with F^λ in place of F. In particular each term $\mathbb{E}[F^\lambda(\bar{y}_i) - F^\lambda(\bar{y}_{i-1})]$ as well as $\mathbb{E}[F^\lambda(x_k) - F^\lambda(\bar{y}_k)]$ enjoys the same bound as its counterpart in (13). The only other difference is that, defining $\zeta_0 := x_0 - x^\star_{\bar{x},\lambda}$ in the proof of Lemma 4, we have $\mathbb{E}\zeta_0^2 \le (r')^2$ in place of the bound r^2, and we appropriately changed η to scale as r' instead.

Privacy. The subsampling-based reduction from Lemma 7 to Lemma 5 is identical, so we only discuss how to obtain an analog of Lemma 5 for Algorithm 4. In each iteration $j \in [T_i]$, by completing the square, we can rewrite Line 8 as

$$y_j \leftarrow \text{argmin}_{y\in\mathbb{B}_{\bar{x}}(r)}\left\{\frac{1}{2}\|y-v\|^2\right\},$$
$$v = \frac{1}{1+\eta_i\lambda}y_{j-1} + \frac{\eta_i\lambda}{1+\eta_i\lambda}\bar{x} - \frac{\eta_i}{1+\eta_i\lambda}\widetilde{\nabla}_{\bar{x}}\widehat{f}^{z_{i,j}}_\rho(y_{j-1}).$$

Now consider our (conditional) bounds on $\mathbb{E}_{\xi_{i,j}}\Phi_j$ in (15) and (22). We claim these still hold true; before projection, the same arguments used in (15) and (22) still hold (in fact improve by $(1+\eta_i\lambda)^2$), and projection only decreases distances. Finally, note that the proof of Lemma 5 only used the choice of step size η through $\eta L\sqrt{T} \le r$ and used the assumed bound on $\frac{r}{\rho}$ to bound the drift growth. As we now have $\eta L\sqrt{T} \le r' \le 2r$, we adjusted the assumed bound on $\frac{r}{\rho}$ by a factor of 2. The remainder of the proof of Lemma 5 is identical. □

Without loss of generality, C_{priv} is the same constant in Proposition 6 and Corollary 2, since we can set both to be the maximum of the two. The same logic applies to the following Proposition 8 and Lemma 10 (which will also be parameterized by a C_{priv}) so we will not repeat it. Finally, the following fact about initial error will also be helpful in the following Section IV-C.

Lemma 8. *We have*

$$\widehat{f^{\text{erm}}_\rho}(\bar{x}) - \left(\widehat{f^{\text{erm}}_\rho}(x^\star_{\bar{x},\lambda}) + \frac{\lambda}{2}\|x^\star_{\bar{x},\lambda}-\bar{x}\|^2\right) \le \frac{2L^2}{\lambda}.$$

Proof. By strong convexity and Lipschitzness of $\widehat{f^{\text{erm}}_\rho}$, we have

$$\frac{\lambda}{2}\|x^\star_{\bar{x},\lambda}-\bar{x}\|^2 \le \widehat{f^{\text{erm}}_\rho}(\bar{x}) - \left(\widehat{f^{\text{erm}}_\rho}(x^\star_{\bar{x},\lambda}) + \frac{\lambda}{2}\|x^\star_{\bar{x},\lambda}-\bar{x}\|^2\right) \le \widehat{f^{\text{erm}}_\rho}(\bar{x}) - \widehat{f^{\text{erm}}_\rho}(x^\star_{\bar{x},\lambda}) \le L\|x^\star_{\bar{x},\lambda}-\bar{x}\|.$$

Rearranging gives $\|x^\star_{\bar{x},\lambda}-\bar{x}\| \le \frac{2L}{\lambda}$, which can be plugged in above to yield the conclusion. □

We also state a slight extension to Lemma 8 which will be used in Section IV-E.

Lemma 9. *Define* $x^\star_{\bar{x},x',\lambda} := \operatorname{argmin}_{x\in\mathbb{B}_{\bar{x}}(r)}\{\widehat{f^{\mathrm{erm}}_\rho}(x) + \frac{\lambda}{2}\|x-x'\|^2\}$, *where* $x' \in \mathbb{R}^d$ *is not necessarily in* $\mathbb{B}_{\bar{x}}(r)$. *Let* $x_0 := \Pi_{\mathbb{B}_{\bar{x}}(r)}(x')$. *We have*

$$\left(\widehat{f^{\mathrm{erm}}_\rho}(x_0) + \frac{\lambda}{2}\|x_0 - x'\|^2\right) - \left(\widehat{f^{\mathrm{erm}}_\rho}(x^\star_{\bar{x},x',\lambda}) + \frac{\lambda}{2}\left\|x^\star_{\bar{x},x',\lambda} - x'\right\|^2\right) \le \frac{2L^2}{\lambda}.$$

Proof. The proof is identical to Lemma 8, where we use $\frac{\lambda}{2}\|x_0 - x'\|^2 \le \frac{\lambda}{2}\|x^\star_{\bar{x},x',\lambda} - x'\|^2$. □

C. Subsampled smoothed ERM solver: the strongly convex case

We next give an ERM algorithm similar to Algorithm 4, but enjoys an improved optimization rate. In particular, it again attains RDP bounds improving with the subsampling parameter $\frac{T}{n}$, and we obtain error guarantees against $x^\star_{\bar{x},\lambda}$ defined in (27) at a rate decaying as $\frac{1}{T}$ or better.

Algorithm 5: Subsampled ReSQued ERM solver, strongly convex case

Input: $\bar{x} \in \mathbb{R}^d$, ball radius, convolution radius, privacy parameter, and regularization parameter $r, \rho, \beta, \lambda > 0$, dataset $\mathcal{D} \in \mathcal{S}^n$, iteration count $T \in \mathbb{N}$
$k \leftarrow \lceil \log\log T\rceil, x_0 \leftarrow \bar{x}$
for $i \in [k]$ **do**
  $\beta_{i-1} \leftarrow 2^{\frac{k-i+1}{2}}\beta$, $r_{i-1} \leftarrow \min(2r, \sqrt{\frac{2D_{i-1}}{\lambda}})$ (see (28)), $T_{i-1} \leftarrow 2^{i-1-k}T$
  $x_i \leftarrow$ output of Algorithm 4 with inputs $(\bar{x}, r, \rho, \beta_{i-1}, \lambda, \mathcal{D}, T_{i-1}, r_{i-1}, x_{i-1})$
end
return x_{k+1}

We now give our analysis of Algorithm 5 below. The proof follows a standard reduction template from the strongly convex case to the convex case (see e.g. Lemma 4.7 in [KLL21]).

Proposition 8. *Let* $x^\star_{\bar{x},\lambda}$ *be defined as in* (27). *Algorithm 5 uses at most* T *gradients and produces* x *such that, for a universal constant* C_{sc},

$$\mathbb{E}\left[\widehat{f^{\mathrm{erm}}_\rho}(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2\right] - \widehat{f^{\mathrm{erm}}_\rho}(x^\star_{\bar{x},\lambda}) - \frac{\lambda}{2}\|x^\star_{\bar{x},\lambda} - \bar{x}\|^2 \le \frac{C_{\mathrm{sc}}L^2}{\lambda}\left(\frac{d}{\beta^2T^2} + \frac{1}{T}\right).$$

Moreover, there is a universal constant $C_{\mathrm{priv}} \ge 1$, *such that if* $\frac{T}{n} \le \frac{1}{C_{\mathrm{priv}}}$, $\beta^2\log^2(\frac{\log\log T}{\delta}) \le \frac{1}{C_{\mathrm{priv}}}$, $\delta \in (0, \frac{1}{6})$, *and* $\frac{\rho}{r} \ge C_{\mathrm{priv}}\log^2(\frac{\log T}{\delta})$, *Algorithm 5 satisfies* $(\alpha, \alpha\tau, \delta)$*-RDP for*

$$\tau := C_{\mathrm{priv}}\left(\beta\log\left(\frac{\log\log T}{\delta}\right)\cdot\frac{T}{n}\right)^2, \qquad \alpha \in \left(1, \frac{1}{C_{\mathrm{priv}}\beta^2\log^2(\frac{\log\log T}{\delta})}\right).$$

Proof. We analyze the utility and privacy separately.

Utility. Denote for simplicity $F^\lambda(x) := \widehat{f^{\mathrm{erm}}_\rho}(x) + \frac{\lambda}{2}\|x-\bar{x}\|^2$, $F^\lambda_\star := F^\lambda(x^\star_{\bar{x},\lambda})$, and $\Delta_i := \mathbb{E}[F^\lambda(x_i) - F^\lambda_\star]$. Moreover, define for all $0 \le i \le k$,

$$E_i := \frac{2C^2_{\mathrm{cvx}}L^2}{\lambda}\cdot\left(\frac{\sqrt{d}}{\beta_iT_i} + \frac{1}{\sqrt{T_i}}\right)^2, \qquad D_i := 4E_i\sqrt[2^i]{\frac{2L^2}{\lambda}\cdot\frac{1}{4E_0}}, \tag{28}$$

where we define $T_k = T$ and $\beta_k = \beta$. By construction, for all $0 \le i \le k-1$, $E_{i+1} = \frac{1}{2}E_i$, and so

$$\frac{D_{i+1}}{4E_{i+1}} = \sqrt{\frac{D_i}{4E_i}} \implies \sqrt{D_iE_i} = D_{i+1}. \tag{29}$$

We claim inductively that for all $0 \le i \le k$, $\Delta_i \le D_i$. The base case of the induction follows because by Lemma 8, we have $\Delta_0 \le \frac{2L^2}{\lambda} = D_0$. Next, suppose that the inductive hypothesis is true up to iteration i. By strong convexity,

$$\mathbb{E}\left[\left\|x_i - x^\star_{\bar{x},\lambda}\right\|^2\right] \le \frac{2\Delta_i}{\lambda} \le \frac{2D_i}{\lambda},$$

where we used the inductive hypothesis. Hence, the expected radius upper bound (defined by r_i) is valid for the call to Algorithm 4. Thus, by Corollary 2,

$$\begin{aligned}\Delta_{i+1} &= \mathbb{E}\left[F^\lambda(x_{i+1}) - F^\lambda_\star\right] \\ &\le C_{\mathrm{cvx}}Lr_i\left(\frac{\sqrt{d}}{\beta_iT_i} + \frac{1}{\sqrt{T_i}}\right) \\ &\le C_{\mathrm{cvx}}L\sqrt{\frac{2D_i}{\lambda}}\left(\frac{\sqrt{d}}{\beta_iT_i} + \frac{1}{\sqrt{T_i}}\right) = \sqrt{D_iE_i} = D_{i+1}.\end{aligned}$$

Here we used (29) in the last equation, which completes the induction. Hence, iterating (29) for $k = \lceil\log_2\log_2 T\rceil$ iterations, where we use $E_0 \ge \frac{L^2}{2\lambda T}$ so that $D_k \le 8E_k$, we have

$$\Delta_k \le 8E_k \le \frac{32C^2_{\mathrm{cvx}}L^2}{\lambda}\cdot\left(\frac{d}{\beta^2T^2} + \frac{1}{T}\right).$$

Privacy. The privacy guarantee follows by combining the privacy guarantee in Corollary 2 and composition of approximate RDP (Lemma 3), where we adjusted the definition of δ by a factor of k. In particular, we use that the privacy guarantee in each call to Corollary 2 is a geometric sequence (i.e., $\beta_i^2T_i^2$ is doubling), and at the end it is $\frac{1}{2}\beta^2T^2$. □

D. Private stochastic proximal estimator

In this section, following the development of [ACJ+21], we give an algorithm which calls Algorithm 5 with several different iteration counts and returns a (random) point $\widehat{x}$ which enjoys a substantially reduced bias for $x^\star_{\bar{x},\lambda}$ defined in (27) compared to the expected number of gradient queries.

Algorithm 6: Bias-reduced ReSQued stochastic proximal estimator

Input: $\bar{x} \in \mathbb{R}^d$, ball radius, convolution radius, privacy parameter, and regularization parameter $r, \rho, \beta, \lambda > 0$, dataset $\mathcal{D} \in \mathcal{S}^n$, iteration count $T \in \mathbb{N}$ with $T \le \lfloor \frac{n}{2C_{\text{priv}}} \rfloor$
$T_{\max} \leftarrow \lfloor \frac{n}{C_{\text{priv}}} \rfloor, j_{\max} \leftarrow \lfloor \log_2 \frac{T_{\max}}{T} \rfloor$
for $k \in [j_{\max}]$ **do**
  Draw $J \sim \text{Geom}(\frac{1}{2})$
  $x_0 \leftarrow$ output of Algorithm 5 with inputs $(\bar{x}, r, \rho, \beta, \lambda, \mathcal{D}, T)$
  if $J \le j_{\max}$ **then**
    $x_J \leftarrow$ output of Algorithm 5 with inputs $(\bar{x}, r, \rho, 2^{-\frac{J}{2}}\beta, \lambda, \mathcal{D}, 2^J T)$
    $x_{J-1} \leftarrow$ output of Algorithm 5 with inputs $(\bar{x}, r, \rho, 2^{-\frac{J-1}{2}}\beta, \lambda, \mathcal{D}, 2^{J-1} T)$
    $\widehat{x}_k \leftarrow x_0 + 2^J(x_J - x_{J-1})$
  end
  else
    $\widehat{x}_k \leftarrow x_0$
  end
end
Return: $\widehat{x} \leftarrow \frac{1}{j_{\max}} \sum_{k \in [j_{\max}]} \widehat{x}_k$

Proposition 9. *Let $x^\star_{\bar{x},\lambda}$ be defined as in (27). We have, for a universal constant C_{bias}:*

$$\|\mathbb{E}\widehat{x} - x^\star_{\bar{x},\lambda}\| \le C_{\text{bias}} \left(\frac{L}{\lambda} \cdot \left(\frac{\sqrt{d}}{\beta n} + \frac{1}{\sqrt{n}} \right) \right),$$

and, for a universal constant C_{var},

$$\mathbb{E}\|\widehat{x} - x^\star_{\bar{x},\lambda}\|^2 \le \frac{C_{\text{var}} L^2}{\lambda^2} \left(\frac{d}{\beta^2 T^2} + \frac{1}{T} \right).$$

Proof. We begin by analyzing the output $\widehat{x}_k$ of a single loop $k \in [j_{\max}]$. For $J \sim \text{Geom}(\frac{1}{2})$, we have $\Pr[J = j] = 2^{-j}$ if $j \in [j_{\max}]$, and $\Pr[J = j] = 0$ otherwise. We denote x_j to be the output of Algorithm 3 with privacy parameter $2^{-\frac{j}{2}}\beta$ and gradient bound $2^j T$. First,

$$\mathbb{E}\widehat{x}_k = \mathbb{E}x_0 + \sum_{j \in [j_{\max}]} \Pr[J = j] 2^j (\mathbb{E}x_j - \mathbb{E}x_{j-1}) = \mathbb{E}x_{j_{\max}}.$$

Since $T \cdot 2^{j_{\max}} \ge \frac{T_{\max}}{2} \ge \frac{n}{2C_{\text{priv}}}$, applying Jensen's inequality gives

$$\begin{aligned} \|\mathbb{E}x_{j_{\max}} - x^\star_{\bar{x},\lambda}\| &\le \sqrt{\mathbb{E}\|x_{j_{\max}} - x^\star_{\bar{x},\lambda}\|^2} \\ &\le \frac{\sqrt{2C_{\text{sc}}} L}{\lambda} \left(\frac{\sqrt{d}}{\beta n} + \frac{1}{\sqrt{n}} \right), \end{aligned}$$

where the last inequality follows from Proposition 8 and strong convexity of the regularized function to convert the function error bound to a distance bound. This implies the first conclusion, our bias bound. Furthermore, for our variance bound, we have

$$\begin{aligned} \mathbb{E}\|\widehat{x}_k - \mathbb{E}\widehat{x}_k\|^2 &\le \mathbb{E}\|\widehat{x}_k - x^\star_{\bar{x},\lambda}\|^2 \\ &\le 2\mathbb{E}\|\widehat{x}_k - x_0\|^2 + 2\mathbb{E}\|x_0 - x^\star_{\bar{x},\lambda}\|^2. \end{aligned}$$

By Proposition 8 and strong convexity, $\mathbb{E}\|x_0 - x^\star_{\bar{x},\lambda}\|^2 \le \frac{C_{\text{sc}} L^2}{\lambda^2}(\frac{d}{\beta^2 T^2} + \frac{1}{T})$. Next,

$$\begin{aligned} \mathbb{E}\|\widehat{x}_k - x_0\|^2 &= \sum_{j \in [j_{\max}]} \Pr[J = j] 2^{2j} \mathbb{E}\|x_j - x_{j-1}\|^2 \\ &= \sum_{j \in [j_{\max}]} 2^j \mathbb{E}\|x_j - x_{j-1}\|^2. \end{aligned}$$

Note that

$$\begin{aligned} \mathbb{E}\|x_j - x_{j-1}\|^2 &\le 2\mathbb{E}\|x_j - x^\star_{\bar{x},\lambda}\|^2 + 2\mathbb{E}\|x_{j-1} - x^\star_{\bar{x},\lambda}\|^2 \\ &\le 2^{-j} \cdot \frac{6C_{\text{sc}} L^2}{\lambda^2} \left(\frac{d}{\beta^2 T^2} + \frac{1}{T} \right), \end{aligned}$$

and hence combining the above bounds yields

$$\mathbb{E}\left\|\widehat{x}_k - \mathbb{E}\widehat{x}_k\right\|^2 \le \frac{14 C_{\text{sc}} j_{\max} L^2}{\lambda^2} \cdot \left(\frac{d}{\beta^2 T^2} + \frac{1}{T} \right).$$

Now, averaging $j_{\max}$ independent copies shows that

$$\begin{aligned} \mathbb{E}\left\|\widehat{x} - x^\star_{\bar{x},\lambda}\right\|^2 &= \left\|\widehat{x} - \mathbb{E}\widehat{x}\right\|^2 + \left\|\mathbb{E}\widehat{x} - x^\star_{\bar{x},\lambda}\right\|^2 \\ &\le \frac{1}{j_{\max}} \cdot \left(\frac{14 C_{\text{sc}} j_{\max} L^2}{\lambda^2} \cdot \left(\frac{d}{\beta^2 T^2} + \frac{1}{T} \right) \right) \\ &\quad + C^2_{\text{bias}} \left(\frac{L}{\lambda} \cdot \left(\frac{\sqrt{d}}{\beta n} + \frac{1}{\sqrt{n}} \right) \right)^2, \end{aligned}$$

where we used our earlier bias bound. The conclusion follows by letting $C_{\text{var}} = C^2_{\text{bias}} + 14 C_{\text{sc}}$. □

We conclude with a gradient complexity and privacy bound, depending on the sampled J.

Lemma 10. *There is a universal constant $C_{\text{priv}} \ge 1$, such that if $\beta^2 \log^2(\frac{\log\log n}{\delta}) \le \frac{1}{C_{\text{priv}}}$, $\delta \in (0, \frac{1}{2})$, and $\frac{\rho}{r} \ge C_{\text{priv}} \log^2(\frac{\log T}{\delta})$, the following holds. Consider one loop indexed by $k \in [j_{\max}]$, and let J be the result of the $\text{Geom}(\frac{1}{2})$ draw. If $J \in [j_{\max}]$, loop k of Algorithm 6 uses at most $2^{J+1}T$ gradients. Furthermore, the loop satisfies $(\alpha, \alpha\tau, \delta)$-RDP for*

$$\tau := 2^J \cdot C_{\text{priv}} \left(\beta \log \left(\frac{\log\log n}{\delta} \right) \cdot \frac{T}{n} \right)^2,$$

$$\alpha \in \left(1, \frac{1}{C_{\text{priv}} \beta^2 \log^2 \left(\frac{\log\log n}{\delta} \right)} \right).$$

If $J \notin [j_{\max}]$, Algorithm 6 uses at most T gradients, and the loop satisfies $(\alpha, \alpha\tau, \delta)$-RDP for

$$\tau := C_{\text{priv}} \left(\beta \log \left(\frac{\log\log n}{\delta} \right) \cdot \frac{T}{n} \right)^2,$$

$$\alpha \in \left(1, \frac{1}{C_{\text{priv}} \beta^2 \log^2 \left(\frac{\log\log n}{\delta} \right)} \right).$$

Proof. This is immediate by Proposition 8, where we applied Lemma 3 and set $\delta \leftarrow \frac{\delta}{3}$ (taking a union bound over the at most 3 calls to Algorithm 5, adjusting C_{priv} as necessary). □

E. Private ERM solver

In this section, we give our main result on privately solving ERM in the setting of Problem 2, which will be used in a reduction framework in Section IV-F to solve the SCO problem as well. Our ERM algorithm is an instantiation of Proposition 1. We first develop a line search oracle (see Definition 3) based on the solver of Section IV-C (Algorithm 5), which succeeds with high probability. To do so, we leverage the following geometric lemma for aggregating independent runs of our solver.

Lemma 11 (Claim 1, [KLL+22]). *There is an algorithm* Aggregate *which takes as input $(S, \Delta) \in (\mathbb{R}^d)^k \times \mathbb{R}_{\geq 0}$, and returns $z \in \mathbb{R}^d$ such that $\|z - y\| \leq \Delta$, if for some unknown point $y \in \mathbb{R}^d$ satisfying at least $0.51k$ points $x \in S$, $\|x - y\| \leq \frac{\Delta}{3}$. The algorithm runs in time $O(dk^2)$.*

Algorithm 7: High probability ReSQued ERM solver, strongly convex case

Input: $\bar{x} \in \mathbb{R}^d$, ball radius, convolution radius, privacy parameter, regularization parameter, and failure probability $r, \rho, \beta, \lambda, \zeta > 0$, dataset $\mathcal{D} \in \mathcal{S}^n$, iteration count $T \in \mathbb{N}$
$k \leftarrow 20 \log(\frac{1}{\zeta})$
for $i \in [k]$ **do**
  $x_i \leftarrow$ output of Algorithm 5 with inputs $(\bar{x}, r, \rho, \beta, \lambda, \mathcal{D}, T)$
end
Return:
$x' \leftarrow \mathsf{Aggregate}(\{x_i\}_{i\in[k]}, \frac{9\sqrt{2C_{\text{sc}}}L}{\lambda}(\frac{d}{\beta^2T^2} + \frac{1}{T})^{\frac{1}{2}})$

Proposition 10. *Let $x^\star_{\bar{x},\lambda}$ be defined as in (27). Algorithm 7 uses at most $18T\log(\frac{1}{\zeta})$ gradients and produces x' such that with probability at least $1 - \zeta$, for a universal constant C_{ls},*

$$\|x' - x^\star_{\bar{x},\lambda}\| \leq \frac{C_{\text{ls}}L}{\lambda} \cdot \left(\frac{\sqrt{d}}{\beta T} + \frac{1}{\sqrt{T}}\right).$$

Moreover, there exists a universal constant $C_{\text{priv}} \geq 1$ such that $\frac{T}{n} \leq \frac{1}{C_{\text{priv}}}, \delta \in (0, \frac{1}{6})$ and $\frac{\rho}{r} \geq C_{\text{priv}} \log^2(\frac{1}{\delta}\log(\frac{T}{\zeta}))$, Algorithm 7 satisfies $(\alpha, \alpha\tau, \delta)$-RDP for

$$\tau := C_{\text{priv}} \log\left(\frac{1}{\zeta}\right)\left(\beta \log\left(\frac{1}{\delta}\log\left(\frac{T}{\zeta}\right)\right) \cdot \frac{T}{n}\right)^2,$$
$$\alpha \in \left(1, \frac{1}{C_{\text{priv}}\beta^2 \log^2\left(\frac{1}{\delta}\log\left(\frac{T}{\zeta}\right)\right)}\right).$$

Proof. For each x_i, by Proposition 8,

$$\mathbb{E}\left[\widehat{f^{\text{erm}}}_r(x_i) + \frac{\lambda}{2}\|x_i - \bar{x}\|^2\right] - \widehat{f^{\text{erm}}}_r(x^\star_{\bar{x},\lambda}) - \frac{\lambda}{2}\|x^\star_{\bar{x},\lambda} - \bar{x}\|^2 \leq \frac{C_{\text{sc}}L^2}{\lambda}\left(\frac{d}{\beta^2T^2} + \frac{1}{T}\right).$$

Further, by strong convexity and Jensen's inequality we have

$$\mathbb{E}[\|x_i - x^\star_{\bar{x},\lambda}\|] \leq \frac{\sqrt{2C_{\text{sc}}}L}{\lambda}\left(\frac{d}{\beta^2T^2} + \frac{1}{T}\right)^{\frac{1}{2}}.$$

Hence, by Markov's inequality, for each $i \in [k]$ we have

$$\Pr\left[\|x_i - x^\star_{\bar{x},\lambda}\| \geq \frac{3\sqrt{2C_{\text{sc}}}L}{\lambda}\left(\frac{d}{\beta^2T^2} + \frac{1}{T}\right)^{\frac{1}{2}}\right] \leq \frac{1}{3}.$$

Hence by a Chernoff bound, with probability $\geq 1 - \zeta$, at least $0.51k$ points $x \in \{x_i\}_{i\in[k]}$ satisfy

$$\|x - x^\star_{\bar{x},\lambda}\| \leq \frac{3\sqrt{2C_{\text{sc}}}L}{\lambda}\left(\frac{d}{\beta^2T^2} + \frac{1}{T}\right)^{\frac{1}{2}}.$$

Hence the precondition of Lemma 11 holds, giving the distance guarantee with high probability. The privacy guarantee follows from Proposition 8 and the composition of approximate RDP, where we adjusted C_{priv} by a constant and the definition of δ by a factor of k. □

Now we are ready to prove our main result on private ERM.

Theorem 3 (Private ERM). *In the setting of Problem 2, let $\epsilon_{\text{dp}} \in (0, 1)$ and $\delta \in (0, \frac{1}{6})$. There is an $(\epsilon_{\text{dp}}, \delta)$-DP algorithm which takes as input $\mathcal{D}$ and outputs $\widehat{x} \in \mathbb{R}^d$ such that*

$$\mathbb{E}\left[f^{\text{erm}}(\widehat{x}) - \min_{x\in\mathbb{B}(R)} f^{\text{erm}}(x)\right] \leq O\left(LR \cdot \left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}\log^{1.5}(\frac{n}{\delta})\log n}{n\epsilon_{\text{dp}}}\right)\right).$$

Moreover, with probability at least $1 - \delta$, the algorithm queries at most the following number of gradients:

$$O\left(\min\left(n + \frac{(nd)^{\frac{2}{3}}}{\epsilon_{\text{dp}}}, \frac{n^2\epsilon_{\text{dp}}^2}{d} + n^{\frac{4}{3}}\epsilon_{\text{dp}}^{\frac{1}{3}}\right)\log^6\left(\frac{n}{\delta}\right)\right).$$

Proof. Throughout this proof, set for a sufficiently large constant C,

$$\epsilon_{\text{opt}} := CLR\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log\frac{1}{\delta}}\log^{1.5}(\frac{n}{\delta})\log n}{n\epsilon_{\text{dp}}}\right),$$
$$\kappa := \frac{LR}{\epsilon_{\text{opt}}},\ \rho := \frac{\epsilon_{\text{opt}}}{L\sqrt{d}},\ r := \frac{\rho}{\sqrt{C}\log^2(\frac{n}{\delta})}, \quad (30)$$
$$\alpha := \frac{4\log\frac{2}{\delta}}{\epsilon_{\text{dp}}},\ \beta := \frac{\epsilon_{\text{dp}}}{C\log(\frac{n}{\delta})\sqrt{\log\frac{1}{\delta}}}.$$

Note that for the given parameter settings, for sufficiently large C, we have

$$\kappa \le \frac{1}{C}\min\left(\sqrt{n}, \frac{n\epsilon_{\text{dp}}}{\sqrt{d\log\frac{1}{\delta}}\log^{1.5}(\frac{n}{\delta})\log n}\right), \qquad \frac{R}{r} \le n\log^2\left(\frac{\log n}{\delta}\right). \tag{31}$$

Our algorithm proceeds as follows. We apply Proposition 1 with $x^\star \leftarrow \arg\min_{x\in\mathbb{B}(R)} f^{\text{erm}}(x)$ and $F \leftarrow \widehat{f^{\text{erm}}_\rho}$, and instantiate the necessary oracles as follows for $C_{\text{ba}}K\log\kappa$ iterations.

1) We use Algorithm 7 with r, ρ, β defined in (30), and

$$T_1 := \sqrt{C}\left(\frac{\kappa\sqrt{d}}{\sqrt{K}\beta\log^2\kappa} + \frac{\kappa^2}{K\log^3\kappa\log\frac{n}{\delta}}\right), \qquad \zeta := \frac{1}{\kappa C_{\text{ba}}K\log\kappa}, \tag{32}$$

as a $(\frac{r}{C_{\text{ba}}}, \lambda)$-line search oracle $\mathcal{O}_{\text{ls}}$.

2) We use Algorithm 5 with r, ρ, β defined in (30), and

$$T_2 := \sqrt{C}\left(\frac{\kappa\sqrt{d}}{\sqrt{K}\beta\sqrt{\log\kappa}} + \frac{\kappa^2}{K\log\kappa}\right), \tag{33}$$

as a $(\frac{\lambda r^2}{C_{\text{ba}}\log^3\kappa}, \lambda)$-ball optimization oracle $\mathcal{O}_{\text{bo}}$.

3) We use Algorithm 6 with r, ρ, β defined in (30), and

$$T_3 := \sqrt{C}\left(\frac{\kappa\sqrt{d}}{\sqrt{K}\beta} + \frac{\kappa^2}{K}\right) \tag{34}$$

as a $(\frac{\epsilon_{\text{opt}}}{C_{\text{ba}}R}, \frac{\epsilon_{\text{opt}}\sqrt{K}}{C_{\text{ba}}R}, \lambda)$-stochastic proximal oracle $\mathcal{O}_{\text{sp}}$.

We split the remainder of the proof into four parts. We first show that the oracle definitions are indeed met. We then bound the overall optimization error against f^{erm}. Finally, we discuss the privacy guarantee and the gradient complexity bound.

Oracle correctness. For the line search oracle, by Proposition 10, it suffices to show

$$\frac{C_{\text{ls}}L}{\lambda}\cdot\left(\frac{\sqrt{d}}{\beta T_1} + \frac{1}{\sqrt{T_1}}\right) \le \frac{r}{C_{\text{ba}}}.$$

This is satisfied for T_1 in (32), since Proposition 1 guarantees $\lambda \ge \frac{\epsilon_{\text{opt}}K^2\log^2\kappa}{R^2C_{\text{ba}}}$. Hence,

$$\frac{C_{\text{ls}}L}{\lambda}\cdot\frac{\sqrt{d}}{\beta T_1}\cdot\frac{C_{\text{ba}}}{r} \le C_{\text{ls}}C_{\text{ba}}^2\cdot\frac{\kappa\sqrt{d}}{\beta\log^2\kappa}\cdot\frac{1}{\sqrt{K}}\cdot\frac{1}{T_1} \le \frac{1}{2},$$
$$\frac{C_{\text{ls}}L}{\lambda}\cdot\frac{1}{\sqrt{T_1}}\cdot\frac{C_{\text{ba}}}{r} \le C_{\text{ls}}C_{\text{ba}}^2\cdot\frac{\kappa}{\log^2\kappa}\cdot\frac{1}{\sqrt{K}}\cdot\frac{1}{\sqrt{T_1}} \le \frac{1}{2},$$

for a sufficiently large C, where we used $K^{1.5} = \frac{R}{r}$ to simplify. By a union bound, the above holds with probability at least $1 - \frac{\epsilon_{\text{opt}}}{LR}$ over all calls to Algorithm 7, since there are at most $C_{\text{ba}}K\log\kappa$ iterations. For the remainder of the proof, let $\mathcal{E}_{\text{ls}}$ be the event that all line search oracles succeed. For the ball optimization oracle, by Proposition 8, it suffices to show

$$\frac{C_{\text{sc}}L^2}{\lambda}\left(\frac{d}{\beta^2T_2^2} + \frac{1}{T_2}\right) \le \frac{\lambda r^2}{C_{\text{ba}}\log^3\kappa}.$$

This is satisfied for our choice of T_2 in (33), again with $\lambda \ge \frac{\epsilon_{\text{opt}}K^2\log^2\kappa}{R^2C_{\text{ba}}}$. Hence,

$$\frac{C_{\text{sc}}L^2}{\lambda}\cdot\frac{d}{\beta^2T_2^2}\cdot\frac{C_{\text{ba}}\log^3\kappa}{\lambda r^2} \le C_{\text{sc}}C_{\text{ba}}^3\cdot\frac{\kappa^2 d}{\beta^2\log\kappa}\cdot\frac{1}{K}\cdot\frac{1}{T_2^2} \le \frac{1}{2},$$
$$\frac{C_{\text{sc}}L^2}{\lambda}\cdot\frac{1}{T_2}\cdot\frac{C_{\text{ba}}\log^3\kappa}{\lambda r^2} \le C_{\text{sc}}C_{\text{ba}}^3\cdot\frac{\kappa^2}{\log\kappa}\cdot\frac{1}{K}\cdot\frac{1}{T_2} \le \frac{1}{2},$$

again for large C. Finally, for the proximal gradient oracle, by Proposition 9, it suffices to show

$$C_{\text{bias}}\left(\frac{L}{\lambda}\cdot\left(\frac{\sqrt{d}}{\beta n} + \frac{1}{\sqrt{n}}\right)\right) \le \frac{\epsilon_{\text{opt}}}{C_{\text{ba}}\lambda R},$$
$$\frac{C_{\text{var}}L^2}{\lambda^2}\left(\frac{d}{\beta^2T_3^2} + \frac{1}{T_3}\right) \le \frac{\epsilon_{\text{opt}}^2K}{C_{\text{ba}}^2\lambda^2R^2}.$$

The first inequality is clear. The second is satisfied for our choice of T_3 in (34), which implies

$$\frac{C_{\text{var}}L^2}{\lambda^2}\cdot\frac{d}{\beta^2T_3^2}\cdot\frac{C_{\text{ba}}^2\lambda^2R^2}{\epsilon_{\text{opt}}^2K} = C_{\text{var}}C_{\text{ba}}^2\cdot\frac{\kappa^2 d}{\beta^2}\cdot\frac{1}{K}\cdot\frac{1}{T_3^2} \le \frac{1}{2},$$
$$\frac{C_{\text{var}}L^2}{\lambda^2}\cdot\frac{1}{T_3}\cdot\frac{C_{\text{ba}}^2\lambda^2R^2}{\epsilon_{\text{opt}}^2K} = C_{\text{var}}C_{\text{ba}}^2\cdot\kappa^2\cdot\frac{1}{K}\cdot\frac{1}{T_3} \le \frac{1}{2}.$$

Optimization error. By Proposition 1, the expected optimization error against $\widehat{f^{\text{erm}}_\rho}$ is bounded by ϵ_{opt} whenever $\mathcal{E}_{\text{ls}}$ occurs. Otherwise, the optimization error is never larger than LR as long as we return a point in $\mathbb{B}(R)$, since the function is L-Lipschitz. Further, we showed $\Pr[\mathcal{E}_{\text{ls}}] \ge 1 - \frac{\epsilon_{\text{opt}}}{LR}$, so the total expected error is bounded by $2\epsilon_{\text{opt}}$. Finally, the additive error between $\widehat{f^{\text{erm}}_\rho}$ and f^{erm} is bounded by $\rho L\sqrt{d} = \epsilon_{\text{opt}}$. The conclusion follows by setting the error bound to $3\epsilon_{\text{opt}}$.

Privacy. We first claim that each call to $\mathcal{O}_{\text{ls}}$, and $\mathcal{O}_{\text{bo}}$ used by Proposition 1 satisfies

$$\left(\alpha, \frac{\epsilon_{\text{dp}}}{6C_{\text{ba}}K\log\kappa}, \frac{\delta}{18C_{\text{ba}}K\log\kappa}\right)\text{-RDP}.$$

We first analyze $\mathcal{O}_{\text{ls}}$. The preconditions of Proposition 10 are met, where $\log(\frac{18C_{\text{ba}}K\log\kappa}{\delta}\log(\frac{T}{\zeta})) \le 2\log\frac{n}{\delta}$ for our parameter settings. Moreover, our α is in the acceptable range. Finally, by Proposition 10 it suffices to note

$$\frac{8\alpha C_{\text{priv}}\beta^2T_1^2\log^3\left(\frac{n}{\delta}\right)}{n^2} \le \frac{128CC_{\text{priv}}\beta^2\log^3(\frac{n}{\delta})\log\frac{1}{\delta}}{n^2\epsilon_{\text{dp}}}\cdot\left(\frac{\kappa^2 d}{K\beta^2\log\kappa} + \frac{\kappa^4}{K^2\log^2\kappa}\right) \le \frac{\epsilon_{\text{dp}}}{6C_{\text{ba}}K\log\kappa},$$

where the second inequality follows for sufficiently large C due to (31). Next, we analyze the privacy of $\mathcal{O}_{\text{bo}}$. The preconditions of Proposition 8 are met, where $\log(\frac{\log\log T}{\delta}) \le \log\frac{n}{\delta}$

for our parameter settings, and our α is again acceptable. Finally, by Proposition 8 it suffices to note

$$\frac{\alpha C_{\text{priv}}\beta^2 T_2^2 \log^2(\frac{n}{\delta})}{n^2} \le \frac{16CC_{\text{priv}}\beta^2\log^2(\frac{n}{\delta})\log\frac{1}{\delta}}{n^2\epsilon_{\text{dp}}} \cdot \left(\frac{\kappa^2 d}{K\beta^2\log\kappa} + \frac{\kappa^4}{K^2\log^2\kappa}\right) \le \frac{\epsilon_{\text{dp}}}{6C_{\text{ba}}K\log\kappa},$$

again for sufficiently large C from (31). Hence, by applying Lemma 3, all of the at most $C_{\text{ba}}K\log\kappa$ calls to $\mathcal{O}_{\text{ls}}$ and $\mathcal{O}_{\text{bo}}$ used by the algorithm combined satisfy

$$\left(\alpha, \frac{\epsilon_{\text{dp}}}{3}, \frac{\delta}{9}\right)\text{-RDP}.$$

Finally, we analyze the privacy of $\mathcal{O}_{\text{sp}}$. Let

$$j_{\max} := \left\lfloor \log_2\left(\frac{1}{T_3}\cdot\left\lfloor\frac{n}{C_{\text{priv}}}\right\rfloor\right)\right\rfloor$$

be the truncation parameter in Algorithm 6. The total number of draws from Geom($\frac{1}{2}$) in Algorithm 6 over the course of the algorithm is $C_{\text{ba}}K\log\kappa\cdot j_{\max}$. It is straightforward to check that the expected number of draws where $J=j$ for all $j\in[j_{\max}]$ is

$$2^{-j_{\max}}C_{\text{ba}}\kappa\log\kappa\cdot j_{\max} = \Omega\left(\frac{T_3}{n}\cdot K\log\kappa\cdot j_{\max}\right),$$

which is superconstant. By Chernoff and a union bound, with probability $\ge 1-\frac{\delta}{n}$, there is a constant C' such that for all $j\in[j_{\max}]$, the number of times we draw $J=j$ is bounded by

$$2^{-j}C'K\log\kappa\log\frac{n}{\delta}.$$

Similarly, the number of times we draw $J\notin[j_{\max}]$ is bounded by $C'K\log\kappa\log\frac{n}{\delta}$. This implies by Lemma 3 that all calls to $\mathcal{O}_{\text{sp}}$ used by the algorithm combined satisfy

$$\left(\alpha, \frac{\epsilon_{\text{dp}}}{6}, \frac{\delta}{18}\right)\text{-RDP}.$$

Here, we summed the privacy loss in Lemma 10 over $0\le J\le j_{\max}$, which gives

$$\sum_{0\le j\le j_{\max}}\left(2^j\cdot\frac{\alpha C_{\text{priv}}\beta^2\log^2(\frac{n}{\delta})T_3^2}{n^2}\right)\left(2^{-j}C'K\log\kappa\log\frac{n}{\delta}\right) \le (j_{\max}+1)\cdot\frac{16CC'C_{\text{priv}}K\beta^2\log^3(\frac{n}{\delta})\log\frac{1}{\delta}\log\kappa}{n^2\epsilon_{\text{dp}}} \cdot\left(\frac{\kappa^2 d}{K\beta^2}+\frac{\kappa^4}{K^2}\right) \le \frac{\epsilon_{\text{dp}}}{6},$$

for sufficiently large C, where we use $\log\kappa, j_{\max}\le\log n$, and $K\ge\log\frac{1}{\delta}$ for our parameter settings. Finally, combining these bounds shows that our whole algorithm satisfies $(\alpha, \frac{\epsilon_{\text{dp}}}{2}, \frac{\delta}{6})$-RDP, and applying Corollary 1, gives the desired privacy guarantee.

Gradient complexity. We have argued that with probability at least $1-\delta$, the number of times we encounter the $J=j$ case of Lemma 10 for all $0\le j\le j_{\max}$ is bounded by $2^{-j}C'K\log\kappa\log\frac{n}{\delta}$. Under this event, Proposition 10, Proposition 8, and Lemma 10 imply the total gradient complexity of our algorithm is at most

$$C_{\text{ba}}K\log\kappa\cdot\left(18T_1\log\frac{1}{\zeta}+T_2+\sum_{0\le j\le j_{\max}}\left(2^{-j}C'\log\frac{n}{\delta}\right)\left(2^{j+1}T_3\right)\right) \le 36C_{\text{ba}}C'K\log n\left(T_1\log n+T_2+T_3\log n\log\frac{n}{\delta}\right),$$

where we use $\zeta\ge n^{-2}$, $j_{\max}\le\log n$, and $\kappa\le n$. The conclusion follows from plugging in our parameter choices from (32), (33), and (34). □

Finally, we note that following the strategy of Section IV-C, it is straightforward to extend Theorem 3 to the strongly convex setting. We state this result as follows.

Corollary 3 (Private regularized ERM)**.** *In the setting of Problem 2, let $\epsilon_{\text{dp}}\in(0,1)$, $\delta\in(0,\frac{1}{6})$, $\lambda\ge 0$, and $x'\in\mathbb{B}(R)$. There is an $(\epsilon_{\text{dp}},\delta)$-DP algorithm which outputs $\widehat{x}\in\mathbb{B}(R)$ such that*

$$\mathbb{E}\left[f^{\text{erm}}(\widehat{x})+\frac{\lambda}{2}\|x-x'\|^2\right] - \min_{x\in\mathbb{B}(R)}\left\{f^{\text{erm}}(x)+\frac{\lambda}{2}\|x-x'\|^2\right\} \le O\left(\frac{L^2}{\lambda}\cdot\left(\frac{1}{n}+\frac{d\log\frac{1}{\delta}\log^3(\frac{n}{\delta})\log^2 n}{n^2\epsilon_{\text{dp}}^2}\right)\right).$$

Moreover, with probability at least $1-\delta$, the algorithm queries at most the following number of gradients:

$$O\left(\min\left(n+\frac{(nd)^{\frac{2}{3}}}{\epsilon_{\text{dp}}}, \frac{n^2\epsilon_{\text{dp}}^2}{d}+n^{\frac{4}{3}}\epsilon_{\text{dp}}^{\frac{1}{3}}\right)\log^6\left(\frac{n}{\delta}\right)\right).$$

Proof. We first note that similar to Corollary 2 (an extension of Proposition 6), it is straightforward to extend Theorem 3 to handle both regularization and an improved upper bound on the distance to the optimum, with the same error rate and privacy guarantees otherwise. The handling of the improved upper bound on the distance follows because the convergence rate of the [ACJ+21] algorithm scales proportionally to the distance to the optimum, when it is smaller than R. The regularization is handled in the same way as Corollary 2, where regularization can only improve the contraction in the privacy proof. One subtle point is that for the regularized problems, we need to obtain starting points for Algorithm 5 when the constraint set is $\mathbb{B}_{\bar{x}}(r)$, but the regularization in the objective is centered around a point not in $\mathbb{B}_{\bar{x}}(r)$ (in our case, the centerpoint will be a weighted combination of $\bar{x}$ and x'). However, by initializing Algorithm 5 at the projection of the regularization centerpoint, the initial function error guarantee in Lemma 8 still holds (see Lemma 9).

The reduction from the claimed rate in this corollary statement to the regularized extension of Theorem 3 then

proceeds identically to the proof of Proposition 8, which calls Corollary 2 repeatedly. □

F. Private SCO solver

Finally, we give our main result on private SCO in this section. To obtain it, we will combine Corollary 3 with a generic reduction in [FKT20], [KLL21], which uses a private ERM solver as a black box. The reduction is based on the iterative localization technique proposed by [FKT20] (which is the same strategy used by Section IV-C), and derived in greater generality by [KLL21].

Proposition 11 (Modification of Theorem 5.1 in [KLL21])**.** *Suppose there is an $(\epsilon_{\text{dp}}, \delta)$-DP algorithm $\mathcal{A}_{\text{erm}}$ with expected excess loss*

$$O\left(\frac{L^2}{\lambda} \cdot \left(\frac{1}{n} + \frac{d \log \frac{1}{\delta} \log^3(\frac{n}{\delta}) \log^2 n}{n^2 \epsilon_{\text{dp}}^2}\right)\right),$$

using $N(n, \epsilon_{\text{dp}}, \delta)$ gradient queries, for some function N, when applied to an L-Lipschitz empirical risk (with n samples, constrained to $\mathbb{B}(R) \subset \mathbb{R}^d$) plus a λ-strongly convex regularizer. Then there is an $(\epsilon_{\text{dp}}, \delta)$-DP algorithm $\mathcal{A}_{\text{sco}}$ using $\sum_{i \in \lceil \log n \rceil} N(\frac{n}{2^i}, \frac{\epsilon_{\text{dp}}}{2^i}, \frac{\delta}{2^i})$ gradient queries, with expected excess population loss

$$O\left(LR \cdot \left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d \log \frac{1}{\delta}} \log^{1.5}(\frac{n}{\delta}) \log n}{n \epsilon_{\text{dp}}}\right)\right).$$

Theorem 5.1 in [KLL21] assumes a slightly smaller risk guarantee for $\mathcal{A}_{\text{erm}}$ (removing the extraneous $\log^3(\frac{n}{\delta}) \log^2 n$ factor), but it is straightforward to see that the proof extends to handle our larger risk assumption. Combining Proposition 11 and Corollary 3 then gives our main result.

Theorem 4 (Private SCO)**.** *In the setting of Problem 2, let $\epsilon_{\text{dp}} \in (0, 1)$ and $\delta \in (0, \frac{1}{6})$. There is an $(\epsilon_{\text{dp}}, \delta)$-DP algorithm which takes as input $\mathcal{D}$ and outputs $\widehat{x} \in \mathbb{R}^d$ such that*

$$\mathbb{E}\left[f^{\text{pop}}(\widehat{x}) - \min_{x \in \mathbb{B}(R)} f^{\text{pop}}(x)\right]$$
$$\leq O\left(LR \cdot \left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d \log \frac{1}{\delta}} \log^{1.5}(\frac{n}{\delta}) \log n}{n \epsilon_{\text{dp}}}\right)\right).$$

Moreover, with probability at least $1 - \delta$, the algorithm queries at most the following number of gradients:

$$O\left(\min\left(n + \frac{(nd)^{\frac{2}{3}}}{\epsilon_{\text{dp}}}, \frac{n^2 \epsilon_{\text{dp}}^2}{d} + n^{\frac{4}{3}} \epsilon_{\text{dp}}^{\frac{1}{3}}\right) \log^6\left(\frac{n}{\delta}\right)\right).$$

ACKNOWLEDGMENT

We thank Vijaykrishna Gurunathan for helpful conversations on parallel convex optimization that facilitated initial insights regarding ReSQue. We also thank the anonymous reviewers for their feedback. YC was supported in part by the Israeli Science Foundation (ISF) grant no. 2486/21 and the Len Blavatnik and the Blavatnik Family foundation. AS was supported in part by a Microsoft Research Faculty Fellowship, NSF CAREER Award CCF-1844855, NSF Grant CCF-1955039, a PayPal research award, and a Sloan Research Fellowship.

REFERENCES

[Abo16] John M. Abowd. The challenge of scientific reproducibility and privacy protection for statistical agencies. *Technical report, Census Scientific Advisory Committee*, 2016.

[ACG+16] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016.

[ACJ+21] Hilal Asi, Yair Carmon, Arun Jambulapati, Yujia Jin, and Aaron Sidford. Stochastic bias-reduced gradient methods. In *Advances in Neural Information Processing Systems, NeurIPS*, 2021.

[AFKT21] Hilal Asi, Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private stochastic convex optimization: Optimal rates in l1 geometry. In *International Conference on Machine Learning, ICML*, 2021.

[App17] Differential Privacy Team Apple. Learning with privacy at scale. *Technical report, Apple*, 2017.

[BBG18] Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy amplification by subsampling: Tight analyses via couplings and divergences. In *Advances in Neural Information Processing Systems, NeurIPS*, 2018.

[BDRS18] Mark Bun, Cynthia Dwork, Guy N. Rothblum, and Thomas Steinke. Composable and versatile privacy via truncated CDP. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, 2018.

[BFGT20] Raef Bassily, Vitaly Feldman, Cristóbal Guzmán, and Kunal Talwar. Stability of stochastic gradient descent on nonsmooth convex losses. *Advances in Neural Information Processing Systems*, 33:4381–4391, 2020.

[BFTT19] Raef Bassily, Vitaly Feldman, Kunal Talwar, and Abhradeep Guha Thakurta. Private stochastic convex optimization with optimal rates. In *Advances in Neural Information Processing Systems, NeurIPS*, 2019.

[BJL+19] Sébastien Bubeck, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, and Aaron Sidford. Complexity of highly parallel non-smooth convex optimization. In *Advances in Neural Information Processing Systems, NeurIPS*, 2019.

[Bot12] Léon Bottou. Stochastic gradient descent tricks. In Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade - Second Edition*, volume 7700 of *Lecture Notes in Computer Science*, pages 421–436. Springer, 2012.

[BS18] Eric Balkanski and Yaron Singer. Parallelization does not accelerate convex optimization: Adaptivity lower bounds for non-smooth convex minimization. *arXiv: 1808.03880*, 2018.

[BST14] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *IEEE 55th Annual Symposium on Foundations of Computer Science, FOCS*, 2014.

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Found. Trends Mach. Learn.*, 8(3-4):231–357, 2015.

[BV14] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2014.

[CH22] Yair Carmon and Danielle Hausler. Distributionally robust optimization via ball oracle acceleration. *arXiv:2203.13225*, 2022.

[CJJ+20] Yair Carmon, Arun Jambulapati, Qijia Jiang, Yujia Jin, Yin Tat Lee, Aaron Sidford, and Kevin Tian. Acceleration with a ball optimization oracle. In *Advances in Neural Information Processing Systems, NeurIPS*, 2020.

[CJJS21] Yair Carmon, Arun Jambulapati, Yujia Jin, and Aaron Sidford. Thinking inside the ball: Near-optimal minimization of the maximal loss. In *Conference on Learning Theory, COLT*, 2021.

[CJST19] Yair Carmon, Yujia Jin, Aaron Sidford, and Kevin Tian. Variance reduction for matrix games. In *Advances in Neural Information Processing Systems, NeurIPS*, 2019.

[CM08] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems, NeurIPS*, 2008.

[CMS11] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

[DBW12] John C Duchi, Peter L Bartlett, and Martin J Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

[DG19] Jelena Diakonikolas and Cristóbal Guzmán. Lower bounds for parallel and randomized convex optimization. In *Conference on Learning Theory, COLT*, 2019.

[DGBSX12] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(1), 2012.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[DRY18] John Duchi, Feng Ruan, and Chulhee Yun. Minimax bounds on stochastic batched convex optimization. In *Conference On Learning Theory, COLT*, 2018.

[Duc18] John C Duchi. Introductory lectures on stochastic optimization. *The Mathematics of Data*, pages 99–186, 2018.

[EPK14] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014.

[FKT20] 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, STOC*, 2020.

[FTS17] Kazuto Fukuchi, Quang Khai Tran, and Jun Sakuma. Differentially private empirical risk minimization with input perturbation. In *International Conference on Discovery Science*, 2017.

[GDG+19] Alexander Gasnikov, Pavel Dvurechensky, Eduard Gorbunov, Evgeniya Vorontsova, Daniil Selikhanovych, César A Uribe, Bo Jiang, Haoyue Wang, Shuzhong Zhang, Sébastien Bubeck, et al. Near optimal methods for minimizing convex functions with lipschitz p-th derivatives. In *Conference on Learning Theory, COLT*, 2019.

[GL12] Saeed Ghadimi and Guanghui Lan. Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework. *SIAM Journal on Optimization*, 22(4):1469–1492, 2012.

[GLL22] Sivakanth Gopi, Yin Tat Lee, and Daogao Liu. Private convex optimization via exponential mechanism. *arXiv:2203.00263*, 2022.

[Gol64] A. A. Goldstein. Convex programming in hilbert space. 70(5):709—710, 1964.

[HK14] Elad Hazan and Satyen Kale. Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization. *J. Mach. Learn. Res.*, 15(1):2489–2512, 2014.

[INS+19] Roger Iyengar, Joseph P Near, Dawn Song, Om Thakkar, Abhradeep Thakurta, and Lun Wang. Towards practical differentially private convex optimization. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

[JLSW20] Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC*, 2020.

[JT14] Prateek Jain and Abhradeep Guha Thakurta. (near) dimension independent risk bounds for differentially private learning. In *International Conference on Machine Learning, ICML*, 2014.

[KJ16] Shiva Prasad Kasiviswanathan and Hongxia Jin. Efficient private empirical risk minimization for high-dimensional learning. In *International Conference on Machine Learning, ICML*, 2016.

[KLL21] Janardhan Kulkarni, Yin Tat Lee, and Daogao Liu. Private non-smooth erm and sco in subquadratic steps. In *Advances in Neural Information Processing Systems, NeurIPS*, 2021.

[KLL+22] Jonathan A. Kelner, Jerry Li, Allen Liu, Aaron Sidford, and Kevin Tian. Semi-random sparse recovery in nearly-linear time. *arXiv:2203.04002*, 2022.

[KTE88] Leonid G. Khachiyan, Sergei Pavlovich Tarasov, and I. I. Erlikh. The method of inscribed ellipsoids. *Soviet Math. Dokl.*, 37:226–230, 1988.

[LLH+22] Xuechen Li, Daogao Liu, Tatsunori Hashimoto, Huseyin A Inan, Janardhan Kulkarni, Yin Tat Lee, and Abhradeep Guha Thakurta. When does differentially private learning not suffer in high dimensions? *arXiv:2207.00160*, 2022.

[Mir17] Ilya Mironov. Rényi differential privacy. In *IEEE 30th Computer Security Foundations Symposium, CSF*, 2017.

[MS13] Renato DC Monteiro and Benar Fux Svaiter. An accelerated hybrid proximal extragradient method for convex optimization and its implications to second-order methods. *SIAM Journal on Optimization*, 23(2):1092–1125, 2013.

[Nem94] Arkadi Nemirovski. On parallel complexity of nonsmooth convex optimization. *Journal of Complexity*, 10(4):451–463, 1994.

[Nes83] Yu E Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. In *Dokl. Akad. Nauk SSSR,*, 1983.

[Nes03] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

[Nes18] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[NY83] Arkadi S. Nemirovski and David B. Yudin. Problem complexity and method efficiency in optimization. 1983.

[Pol64] Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[RBHT12] Benjamin IP Rubinstein, Peter L Bartlett, Ling Huang, and Nina Taft. Learning in a large function space: Privacy-preserving mechanisms for svm learning. *Journal of Privacy and Confidentiality*, 4(1):65–100, 2012.

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014.

[SBB+18] Kevin Scaman, Francis Bach, Sébastien Bubeck, Laurent Massoulié, and Yin Tat Lee. Optimal algorithms for non-smooth distributed optimization in networks. In *Advances in Neural Information Processing Systems, NeurIPS*, 2018.

[Sha07] Shai Shalev-Shwartz. Online learning: Theory, algorithms, and applications. PhD thesis, Hebrew University, 2007.

[Smi09] Adam Smith. Differential privacy and the secrecy of the sample. https://adamdsmith.wordpress.com/2009/09/02/sample-secrecy/, 2009. Accessed: 2022-11-06.

[SSTT21] Shuang Song, Thomas Steinke, Om Thakkar, and Abhradeep Thakurta. Evading the curse of dimensionality in unconstrained private glms. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, 2021.

[SU15] Thomas Steinke and Jonathan Ullman. Between pure and approximate differential privacy. *arXiv:1501.06095*, 2015.

[Tal22] Kunal Talwar. Ppml workshop talk: Open questions in differentially private machine learning. https://machinelearning.apple.com/video/open-questions, 2022. Accessed: 2022-11-06.

[Wan18] Yu-Xiang Wang. Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain. *arXiv:1803.02596*, 2018.

[WBSS21] Blake E Woodworth, Brian Bullins, Ohad Shamir, and Nathan Srebro. The min-max complexity of distributed stochastic convex optimization with intermittent communication. In *Conference on Learning Theory, COLT*, 2021.

[ZZMW17] Jiaqi Zhang, Kai Zheng, Wenlong Mou, and Liwei Wang. Efficient private erm for smooth objectives. In *International Joint Conference on Artificial Intelligence, IJCAI*, 2017.

APPENDIX

Fact 2. *Let $p \in \mathbb{N}$. For any integer r such that $0 \le r \le p-1$, $\sum_{0\le q\le p}(-1)^q \binom{p}{q} q^r = 0$.*

Proof. We recognize the formula as a scaling of the Stirling number of the second kind with r objects and p bins, i.e., the number of ways to put r objects into p bins such that each bin has at least one object. When $r < p$ there are clearly no such ways. □

Fact 3. *Let $p \in \mathbb{N}$ be even and $p \ge 2$. Let $\|x\|, \|y\| \le \frac{1}{p}$. Then for $g(q) := \left((p-q)^2 - (p-q)\right)\|x\|^2 + (q^2-q)\|y\|^2 + 2q(p-q)\langle x, y\rangle$,*

$$\sum_{0\le q\le p} (-1)^q \binom{p}{q} \exp\left(\frac{1}{2}g(q)\right) \le (12p\|x-y\|)^p.$$

Proof. Fix some x. Let $f_x(y)$ be the left-hand side displayed above, and let

$$f_x^q(y) := \exp\left(\frac{1}{2}g(q)\right).$$

We will perform a p^{th} order Taylor expansion of f_x around x, where we show that partial derivatives of order at most $p-1$ are all zero at x, and we bound the largest order derivative tensor.

Derivatives of f_x^q. Fix some $0 \le q \le p$, and define

$$C_q := q^2 - q,\ F_q := f_x^q(y),\ v_q := (q^2-q)y + q(p-q)x. \quad (35)$$

Note that for fixed q, F_q and v_q are functions of y, and we defined them such that $\nabla_y v_q = C_q \mathbf{I}_d$, $\nabla_y F_q = v_q F_q$. Next, in the following we use $\sum_{\text{sym}}$ to mean a symmetric sum over all choices of tensor modes, e.g. $\sum_{\text{sym}} v_q^{\otimes 2} \otimes \mathbf{I}_d$ means we will choose 2 of the 4 modes where the action is $v_q^{\otimes 2}$. To gain some intuition for the derivatives of F_q, we begin by evaluating the first few via product rule:

$$\begin{aligned}
\nabla f_x^q(y) &= F_q v_q,\\
\nabla^2 f_x^q(y) &= F_q v_q^{\otimes 2} + C_q F_q \mathbf{I}_d,\\
\nabla^3 f_x^q(y) &= F_q v_q^{\otimes 3} + C_q F_q \sum_{\text{sym}} v_q \otimes \mathbf{I}_d,\\
\nabla^4 f_x^q(y) &= F_q v_q^{\otimes 4} + C_q F_q \sum_{\text{sym}} v_q^{\otimes 2} \otimes \mathbf{I}_d + 3C_q^2 F_q \mathbf{I}_d \otimes \mathbf{I}_d.
\end{aligned}$$

For any fixed $0 \le r \le p$, we claim that the r^{th} derivative tensor has the form

$$\nabla^r f_x^q(y) = F_q \left(\sum_{0\le s\le \lfloor \frac{r}{2} \rfloor} \frac{N_{r,s}}{\binom{r}{2s}} \left((C_q)^s \sum_{\text{sym}} v_q^{\otimes(r-2s)} \otimes \mathbf{I}_d^{\otimes s} \right) \right), \quad (36)$$

where the $N_{r,s}$ are nonnegative coefficients which importantly do not depend on q. To see this we proceed by induction; the base cases are computed above. Every time we take the derivative of a "monomial" term of the form $F_q (C_q)^s v_q^{\otimes(r-2s)} \otimes \mathbf{I}_d^{\otimes s}$

via product rule, we will have one term in which F_q becomes $v_q F_q$ (and hence we obtain a $F_q C_q^s v_q^{\otimes(r+1-2s)} \otimes \mathbf{I}_d^{\otimes s}$ monomial), and $r-2s$ many terms where a v_q becomes $C_q \mathbf{I}_d$ (and hence we obtain a $F_q C_q^{s+1} v_q^{\otimes(r-1-2s)} \otimes \mathbf{I}_d^{\otimes(s+1)}$ monomial). For fixed $0 \le s \le \lfloor \frac{r+1}{2} \rfloor$, we hence again see that $N_{r+1,s}$ has no dependence on q.

Next, note $\sum_{0\le s\le \lfloor \frac{r}{2} \rfloor} N_{r,s}$ has a natural interpretation as the total number of "monomial" terms of the form $F_q(C_q)^s v_q^{\otimes(r-2s)} \otimes \mathbf{I}_d^{\otimes s}$ when expanding $\nabla^r f_x^q(y)$. We claim that for all $0 \le q \le p$ and $0 \le r \le p-1$,

$$\frac{\sum_{0\le s\le \lfloor \frac{r+1}{2} \rfloor} N_{r+1,s}}{\sum_{0\le s\le \lfloor \frac{r}{2} \rfloor} N_{r,s}} \le p. \tag{37}$$

To see this, consider taking an additional derivative of (36) with respect to y. Each monomial of the form $F_q(C_q)^s v_q^{\otimes(r-2s)} \otimes \mathbf{I}_d^{\otimes s}$ contributes at most $r-2s+1 \le p$ monomials to the next derivative tensor via product rule, namely one from F_q and one from each copy of v_q. Averaging this bound over all monomials yields the claim (37), since each contributes at most p.

Taylor expansion at x. Next, we claim that for all $0 \le r \le p-1$,

$$\nabla^r f_x(x) = 0. \tag{38}$$

To see this, we have that $((p-q)^2-(p-q))+(q^2-q)+2q(p-q) = p^2-p$ is independent of q, and hence all of the F_q are equal to some value F when $y=x$. Furthermore, when $y=x$ we have that $v_q = q(p-1)x$. Now, from the characterization (36) and summing over all q, any monomial of the form $x^{\otimes(r-2s)} \otimes \mathbf{I}_d^{\otimes s}$ has a total coefficient of

$$\begin{aligned} &FN_{r,s} \sum_{0\le q\le p} (-1)^q \binom{p}{q} (C_q)^s (q(p-1))^{r-2s} \\ &= FN_{r,s}(p-1)^{r-2s} \sum_{0\le q\le p} (-1)^q \binom{p}{q} C_q^s q^{r-2s}. \end{aligned}$$

Since C_q is a quadratic in q, each summand $(C_q)^s q^{r-2s}$ is a polynomial of degree at most $r \le p-1$ in q, so applying Fact 2 to each monomial yields the claim (38).

Taylor expansion at y. Finally, we will bound the injective tensor norm of $\nabla^p f_x(y)$, where the injective tensor norm of a degree-p symmetric tensor $\mathbf{T}$ is the maximum value of $\mathbf{T}[v^{\otimes p}]$ over unit norm v. We proceed by bounding the injective tensor norm of each monomial and then summing.

First, for any $0 \le p \le q$, under our parameter settings it is straightforward to see $\|v_q\| \le p$ and $F_q \le 2$. Also, for any $0 \le s \le \frac{p}{2}$ we have $C_q^s \le p^{2s}$, and by repeatedly applying (37), we have $\sum_{0\le s\le \lfloor \frac{p}{2} \rfloor} N_{p,s} \le p^p$. In other words, each of the monomials of the form $F_q(C_q)^s v_q^{\otimes(r-2s)} \otimes \mathbf{I}_d^{\otimes s}$ has injective tensor norm at most $2p^p$ (since each C_q contributes two powers of p, and each v_q contributes one power of p), and there are at most p^p such monomials. Hence, by triangle inequality over the sum of all monomials,

$$\left|\nabla^p f_x^q(y)[(y-x)^{\otimes p}]\right| \le 2p^{2p} \|y-x\|^p.$$

By summing the above over all q (reweighting by $(-1)^q\binom{p}{q}$), and using that the unsigned coefficients sum to $\sum_{0\le q\le p}\binom{q}{p} = 2^p$, we have

$$\left|\nabla^p f_x(y)[(y-x)^{\otimes p}]\right| \le 4^p p^{2p} \|x-y\|^p.$$

The conclusion follows by a Taylor expansion from x to y of order p, and using $p^p \le 3^p p!$. □

Proof of Lemma 2. For the first claim,

$$\begin{aligned} &\int \frac{(\gamma_\rho(x-\bar{x}-\xi))^p}{(\gamma_\rho(\xi))^{p-1}} \mathrm{d}\xi = (2\pi\rho)^{-\frac{d}{2}} \\ &\cdot \int \exp\left(-\frac{1}{2\rho^2}\left(p\|x-\bar{x}\|^2 - 2p\langle x-\bar{x},\xi\rangle + \|\xi\|^2\right)\right)\mathrm{d}\xi \\ &= \exp\left(\frac{p^2-p}{2\rho^2}\|x-\bar{x}\|^2\right) \le 2, \end{aligned}$$

where the second equality used the calculation in (6), and the inequality used the assumed bound on $\|x-\bar{x}\|$. We move onto the second claim. First, we prove the statement for all even $p \in \mathbb{N}$. Denote $v := x-\bar{x}$ and $v' := x'-\bar{x}$ for simplicity. Explicitly expanding the numerator yields that

$$(2\pi\rho)^{\frac{d}{2}} \int \frac{(\gamma_\rho(v-\xi)-\gamma_\rho(v'-\xi))^p}{(\gamma_\rho(\xi))^{p-1}}\mathrm{d}\xi = \sum_{0\le q\le p}(-1)^q\binom{p}{q} S_q$$

where we define $h(\xi) := (p-q)\|v\|^2 + q\|v'\|^2 - 2(p-q)\langle v,\xi\rangle - 2q\langle v',\xi\rangle + \|\xi\|^2$ and $H_q := \left((p-q)^2-(p-q)\right)\|v\|^2 + (q^2-q)\|v'\|^2 + 2q(p-q)\langle v,v'\rangle$, and compute

$$\begin{aligned} S_q &:= (2\pi\rho)^{\frac{d}{2}} \int \frac{(\gamma_\rho(v-\xi))^{p-q}(\gamma_r(v'-\xi))^q}{(\gamma_\rho(\xi))^{p-1}}\mathrm{d}\xi \\ &= \int \exp\left(-\frac{1}{2\rho^2}h(\xi)\right)\mathrm{d}\xi \\ &= (2\pi\rho)^{\frac{d}{2}} \exp\left(\frac{1}{2\rho^2}H_q\right). \end{aligned}$$

In the last line, we again used (6) to compute the integral. When $p \ge 2$ and is even, a strengthening of the conclusion then follows from Fact 3 (where we overload $x \leftarrow \frac{v}{\rho}$, $y \leftarrow \frac{v'}{\rho}$ in its application). In particular, this shows the desired claim where the base of the exponent is $\frac{12p}{\rho}\|x-x'\|$ instead of $\frac{24p}{\rho}\|x-x'\|$. We move to general $p \ge 2$. Define the random variable

$$Z := \left|\frac{\gamma_\rho(x-\bar{x}-\xi)-\gamma_\rho(x'-\bar{x}-\xi)}{\gamma_\rho(\xi)}\right|.$$

Recall that we have shown for all even $p \ge 2$,

$$\mathbb{E}Z^p \le \left(\frac{12p\|x-x'\|}{\rho}\right)^p.$$

Now, let $p \ge 2$ be sandwiched between the even integers q and $q+2$. Hölder's inequality and the above inequality (for $p \leftarrow q$ and $p \leftarrow q+2$) demonstrate

$$\mathbb{E}Z^p \le \left(\mathbb{E}Z^q\right)^{\frac{q+2-p}{2}}\left(\mathbb{E}Z^{q+2}\right)^{\frac{p-q}{2}} \le \left(\frac{12(q+2)\|x-x'\|}{\rho}\right)^p,$$

where we use $q(q+2-p)+(q+2)(p-q) = 2p$. The conclusion follows since $q + 2 \le 2p$. □

Fact 4. *Let Z be a nonnegative scalar random variable, let $C \ge 0$ be a fixed scalar, and let $p \in \mathbb{N}$ and $p \ge 2$. Then*

$$\left(\mathbb{E}\left[(Z + C)^p\right]\right)^{\frac{1}{p}} \le \mathbb{E}\left[Z^p\right]^{\frac{1}{p}} + C.$$

Proof. Denote $A := \mathbb{E}\left[Z^p\right]^{\frac{1}{p}}$. Taking p^{th} powers of both sides, we have the conclusion if

$$\begin{aligned} &(A + C)^p - \mathbb{E}\left[(Z + C)^p\right] \ge 0 \\ \iff &\sum_{q \in [p-1]} \binom{p}{q} C^{p-q} \left(A^q - \mathbb{E}\left[Z^q\right]\right) \ge 0. \end{aligned}$$

Here we use that the $q = 0$ and $q = p$ terms cancel. We conclude since Jensen's inequality yields

$$\mathbb{E}[Z^p] \ge \mathbb{E}[Z^q]^{\frac{p}{q}} \implies A^q \ge \mathbb{E}[Z^q], \text{ for all } q \in [p-1].$$

□

In this section, we discuss how to obtain Proposition 1 from the analysis in [ACJ+21]. We separate the discussion into four parts, corresponding to the iteration count, the line search oracle parameters, the ball optimization oracle parameters, and the proximal gradient oracle parameters. We note that Proposition 2 in [ACJ+21] states that they obtain function error ϵ_{opt} with constant probability; however, examining the proof shows it actually yields an expected error bound. Additionally, Proposition 2 in [ACJ+21] is stated for $x^\star$ (the comparison point in the error guarantee) defined to be the minimizer of F, but examining the proof shows that the only property about $x^\star$ it uses is that $x^\star \in \mathbb{B}(R)$.

a) Iteration count.: The bound $C_{\text{ba}} K \log \kappa$ on the number of iterations follows immediately from the value $K_{\max}$ stated in Proposition 2 of [ACJ+21], where we set $\lambda_{\min} \leftarrow \lambda_\star$ and $\epsilon \leftarrow \epsilon_{\text{opt}}$.

b) Line search oracle parameters.: The line search oracle is called in the implementation of Line 2 of Algorithm 4 in [ACJ+21]. Our implementation follows the development of Appendix D.2.3 in [ACJ+21], which is a restatement of Proposition 2 in [CJJS21]. The bound $C_{\text{ba}} \log(\frac{R\kappa}{r})$ on the number of calls to the oracle is immediate from the statement of Proposition 2. For the oracle parameter $\Delta = \frac{r}{C_{\text{ba}}}$, we note that the proof of Proposition 2 of [CJJS21] only requires that we obtain points at distance at most $\frac{r}{17}$ from $x^\star_{\widehat{x},\lambda}$, although it is stated as requiring a function error guarantee. This is evident where the proof applies Lemma 3 of the same paper.

c) Ball optimization oracle parameters.: The ball optimization oracle is called in the implementation of Line 5 of Algorithm 4 in [ACJ+21]. In iteration k of the algorithm, the error requirement is derived through the potential bound in Lemma 5 of [ACJ+21]. More precisely, Lemma 5 shows that (following their notation), conditioned on all randomness through iteration k,

$$\begin{aligned} &\mathbb{E}\left[A_{k+1}\left(F(x_{k+1}) - F(x^\star)\right) + \left\|v_{k+1} - x^\star\right\|^2\right] \\ &\quad - \left(A_k\left(F(x_k) - F(x^\star)\right) + \left\|v_k - x^\star\right\|^2\right) \\ &\le -\frac{1}{6}\lambda_{k+1} A_{k+1} \left\|\widehat{x}_{k+1} - y_k\right\|^2 + A_{k+1}\phi_{k+1} \\ &\quad + a_{k+1}^2 \sigma_{k+1}^2 + 2R a_{k+1}\delta_{k+1}, \end{aligned}$$

where the terms $a_{k+1}^2\sigma_{k+1}^2 + 2Ra_{k+1}\delta_{k+1}$ are handled identically in [ACJ+21] and our Proposition 1 (see the following discussion). For the remaining two terms, Proposition 4 of [ACJ+21] guarantees that as long as the method does not terminate, one of the following occurs.

1) $\left\|\widehat{x}_{k+1} - y_k\right\|^2 = \Omega(r^2)$.
2) $\lambda_{k+1} = O(\lambda_\star)$.

In the first case, as long as ϕ_{k+1} (the error tolerance to the ball optimization oracle) is set to be $\frac{\lambda_{k+1} r^2}{C_{\text{ba}}}$ for a sufficiently large C_{ba} (which it is smaller than by logarithmic factors), up to constant factors the potential proof is unaffected. The total contributions to the potential due to all $A_{k+1}\phi_{k+1}$ losses from the iterations of the second case across the entire algorithm is bounded by

$$O\left((K \log \kappa) \cdot \frac{R^2}{\epsilon_{\text{opt}}} \cdot \frac{\lambda_\star r^2}{\log^3 \kappa}\right) = O\left(R^2\right).$$

Here, the first term is the iteration count, the second term is due to an upper bound on A_{k+1}, and the third term is bounded since $\lambda_{k+1} = O(\lambda_\star)$. The initial potential in the proof of Proposition 2 of [ACJ+21] is R^2, so the final potential is unaffected by more than constant factors. For a more formal derivation of the same improved error tolerance, we refer the reader to [CH22], Lemma 8.

d) Stochastic proximal oracle parameters.: Our stochastic proximal oracle parameters are exactly the settings of δ_k, σ_k required by Proposition 2 of [ACJ+21], except we simplified the bound on $\sigma_k^2 = O(\frac{\epsilon}{a_k})$ (note we use ϵ_{opt} in place of ϵ). In particular, following notation of [ACJ+21], we have

$$\frac{\epsilon}{a_k} = \frac{\epsilon\sqrt{\lambda_k}}{\sqrt{A_k}} = \Omega\left(\epsilon \cdot \sqrt{\lambda_\star} \cdot \frac{\sqrt{\epsilon}}{R}\right) = \Omega\left(\frac{\epsilon^2 K}{R^2} \log \kappa\right).$$

The first equality used $\lambda_k a_k^2 = A_k$ for the parameter choices of Algorithm 4 in [ACJ+21]. The second equality used that all $\lambda_k = \Omega(\lambda_\star)$ and all $A_k = O(\frac{R^2}{\epsilon})$ in Algorithm 4 in [ACJ+21], where we chose $\lambda_\star = \frac{\epsilon K^2}{R^2}\log^2 \kappa$. The final equality plugged in this bound on $\lambda_\star$ and simplified. Hence, obtaining a variance as declared in Proposition 1 suffices to meet the requirement.

In this section, we discuss how to obtain Proposition 2 (which is based on Proposition 1 in [CH22]) from the analysis in [CH22]. The iteration count discussion is the same as in Appendix A. We separate the discussion into parts corresponding to the two requirements in Proposition 2. Throughout, we will show how to use the analysis in [CH22] to guarantee that with probability at least $1 - \Omega(\frac{1}{\kappa})$, the algorithm has expected

function error $O(\epsilon_{\text{opt}})$; because the maximum error over $\mathbb{B}(R)$ is $\leq LR$, this corresponds to an overall error $O(\epsilon_{\text{opt}})$, and we may adjust C_{ba} by a constant to compensate.

e) Per-iteration requirements.: The ball optimization error guarantees are as stated in Proposition 1 of [CH22], except we dropped the function evaluations requirement. To see that this is obtainable, note that [CH22] obtains their line search oracle (see Proposition 1) by running $O(\log(\frac{R\kappa}{r}))$ ball optimization oracles to $O(\lambda r^2)$ expected error, querying the function value, and applying Markov's inequality to argue at least one will succeed with high probability. We instead execute $O(\log(\frac{R\kappa}{r}))$ independent runs and apply a Chernoff bound to argue that with probability $O(\frac{1}{K\kappa\cdot\text{polylog}(K\kappa)})$, the preconditions of Aggregate in Lemma 11 are met with $\Delta = O(r)$, as required by the line search oracle (see Algorithm 7). Finally, applying a union bound over all iterations implies that the overall failure probability due to these line search oracles is $O(\frac{1}{\kappa})$ as required by our earlier argument.

f) Additional requirements.: The error requirements of the queries which occur every $\approx 2^{-j}$ iterations are as stated in [CH22]. The only difference is that we state the complexity deterministically (Proposition 1 of [CH22] implicitly states an expected gradient bound). The stochastic proximal oracle is implemented as Algorithm 2, [CH22]; it is also adapted with slightly different parameters as Algorithm 6 of this paper. The expected complexity bound is derived by summing over all $j \in [\lceil \log_2 K + C_{\text{ba}} \rceil]$, the probability j is sampled in each iteration of Algorithm 2 of [CH22]. For all j a Chernoff bound shows that the number of times in the entire algorithm j is sampled is $O(2^{-j}K\log(\frac{R\kappa}{r}))$ (within a constant of its expectation), with probability $1-\Omega(\text{poly}(\frac{r}{R\kappa}))$. Taking a union bound over all j shows the failure probability of our complexity bound is $O(\frac{1}{\kappa})$ as required.

In this section, we discuss how to obtain Proposition 4 using results in [GL12]. We first state the following helper fact on the smoothness of a convolved function $\widehat{f}_\rho$ (see Definition 1).

Fact 5 (Lemma 8, [BJL+19]). *If $f : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz, $\widehat{f}_\rho$ (see Definition 1) is $\frac{L}{\rho}$-smooth.*

The statement of Proposition 4 then follows from recursively applying Proposition 9 of [GL12] on the objective $\Psi = \widehat{f}_\rho + \frac{\lambda}{2}\|\cdot - \bar{x}\|^2$, which is λ-strongly convex and $(\frac{L}{\rho}+\lambda)$-smooth, together with the divergence choice of $V(x_0, x^*) := \frac{1}{2}\|x_0 - x^*\|^2$, which satisfies $\nu = 1$. Our parameter choices in Algorithm 2 are the same as in [GL12], where we use that our variance bound is $3L^2$ (Lemma 1).

In particular, denote the iterate x_T^{ag} after the k^{th} outer loop by x^k. We will inductively assume that $\mathbb{E}\frac{1}{2}\|x^{k-1} - x^\star_{\bar{x},\lambda}\|^2 \leq \frac{r^2}{2^{k-1}}$ (clearly the base case holds). This then implies

$$\begin{aligned}\mathbb{E}\left[\frac{\lambda}{2}\|x^k - x^\star_{\bar{x},\lambda}\|^2\right] &\leq \mathbb{E}\left[\Psi(x^k) - \Psi(x^\star_{\bar{x},\lambda})\right] \\ &\leq \frac{2(\frac{L}{\rho}+\lambda)\|x^{k-1} - x^\star_{\bar{x},\lambda}\|^2}{T(T+1)} \\ &\quad + \frac{24L^2}{\lambda N_k(T+1)} \\ &\leq \frac{\lambda}{2^k}r^2\end{aligned}$$

where the second inequality is Proposition 9 in [GL12] (cf. equation (4.21) therein), and the last is by our choice of T and N_k. Thus, when $K > \log_2(\frac{\lambda r^2}{\phi})$ we have $\mathbb{E}\Psi(x_T^{\mathsf{ag}}) - \Psi(x^\star_{\bar{x},\lambda}) \leq \phi$ as in the last outer loop $k = K$. The computational depth follows immediately from computing TK, and the total oracle queries and computational complexity follow since N_K asymptotically dominates:

$$\begin{aligned}T\cdot\left(\sum_{k\in[K]} N_k\right) &= O\left(TN_K + TK\right) \\ &= O\left(\sqrt{1+\frac{L}{\rho\lambda}}\log\left(\frac{\lambda r^2}{\phi}\right) + \frac{L^2}{\lambda\phi}\right).\end{aligned}$$