Stochastic Bias-Reduced Gradient Methods

Hilal Asi* **Yair Carmon†** **Arun Jambulapati*** **Yujia Jin*** **Aaron Sidford***

Abstract

We develop a new primitive for stochastic optimization: a low-bias, low-cost estimator of the minimizer $x_\star$ of any Lipschitz strongly-convex function. In particular, we use a multilevel Monte-Carlo approach due to Blanchet and Glynn [8] to turn any optimal stochastic gradient method into an estimator of $x_\star$ with bias δ, variance $O(\log(1/\delta))$, and an expected sampling cost of $O(\log(1/\delta))$ stochastic gradient evaluations. As an immediate consequence, we obtain cheap and nearly unbiased gradient estimators for the Moreau-Yoshida envelope of any Lipschitz convex function, allowing us to perform dimension-free randomized smoothing. We demonstrate the potential of our estimator through four applications. First, we develop a method for minimizing the maximum of N functions, improving on recent results and matching a lower bound up to logarithmic factors. Second and third, we recover state-of-the-art rates for projection-efficient and gradient-efficient optimization using simple algorithms with a transparent analysis. Finally, we show that an improved version of our estimator would yield a nearly linear-time, optimal-utility, differentially-private non-smooth stochastic optimization method.

1 Introduction

Consider the fundamental problem of minimizing a μ-strongly convex function $F : \mathcal{X} \to \mathbb{R}$ given access to a stochastic (sub-)gradient estimator $\hat{\nabla} F$ satisfying $\mathbb{E}\,\hat{\nabla} F(x) \in \partial F(x)$ and $\mathbb{E}\|\hat{\nabla} F(x)\|^2 \le G^2$ for every $x \in \mathcal{X}$. Is it possible to transform the unbiased estimator $\hat{\nabla} F$ into a (nearly) unbiased estimator of the minimizer $x_\star \coloneqq \operatorname{argmin}_{x\in\mathcal{X}} F(x)$? In particular, can we improve upon the $O(G/(\mu\sqrt{T}))$ bias achieved by T iterations of stochastic gradient descent (SGD)?

In this paper, we answer this question in the affirmative, proposing an *optimum estimator* $\hat{x}_\star$, which (for any fixed $\delta > 0$) has

$$\text{bias } \|\mathbb{E}\hat{x}_\star - x_\star\| = O(\delta) \text{ and variance } \mathbb{E}\|\hat{x}_\star - \mathbb{E}\hat{x}_\star\|^2 = O\left(\frac{G^2}{\mu^2}\log\left(\frac{G}{\mu\delta}\right)\right),$$

and, *in expectation*, costs $O(\log(\frac{G}{\mu\delta}))$ evaluations of $\hat{\nabla} F$.[3] Setting $\delta = G/(\mu\sqrt{T})$, we obtain the same bias bound as T iterations of SGD, but with expected cost of only $O(\log T)$ stochastic gradient evaluations (the worst-case cost is T). Further, the bias can be made arbitrarily small with only logarithmic increase in the variance and the stochastic gradient evaluations of our estimator, and therefore—paralleling the term "nearly linear-time" [27]—we call $\hat{x}_\star$ *nearly unbiased*.

Our estimator is an instance of the multilevel Monte Carlo technique for de-biasing estimator sequences [25] and more specifically the method of Blanchet and Glynn [8]. Our key observation is that this method is readily applicable to strongly-convex variants of SGD, or indeed any stochastic optimization method with the same (optimal) rate of convergence.

*Stanford University, {asi,jmblpati,yujiajin,sidford}@stanford.edu

†Tel Aviv University, ycarmon@tauex.tau.ac.il

[3] When $\mathcal{X} = \mathbb{B}_R(x_0) \subset \mathbb{R}^d$, $F(x) = \frac{1}{n}\sum_{i\in[n]} \hat{F}(x;i)$, and $\hat{\nabla} F$ is the subgradient of a uniformly random $\hat{F}(x;i)$ we can also get an estimator with bias 0 and expected cost $O(\log(nd))$. See Appendix A.1 for details.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

Objective	Expensive operation $\mathcal{O}$	$\mathcal{N}_{\mathcal{O}}$	$\mathbb{E}\mathcal{N}_{\hat{\nabla} f}$
$\max_{i\in[N]} f_{(i)}(x)$ (Sec. 4)	$f_{(1)}(x), \ldots, f_{(N)}(x)$	$\widetilde{O}(\epsilon^{-2/3})$	$\widetilde{O}(\epsilon^{-2})$
$f(x)$ in domain $\mathcal{X}$ (Sec. 3)	$\mathsf{Proj}_{\mathcal{X}}(x)$	$O(\epsilon^{-1})$	"
$\Lambda(x) + f(x)$ for L-smooth Λ (Sec. 5)	$\nabla\Lambda(x)$	$O\big(\sqrt{L/\epsilon}\big)$	"

Table 1. Summary of our applications of accelerated bias-reduced stochastic gradient methods. We use $\mathcal{N}_{\mathcal{O}}$ and $\mathcal{N}_{\hat{\nabla} f}$ to denote the number of expensive operations and subgradient estimations, respectively. The $\widetilde{O}$ notation hides polylogarithmic factors. See Section 1.2 for additional description.

1.1 Estimating proximal points and Moreau-Yoshida envelope gradients

Given a convex function f and regularization level λ, the proximal point of y is $\mathsf{P}_{f,\lambda}(y) \coloneqq \operatorname{argmin}_{x\in\mathbb{R}^d}\big\{f(x) + \frac{\lambda}{2}\|x-y\|^2\big\}$. Since computing $\mathsf{P}_{f,\lambda}$ amounts to solving a λ-strongly-convex problem, our technique provides low-bias and cheap proximal point estimators. Proximal points are ubiquitous in optimization [43, 19, 52, 38] and estimating them efficiently with low bias opens up new algorithmic possibilities. One of these possibilities is estimating the gradient of the Moreau-Yoshida envelope $f_\lambda(y) \coloneqq \min_{x\in\mathbb{R}^d}\big\{f(x) + \frac{\lambda}{2}\|x-y\|^2\big\}$, which is a λ-smooth, $G^2/(2\lambda)$-accurate approximation of any G-Lipschitz f (see, e.g., [43, 29] and Appendix B.3). Since $\nabla f_\lambda(y) = \lambda(y - \mathsf{P}_{f,\lambda}(y))$, our optimum estimator provides a low-bias estimator for $\nabla f_\lambda(y)$ with second moment and expected cost greater than those of $\hat{\nabla} f$ by only a logarithmic factor. Thus, for any non-smooth f we can turn $\hat{\nabla} f$ into a gradient estimator for the smooth surrogate f_λ, whose smoothness is independent of the problem dimension, allowing us to perform *dimension-free* randomized smoothing [20].

1.2 Applications via accelerated bias-reduced methods

Our optimum estimator is a new primitive in stochastic convex optimization and we expect it to find multiple applications. We now describe three such applications: the first improves on previously known complexity bounds while the latter two recover existing bounds straightforwardly. For simplicity of presentation we assume (in the introduction only) $\mathbb{E}\|\hat{\nabla} f\|^2 \le 1$ and unit domain size.

In each application, we wish to minimize an objective function given access to a cheap subgradient estimator $\hat{\nabla} f$ as well as an expensive application-specific operation $\mathcal{O}$ (e.g., a projection to a complicated set). Direct use of the standard stochastic gradient method finds an ϵ-accurate solution using $O(\epsilon^{-2})$ computations of both $\hat{\nabla} f$ and $\mathcal{O}$, and our goal is to improve the $\mathcal{O}$ complexity without hurting the $\hat{\nabla} f$ complexity.

To that end, we design stochastic accelerated methods consisting of T iterations, each one involving only a constant number of $\mathcal{O}$ and proximal point computations, which we approximate by averaging copies of our optimum estimator.[4] Its low bias allows us to bound $T \ll \epsilon^{-2}$ as though our proximal points were exact, while maintaining an $\widetilde{O}(\epsilon^{-2})$ bound on the total expected number of $\hat{\nabla} f$ calls.[5] Thus, we save expensive operations without substantially increasing the gradient estimation cost. Table 1 summarizes each application, and we briefly describe them below.

Minimizing the maximal loss (Section 4). Given N convex, 1-Lipschitz functions $f_{(1)}, \ldots, f_{(N)}$ we would like to find an ϵ-approximate minimizer of their maximum $f_{\max}(x) = \max_{i\in[N]} f_{(i)}(x)$. This problem naturally arises when optimizing worst-case behavior, as in maximum margin classification and robust optimization [53, 15, 45, 6]. We measure complexity by the number of individual function and subgradient evaluations, so that the expensive operation of evaluating $f_{(1)}, \ldots, f_{(N)}$ at a single point has complexity $O(N)$ and the subgradient method solves this problem with complexity $O(N\epsilon^{-2})$. Carmon et al. [13] develop an algorithm for minimizing $f_{\max}$ with complexity $\widetilde{O}(N\epsilon^{-2/3} + \epsilon^{-8/3})$, improving on the subgradient method for sufficiently large N. Using our bias-reduced Moreau gradient envelope estimator in a Monteiro-Svaiter-type accelerated proximal point method [12, 11, 38], we obtain improved complexity $\widetilde{O}(N\epsilon^{-2/3} + \epsilon^{-2})$. This matches (up to logarithmic factors) a lower bound shown in [13], settling the complexity of minimizing the maximum

[4] While averaging is parallelizable, our optimum estimator itself is sequential. Consequently, our approach does not yield improve parallelism; see Appendix A.2 for further discussion

[5] It is easy to turn expected complexity bounds into deterministic ones; see Appendix A.3.

of N non-smooth functions. Our result reveals a surprising fact: for $N \ll (GR/\epsilon)^{-4/3}$, minimizing the maximum of N functions is no harder than minimizing their average.

Projection-efficient optimization via dimension-free randomized smoothing (Section 3). Consider the problem of minimizing a convex function f using an unbiased gradient estimator $\hat{\nabla} f$ over convex set $\mathcal{X}$ for which Euclidean projections are expensive to compute (for example, the cone of PSD matrices). When f is L-smooth, a stochastic version of Nesterov's accelerated gradient descent (AGD) [16] performs only $O(\sqrt{L/\epsilon})$ projections. For non-smooth f we instead apply AGD to the Moreau envelope smoothing of f (with appropriate $\lambda = O(\epsilon^{-1})$) using our nearly-unbiased stochastic estimator for ∇f_λ. This yields a solution in $O(\epsilon^{-1})$ projections and $\widetilde{O}(\epsilon^{-2})$ evaluations of $\hat{\nabla} f$. Our algorithm provides a simple alternative to the recent work of Thekumparampil et al. [51] whose performance guarantees are identical up to a logarithmic factor.

Gradient-efficient composite optimization (Section 5). We would like to minimize $\Psi(x) = \Lambda(x) + f(x)$, where Λ is convex and L-smooth but we can access it only via computing (expensive) exact gradients, while f is a non-smooth convex functions for which we have a (cheap) unbiased subgradient estimator $\hat{\nabla} f$. Problems of this type include inverse problems with sparsity constraints and regularized loss minimization in machine learning [34]. To save $\nabla\Lambda$ computations, it is possible to use composite AGD [41] which solves $O(\sqrt{L/\epsilon})$ subproblems of the form $\mathrm{minimize}_x \{\langle \nabla\Lambda(y), x\rangle + f(x) + \frac{\beta}{2}\|x - x'\|^2\}$. Lan [34] designed a specialized method, gradient sliding, for which the total subproblem solution cost is $O(\epsilon^{-2})$ evaluations of $\hat{\nabla} f$. We show that a simple alternative—estimating the subproblem solutions via our low-bias optimum estimator—recovers its guarantees up to logarithmic factors.

1.3 Non-smooth differentially private stochastic convex optimization

We now discuss a potential application of our technique that is conditional on the existence of an improved optimum estimator. In it, we minimize the population objective function $f(x) = \mathbb{E}_{S\sim P}\hat{f}(x; S)$ under the well-known constraint of differential privacy [22]. Given n i.i.d. samples $S_i \sim P$ and assuming that each $\hat{f}$ is 1-Lipschitz, convex and sufficiently smooth, Feldman et al. [23] develop algorithms that obtain the optimal error and compute $O(n)$ subgradients of $\hat{f}$. The non-smooth case is more challenging and the best existing bound is $O(n^{11/8})$ for the high-dimensional setting $d = n$ [32, 3]. In Section 6 we show that our optimum estimator, combined with recent localization techniques [23], reduces the problem to private mean estimation. Unfortunately, our estimator is heavy-tailed, leading to insufficient utility. Nevertheless, assuming a version of our estimator that has bounded outputs, we give an algorithm that queries $\widetilde{O}(n)$ subgradients for non-smooth functions, solving a longstanding open problem in private optimization [14, 4]. This motivates the study of improved versions of our estimators that have constant sensitivity.

1.4 Related work

Multilevel Monte-Carlo (MLMC) techniques originate from the literature on parametric integration for solving integral and differential equations [25]. Our approach is based on an MLMC variant put forth by Blanchet and Glynn [8] for estimating functionals of expectations. Among several applications, they propose [8, Section 5.2] an estimator for $\mathrm{argmin}_x \mathbb{E}_{S\sim P}\hat{f}(x; S)$ where $\hat{f}(\cdot; s)$ is convex for all s and assuming access to minimizers of empirical objectives of the form $\sum_{i\in[N]} \hat{f}(x; s_i)$. The authors provide a preliminary analysis of the estimator's variance (later elaborated in [9]) using an asymptotic Taylor expansion around the population minimizer. In comparison, we study the more general setting of stochastic gradient estimators and provide a complete algorithm based on SGD, along with a non-asymptotic analysis and concrete settings where our estimator is beneficial.

A number of works have used the Blanchet-Glynn estimator in the context of optimization and machine learning. These applications include estimating the ratio of expectations for semisupervised learning [7], estimating gradients of distributionally robust optimization objectives [35], and estimating gradients in deep latent variable models [47]. Our estimator is similar to that of Levy et al. [35] in that we also have to pick a "critical" doubling probability for the (random) computational budget, which makes the expected cost and variance of our estimators depend logarithmically on the bias.

1.5 Limitations

Our paper demonstrates that our proposed optimum estimator is a useful *proof device*: it allows us to easily prove upper bounds on the complexity of structured optimization problems, and at least in one case (minimizing the maximum loss) improve over previously known bounds. However, our work does not investigate the *practicality* of our optimum estimator, as implementation and experiments are outside its scope.

Nevertheless, let us briefly discuss the practical prospects of the algorithms we propose. On the one hand, our optimum estimator itself is fairly easy to implement, adding only a few parameters on top of a basic gradient method. On the other hand, in the settings of Sections 3 and 5, gradient-sliding based methods [34, 51] are roughly as simple to implement and enjoy slightly stronger convergence bounds (better by logarithmic factors) than our optimum estimator. Consequently, in these settings we have no reason to assume that our algorithms are better in practice. In the setting of Section 4 (minimizing the maximum loss) our algorithm does enjoy a stronger guarantee than the previous best bound [13]. However, both our algorithm and [13] are based on an accelerated proximal point method that, in its current form, is not practical [13, Sec. 6.2]. Thus, evaluating the benefit of stochastic bias reduction in the context of minimizing the maximum loss would require us to first develop a practical accelerated proximal point algorithm, which is an open question under active research [see, e.g., 50].

Another limitation of our optimum estimator is that, while it has a bounded second moment, its higher moments are unbounded. While this does not matter for most of our results, the lack of higher moment bounds prevents us from setting the complexity of non-smooth private stochastic convex optimization in Section 6. Finding an optimum estimator that is bounded with high probability—or proving that one does not exist—remains an open question for future work.

Finally, our analyses are limited to convex objective functions. However, while outside the scope of the paper, we believe our results are possibly relevant for non-convex settings as well. In particular, for smooth non-convex functions (and weakly-convex functions [17] more broadly) the problem of computing proximal points with sufficiently high regularization is strongly convex and our estimator applies. Such non-convex proximal points play an important role in non-convex optimization [17] with applications in deep learning [see, e.g., 49]. Applying the optimum-estimator technique in non-convex optimization is therefore a viable direction for future work.

1.6 Notation

We let $\mathbb{B}_R(x) = \{y \in \mathbb{R}^d : \|y - x\| \le R\}$ denote the ball of radius R around x, where $\|\cdot\|$ is the Euclidean norm throughout. We write $\mathsf{Proj}_{\mathcal{S}}$ for the Euclidean projection to $\mathcal{S}$. We write $\mathbb{1}_{\{A\}}$ for the indicator of event A, i.e., $\mathbb{1}_{\{A\}} = 1$ when A holds and 0 otherwise. Throughout the paper, $\hat{\nabla} f$ denotes a (stochastic) subgradient estimator for the function f, and $\mathcal{X} \subset \mathbb{R}^d$ denotes the optimization domain, which we always assume is closed and convex. We use $\mathsf{P}_{f,\lambda}$ to denote the proximal operator (2) and f_λ to denote the Moreau envelope (3) associated with function f and regularization parameter λ. Finally, we use $\mathcal{N}_f$ and $\mathcal{N}_{\hat{\nabla} f}$ to denote function and subgradient estimator evaluation complexity, respectively.

2 A multilevel Monte-Carlo optimum estimator

In this section, we construct a low-bias estimator for the minimizer of any strongly convex function $F : \mathcal{X} \to \mathbb{R}$. This estimator is the key component of our algorithms in the subsequent sections, which use it to approximate proximal points and Moreau envelope gradients. We assume that F is of the form $F = f + \psi$, where the function ψ is "simple" and that f satisfies the following.

Assumption 1. *The function $f : \mathcal{X} \to \mathbb{R}$ is convex (with closed and convex domain $\mathcal{X}$) and is accessible via an unbiased subgradient estimator $\hat{\nabla} f$ which satisfies $\mathbb{E}\|\hat{\nabla} f(x)\|^2 \le G^2$ for all x.*

Our applications only use ψ of the form $\psi(x) = \frac{\lambda}{2}\|x - x'\|^2$ but our estimator applies more broadly to cases where $\operatorname{argmin}_x \left\{ \langle v, x\rangle + \psi(x) + \frac{1}{2\eta}\|x - y\|^2 \right\}$ is easy to compute for all v and y.

2.1 ODC algorithms

Our estimator can use, in a black-box fashion, any method for minimizing F with sufficiently fast convergence to $x_\star = \operatorname{argmin}_{x \in \mathcal{X}} F(x)$. We highlight the required convergence property as follows.

Algorithm 1: OPTEST$(\hat{\nabla} f, \psi, \mu, \delta, \sigma^2, \mathcal{X})$

▷ δ, σ^2 are required bias and square error
▷ c is the ODC algorithm constant
$T_{\max} = \left\lceil \frac{4cG^2}{\mu^2 \min\{\delta^2, \frac{1}{2}\sigma^2\}} \right\rceil$
$N = \left\lceil \frac{32cG^2 \log(T_{\max})}{\mu^2 \sigma^2} \right\rceil$
for $i = 1, \ldots, N$ **do**
  $\hat{x}_\star^{(i)}$ = a draw of the estimator (1)
return $\frac{1}{N} \sum_{i \in [N]} \hat{x}_\star^{(i)}$

Algorithm 2: MORGRADEST$(\hat{\nabla} f, y, \lambda, \delta, \sigma^2, \mathcal{X})$

▷ δ, σ^2 are required bias and square error
▷ λ is the regularization level
▷ y is the point at which to estimate $\nabla f_\lambda(y)$
$\psi_\lambda(x) = \frac{\lambda}{2}\|x - y\|^2$
$\hat{x}_\star = \text{OPTEST}\Big(\hat{\nabla} f, \psi_\lambda, \lambda, \frac{\delta}{\lambda}, \frac{\sigma^2}{\lambda^2}, \mathcal{X}\Big)$
return $\lambda(y - \hat{x}_\star)$

Definition 1. *An* optimal-distance-convergence *algorithm* ODC *takes as input $\hat{\nabla} f$ satisfying Assumption 1, a simple function ψ and a budget $T \geq 1$. If $F = f + \psi$ is μ-strongly convex with minimizer $x_\star$, the algorithm's output $x = \text{ODC}(\hat{\nabla} f, \psi, T)$ requires at most T evaluations of $\hat{\nabla} f$ to compute and satisfies $\mathbb{E}\|x - x_\star\|^2 \leq c\frac{G^2}{\mu^2 T}$ for some constant $c > 0$.*

Standard lower bound constructions imply that the $O(\frac{G^2}{\mu^2 T})$ squared distance convergence rate is indeed optimal; see Appendix A.4 for additional discussion. Conversely, ODC algorithms are readily available in the literature [44, 28] since any point x satisfying $\mathbb{E}F(x) - F(x_\star) = O(\frac{G^2}{\mu T})$ (the optimal rate of convergence in strongly convex, Lipschitz optimization) also satisfies $\mathbb{E}\|x - x_\star\|^2 \leq O(\frac{G^2}{\mu^2 T})$ by due to the strong convexity of F. We provide a concrete ODC algorithm consisting of a generalization of epoch SGD [28], which allows us to optimize over the composite objective $F = f + \psi$ instead of only f as in the prior study of epoch SGD.

Lemma 1. EPOCHSGD *(Algorithm 8 in Appendix B.1) is an ODC algorithm with constant $c = 32$.*

2.2 Constructing an optimum estimator

To turn any ODC algorithm into a low-bias, low-cost and near-constant variance optimum estimator, we use the multilevel Monte Carlo (MLMC) technique of Blanchet and Glynn [8]. Given a problem instance $\hat{\nabla} f, \psi$, an algorithm ODC and a cutoff parameter $T_{\max} \in \mathbb{N}$, our estimator $\hat{x}_\star$ is:

$$\text{Draw } J \sim \mathsf{Geom}\left(\tfrac{1}{2}\right) \in \mathbb{N} \text{ and, writing } x_j \coloneqq \text{ODC}(\hat{\nabla} f, \psi, 2^j), \text{ compute}$$
$$\hat{x}_\star = x_0 + \begin{cases} 2^J(x_J - x_{J-1}) & 2^J \leq T_{\max} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

We note that for certain ODC algorithms it is possible to extract x_0, x_{J-1} from the intermediate steps of computing x_J, so that we only need to invoke ODC once. This is particularly simple to do for EPOCHSGD, as we explain in Appendix B.1. The key properties of our estimator are as follows.

Proposition 1. *Let f and $\hat{\nabla} f$ satisfy Assumption 1, $F = f + \psi$ be μ-strongly convex with minimizer $x_\star$ and $T_{\max} \in \mathbb{N}$. For any ODC algorithm with constant c, the estimator* (1) *has bias $\|\mathbb{E}\hat{x}_\star - x_\star\| \leq \sqrt{2c}\frac{G}{\mu\sqrt{T_{\max}}}$ and variance $\mathbb{E}\|\hat{x}_\star - \mathbb{E}\hat{x}_\star\|^2 \leq 16c\frac{G^2}{\mu^2}\log_2(T_{\max})$. Moreover, the expected number of $\hat{\nabla} f$ evaluations required to compute $\hat{x}_\star$ is $O(\log T_{\max})$.*

Proof. Let $j_{\max} = \max\{j \in \mathbb{N} \mid 2^j \leq T_{\max}\} = \lfloor \log_2 T_{\max} \rfloor$. The expectation of $\hat{x}_\star$ is

$$\mathbb{E}\hat{x}_\star = \mathbb{E}x_0 + \sum_{j=1}^{j_{\max}} \mathbb{P}(J = j) 2^j (\mathbb{E}x_j - \mathbb{E}x_{j-1}) = \mathbb{E}x_{j_{\max}},$$

where the second equality follows from $\mathbb{P}(J = j) = 2^{-j}$ and the sum telescoping. Noting that $x_{j_{\max}} = \text{ODC}(\hat{\nabla} f, \psi, T)$ for $T = 2^{j_{\max}} \geq T_{\max}/2$, we have that

$$\|\mathbb{E}x_{j_{\max}} - x_\star\| \leq \sqrt{\mathbb{E}\|x_{j_{\max}} - x_\star\|^2} \leq \sqrt{c}\frac{G}{\mu\sqrt{T_{\max}/2}}$$

by Definition 1. To bound the variance we use $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$ and note that

$$\mathbb{E}\|\hat{x}_\star - \mathbb{E}\hat{x}_\star\|^2 \le \mathbb{E}\|\hat{x}_\star - x_\star\|^2 \le 2\mathbb{E}\|\hat{x}_\star - x_0\|^2 + 2\mathbb{E}\|x_0 - x_\star\|^2.$$

The ODC property implies that $\mathbb{E}\|x_0 - x_\star\|^2 \le cG^2/\mu^2$. For the term $\mathbb{E}\|\hat{x}_\star - x_0\|^2$ we have

$$\mathbb{E}\|\hat{x}_\star - x_0\|^2 = \sum_{j=1}^{j_{\max}} \mathbb{P}(J=j)2^{2j}\mathbb{E}\|x_j - x_{j-1}\|^2 = \sum_{j=1}^{j_{\max}} 2^j \mathbb{E}\|x_j - x_{j-1}\|^2, \text{ and}$$

$$\mathbb{E}\|x_j - x_{j-1}\|^2 \le 2\mathbb{E}\|x_j - x_\star\|^2 + 2\mathbb{E}\|x_{j-1} - x_\star\|^2 \le 6c\frac{G^2}{\mu^2}2^{-j}.$$

Substituting, we get $\mathbb{E}\|\hat{x}_\star - x_0\|^2 \le 6c\frac{G^2}{\mu^2} j_{\max}$ and $\mathbb{E}\|\hat{x}_\star - \mathbb{E}\hat{x}_\star\|^2 \le 16c\frac{G^2}{\mu^2}\log_2(T_{\max})$. Finally, the expected number of $\hat{\nabla} f$ evaluations is $1 + \sum_{j=1}^{j_{\max}} \mathbb{P}(J=j)(2^j + 2^{j-1}) = O(j_{\max})$. □

The function OPTEST in Algorithm 1 computes an estimate of $x_\star$ with and desired bias δ and square error σ^2 by averaging independent draws of the MLMC estimator (1). The following guarantees are immediate from Proposition 1; see Appendix B.2 for a short proof.

Theorem 1. *Let f and $\hat{\nabla} f$ satisfy Assumption 1, $F = f + \psi$ be μ-strongly convex with minimizer $x_\star \in \mathcal{X}$, and $\delta, \sigma > 0$. The function* OPTEST$(\hat{\nabla} f, \psi, \mu, \delta, \sigma^2, \mathcal{X})$ *outputs $\hat{x}_\star$ satisfying*

$$\|\mathbb{E}\hat{x}_\star - x_\star\| \le \delta \quad \textit{and} \quad \mathbb{E}\|\hat{x}_\star - x_\star\|^2 \le \sigma^2$$

using $\mathcal{N}_{\hat{\nabla} f}$ stochastic gradient computations, where

$$\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\left(\frac{G^2}{\mu^2\sigma^2}\log^2\left(\frac{G}{\mu\min\{\delta,\sigma\}}\right) + \log\left(\frac{G}{\mu\min\{\delta,\sigma\}}\right)\right).$$

2.3 Estimating proximal points and Moreau envelope gradients

The proximal point of function $f : \mathcal{X} \to \mathbb{R}$ with regularization level λ at point y is

$$\mathsf{P}_{f,\lambda}(y) \coloneqq \operatorname*{argmin}_{x\in\mathcal{X}}\{f(x) + \tfrac{\lambda}{2}\|x-y\|^2\}. \tag{2}$$

When f satisfies Assumption 1, we may use OPTEST (with $\psi(x) = \frac{\lambda}{2}\|x-y\|^2$ and $\mu = \lambda$) to obtain a reduced-bias proximal point estimator. The proximal point $\mathsf{P}_{f,\lambda}(y)$ is closely related to the Moreau envelope

$$f_\lambda(y) \coloneqq \min_{x\in\mathcal{X}}\{f(x) + \tfrac{\lambda}{2}\|x-y\|^2\} \tag{3}$$

via the relationship $\nabla f_\lambda(y) = \lambda(y - \mathsf{P}_{f,\lambda}(y))$ (see Appendix B.3). Therefore, we can use our optimum estimator to turn $\widetilde{O}(1)$ calls to $\hat{\nabla} f$ into a nearly unbiased estimator for ∇f_λ. We formulate this as:

Corollary 2. *Let f and $\hat{\nabla} f$ satisfy Assumption 1, let $y \in \mathcal{X}$ and let $\lambda, \sigma, \delta > 0$. The function* MORGRADEST$(\hat{\nabla} f, \lambda, y, \delta, \sigma^2, \mathcal{X})$ *outputs $\hat{\nabla} f_\lambda(y)$ satisfying $\|\mathbb{E}\hat{\nabla} f_\lambda(y) - \nabla f_\lambda(y)\| \le \delta$ and $\mathbb{E}\|\hat{\nabla} f_\lambda(y) - \nabla f_\lambda(y)\|^2 \le \sigma^2$ and has complexity $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\left(\frac{G^2}{\sigma^2}\log^2\left(\frac{G}{\min\{\delta,\sigma\}}\right) + \log\left(\frac{G}{\min\{\delta,\sigma\}}\right)\right)$.*

3 Projection-efficient convex optimization

In this section, we combine the bias-reduced Moreau envelope gradient estimator with a standard accelerated gradient method to recover the result of Thekumparampil et al. [51]. We consider the problem of minimizing a function f satisfying Assumption 1 over the domain $\mathbb{B}_R(0)$ subject to the constraint $x \in \mathcal{X}$, where $\mathcal{X} \subset \mathbb{B}_R(0)$ is a complicated convex set that we can only access via (expensive) projections of the form $\mathsf{Proj}_{\mathcal{X}}(x) \coloneqq \operatorname{argmin}_{y\in\mathcal{X}}\|y - x\|$. We further assume that an initial point $x_0 \in \mathcal{X}$ satisfies $\|x_0 - x_\star\| \le D$.

Algorithm 3 applies a variant of Nesterov's accelerated gradient descent method (related to [2, 1]) on the (λ-smooth) Moreau envelope f_λ defined in eq. (3). Since computing the Moreau envelope

Algorithm 3: Stochastic accelerated gradient descent on the Moreau envelope

Input: A gradient estimator $\hat{\nabla} f$ satisfying Assumption 1 in $\mathbb{B}_R(0)$, projection oracle $\mathsf{Proj}_{\mathcal{X}}$, and initial point $x_0 = v_0$ with $\|x_0 - x_\star\| \le D$.

Parameters : Iteration budget T , Moreau regularization λ, approximation parameters δ_k, σ_k^2

for $k = 1, \cdots, T$ **do**

$y_{k-1} = \frac{k-1}{k+1} x_k + \frac{2}{k+1} v_{k-1}$

$g_k = \textsc{MorGradEst}(\hat{\nabla} f, y_{k-1}, \lambda, \delta_k, \sigma_k^2, \mathbb{B}_R(0))$

$x_k = \mathsf{Proj}_{\mathcal{X}}\left(y_{k-1} - \frac{1}{3\lambda} g_k\right)$

$v_k = \mathsf{Proj}_{\mathbb{B}_R(0)}\left(v_{k-1} - \frac{k}{6\lambda} g_k\right)$

return x_T

does not involve projection to $\mathcal{X}$, for sufficiently accurate approximation of ∇f_λ we require only $T = O(\sqrt{\lambda D^2/\epsilon})$ projections to $\mathcal{X}$ for finding an $O(\epsilon)$-suboptimal point of f_λ constrained to $\mathcal{X}$. For that point to be also ϵ-suboptimal for f itself, we must choose λ of the order of G^2/ϵ, so that the number of projections is $O(GD/\epsilon)$.

As noted in [51] computing ∇f_λ to accuracy $O(\epsilon/R)$ is sufficient for the above guarantee to hold, but doing so using a stochastic gradient method requires $O((GD/\epsilon)^2)$ evaluations of $\hat{\nabla} f$ per iteration, and $O((GD/\epsilon)^3)$ evaluations in total. To improve this, we employ Algorithm 2 to compute nearly-unbiased estimates for ∇f_λ and bound the error incurred by their variance. Our result matches the gradient sliding-based technique of Thekumparampil et al. [51] up to polylogarithmic factors while retaining the conceptual simplicity of directly applying AGD on the Moreau envelope. We formally state the guarantees of our method below, and provide a self-contained proof in Appendix C.

Theorem 3. *Let $f : \mathbb{B}_R(0) \to \mathbb{R}$ and $\hat{\nabla} f$ satisfy Assumption 1. Let $\mathcal{X} \subseteq \mathbb{B}_R(0)$ be a convex set admitting a projection oracle $\mathsf{Proj}_{\mathcal{X}}$. Let $x_0 \in \mathcal{X}$ be an initial point with $\|x - x_\star\| \le D$ for some $x_\star \in \mathcal{X}$. With $\lambda = \frac{2G^2}{\epsilon}$, $\delta_k = \frac{\epsilon}{8R}$, $\sigma_k^2 = \frac{2\epsilon\lambda}{k+1}$, and $T = \frac{7GD}{\epsilon}$ Algorithm 3 computes $x \in \mathcal{X}$ with $\mathbb{E}[f(x)] \le f(x_\star) + \epsilon$ with complexity $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\left(\frac{G^2 D^2}{\epsilon^2} \log^2\left(\frac{GR}{\epsilon}\right)\right)$ and $O\left(\frac{GD}{\epsilon}\right)$ calls to $\mathsf{Proj}_{\mathcal{X}}$.*

4 Accelerated proximal methods and minimizing the maximal loss

In this section we apply our estimator in an accelerated proximal point method and use it to obtain an optimal rate for minimizing the maximum of N convex functions (up to logarithmic factors).

4.1 Accelerated proximal point method via Moreau gradient estimation

Algorithm 4 is an Monteiro-Svaiter-type [38, 12] accelerated proximal point method [36, 24] that leverages our reduced-bias Moreau envelope gradient estimator. To explain the method, we contrast it with stochastic AGD on the Moreau envelope (Algorithm 3). First and foremost, Algorithm 3 provides a suboptimality bound on the Moreau envelope f_λ (which for small λ is far from f) while Algorithm 4 minimizes f itself.

Second, while Algorithm 3 uses a fixed regularization parameter λ, Algorithm 4 handles an arbitrary sequence $\{\lambda_k\}$ given by a black-box function NEXTLAMBDA. To facilitate our application of the method to minimizing the maximal loss—where gradient estimation is only tractable in small Euclidean balls around a reference point—we include an optional parameter r such that the proximal point movement bound $\|\mathsf{P}_{f,\lambda_{k+1}}(y_k) - y_k\| \le r$ holds for all k. However, most of our analysis of Algorithm 4 does not require this parameter (i.e., holding for $r = \infty$), making it potentially applicable to other settings that use accelerated proximal point methods [11, 38, 50].

The third and final notable difference between Algorithms 3 and 4 is the method of updating the x_k iteration sequence. While a projected stochastic gradient descent step suffices for Algorithm 3, here we require a more direct approximation of function value decrease attained by the exact proximal mapping $\mathsf{P}_{f,\lambda}$ (see eq. (2)). For a given accuracy φ, we define the φ-approximate proximal mapping

$$\widetilde{\mathsf{P}}^{\varphi}_{f,\lambda}(y) \coloneqq \text{ any } x \in \mathcal{X} \text{ such that } \mathbb{E}F(x) \le F(\mathsf{P}_{f,\lambda}(y)) + \varphi \text{ for } F(z) \coloneqq f(z) + \frac{\lambda}{2}\|z - y\|^2. \quad (4)$$

Algorithm 4: Stochastic accelerated proximal point method

Input: Gradient estimator $\hat{\nabla} f$, function NEXTLAMBDA, initialization $x_0 = v_0$ and $A_0 \geq 0$.
Parameters: Approximation parameters $\{\varphi_k, \delta_k, \sigma_k\}$, stopping parameters $A_{\max}$ and $K_{\max}$, optional movement bound $r > 0$.

1 **for** $k = 0, 1, \ldots$ **do**
2 $\quad \lambda_{k+1} = \text{NEXTLAMBDA}(x_k, v_k, A_k)$ ▷ guaranteeing that $\|\mathsf{P}_{f,\lambda_{k+1}}(y_k) - y_k\| \leq r$
3 $\quad a_{k+1} = \frac{1}{2\lambda_{k+1}} \sqrt{1 + 4\lambda_{k+1} A_k}$ and $A_{k+1} = A_k + a_{k+1}$ and $\mathcal{X}_k = \mathcal{X} \cap \mathbb{B}_r(y_k)$
4 $\quad y_k = \frac{A_k}{A_{k+1}} x_k + \frac{a_{k+1}}{A_{k+1}} v_k$ and $x_{k+1} = \widetilde{\mathsf{P}}^{\varphi_{k+1}}_{f,\lambda_{k+1}}(y_k)$ ▷ defined in eq. (4)
5 $\quad g_{k+1} = \text{MORGRADEST}(\hat{\nabla} f, y_k, \lambda_{k+1}, \delta_k, \sigma_k^2, \mathcal{X}_k)$ and $v_{k+1} = \mathsf{Proj}_{\mathcal{X}}\big(v_k - \frac{1}{2} a_{k+1} g_{k+1}\big)$
6 $\quad$ **if** $A_{k+1} \geq A_{\max}$ **or** $k + 1 = K_{\max}$ **then return** x_{k+1}

Note that $\widetilde{\mathsf{P}}^0_{f,\lambda} = \mathsf{P}_{f,\lambda}$ and that for $\varphi > 0$ we can compute $\widetilde{\mathsf{P}}^{\varphi}_{f,\lambda}$ with an appropriate SGD variant (such as EPOCHSGD) using $O(G^2/(\lambda\varphi))$ evaluations of $\hat{\nabla} f$.

With the differences between the algorithms explained, we emphasize their key similarity: both algorithms update the v_k sequence using our bias reduction method MORGRADEST (Algorithm 2), which holds the key to their efficiency. The following proposition shows that Algorithm 4 has the same bound on $K_{\max}$ as an exact accelerated proximal point method [12], while requiring at most $\widetilde{O}(G^2 R^2 \epsilon^{-2})$ stochastic gradient evaluations; see proof in Appendix D.1.

Proposition 2. *Let $f : \mathcal{X} \to \mathbb{R}$ and $\hat{\nabla} f$ satisfy Assumption 1, and let $\mathcal{X} \subseteq \mathbb{B}_R(x_0)$. For a target accuracy $\epsilon \leq GR$ let $\varphi_{k+1} = \frac{\epsilon}{60\lambda_{k+1} a_{k+1}}$, $\delta_{k+1} = \frac{\epsilon}{120R}$, $\sigma^2_{k+1} = \frac{\epsilon}{60 a_{k+1}}$, $A_0 = \frac{R}{G}$, and $A_{\max} = \frac{9R^2}{\epsilon}$. If $\lambda_k \geq \lambda_{\min} \geq \frac{1}{A_{\max}} = \Omega(\frac{\epsilon}{R^2})$ for all $k \leq K_{\max}$, then lines 4 and 5 of Algorithm 4 have total complexity $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\Big(K_{\max} \log \frac{GR}{\epsilon} + \frac{G^2 R^2}{\epsilon^2} \log^2 \frac{GR}{\epsilon}\Big)$. If in addition $\|\mathsf{P}_{f,\lambda_k}(y_{k-1}) - y_{k-1}\| \geq 3r/4$ whenever $\lambda_k \geq 2\lambda_{\min}$ then for $K_{\max} = O\Big(\big(\frac{R}{r}\big)^{2/3} \log\big(\frac{GR}{\epsilon}\big) + \sqrt{\frac{\lambda_{\min} R^2}{\epsilon}}\Big)$, the algorithm's output x_K satisfies $f(x_K) - f(x_\star) \leq \epsilon$ with probability at least $\frac{2}{3}$.*

4.2 Minimizing the maximal loss

We now consider objectives of the form $f_{\max}(x) \coloneqq \max_{i \in [N]} f_{(i)}(x)$ where each function $f_{(i)} : \mathcal{X} \to \mathbb{R}$ is convex and G-Lipschitz. Our approach to minimizing $f_{\max}$ largely follows Carmon et al. [13]; the main difference is that we approximate proximal steps via Algorithm 4 and our reduced-bias bias estimator. The first step of the approach is to replace $f_{\max}$ with the "softmax" function, defined for a given target accuracy ϵ as

$$f_{\text{smax}}(x) \coloneqq \epsilon' \log\left(\sum_{i \leq N} \exp\big(f_{(i)}(x)/\epsilon'\big)\right), \text{ where } \epsilon' \coloneqq \frac{\epsilon}{2 \log N}.$$

Since $f_{\text{smax}}(x) - f_{\max}(x) \in [0, \frac{\epsilon}{2}]$, any $\frac{\epsilon}{2}$-accurate solution of f_{smax} is ϵ-accurate for $f_{\max}$.

The second step is to develop an efficient gradient estimator for f_{smax}; this is non-trivial because f_{smax} is not a finite sum or expectation. In [13] this is addressed via an "exponentiated softmax" trick; we develop an alternative, rejection sampling-based approach that fits Algorithm 4 more directly (see Algorithm 9). To produce an unbiased estimate for $\nabla f_{\text{smax}}(x)$ for x in a ball of radius $r_\epsilon = \epsilon'/G$ we require a single $\nabla f_{(i)}(x)$ evaluation (for some i), $O(1)$ evaluations of $f_{(i)}(x)$ in expectation, and evaluation of the N functions $f_{(1)}(y), \ldots, f_{(N)}(y)$ for pre-processing. Plugging this estimator into Algorithm 4 with $r = r_\epsilon$, the total pre-processing overhead of lines 4 and 5 is $O(K_{\max} N)$.

The final step is to find a function NEXTLAMBDA such that $\|\mathsf{P}_{f_{\text{smax}},\lambda_{t+1}}(y_k) - y_k\| \leq r_\epsilon$ for all k (enabling gradient estimation), and $\|\mathsf{P}_{f_{\text{smax}},\lambda_{t+1}}(y_k) - y_k\| \geq \frac{3}{4} r_\epsilon$ when $\lambda_{k+1} > 2\lambda_{\min}$ (allowing us to bound $K_{\max}$ with Proposition 2). Here we use the bisection subroutine from [13] as is (see Algorithm 10). By judiciously choosing $\lambda_{\min}$—an improvement over the analysis in [13]—we obtain the following complexity guarantee on $\mathcal{N}_{f_{(i)}}$ and $\mathcal{N}_{\partial f_{(i)}}$, the total numbers of individual function and subgradient evaluations, respectively. (See proof Appendix D.2).

Theorem 4. *Let $f_{(1)}, \ldots, f_{(N)} : \mathcal{X} \to \mathbb{R}$ be convex and G-Lipschitz and let $\mathcal{X} \subseteq \mathbb{B}_R(x_0)$. For any $\epsilon < \frac{1}{2}GR/\log N$, Algorithm 4 (with $\widetilde{\mathsf{P}}^{\varphi}_{f_{\max},\lambda}$ implemented in Algorithm 8, $\hat{\nabla} f_{\max}$ given by Algorithm 9 and* NEXTLAMBDA *given by Algorithm 10 with $\lambda_{\min} = \widetilde{O}(\epsilon/(r_\epsilon^{4/3}R^{2/3}))$ outputs $x \in \mathcal{X}$ that with probability at least $\frac{1}{2}$ is ϵ-suboptimal for $f_{\max}(x) = \max_{i\in[N]} f_{(i)}(x)$ and has complexity $\mathbb{E}\mathcal{N}_{f_{(i)}} = O\Big(\Big[N\Big(\frac{GR\log N}{\epsilon}\Big)^{2/3} + \big(\frac{GR}{\epsilon}\big)^2\Big]\log^2\frac{GR}{\epsilon}\Big)$ and $E\mathcal{N}_{\partial f_{(i)}} = O\Big(\big(\frac{GR}{\epsilon}\big)^2\log^2\frac{GR}{\epsilon}\Big)$.*

The rate given by Theorem 4 matches (up to logarithmic factors) the lower bound $\Omega(N(GR/\epsilon)^2 + (GR/\epsilon)^2)$ shown in [13] and is therefore near-optimal.

5 Gradient-efficient composite optimization

Consider the problem of finding a minimizer of the following convex composite optimization problem

$$\underset{x\in\mathcal{X}}{\text{minimize}}\ \Psi(x) \coloneqq \Lambda(x) + f(x) \text{ where } \Lambda \text{ is } L\text{-smooth and } f \text{ satisfies Assumption 1}, \tag{5}$$

given x_0 such that $\|x_0 - x_\star\| \le R$ for some $x_\star \in \operatorname{argmin}_{x\in\mathcal{X}} \Psi(x)$. Lan [34] developed a method called "gradient sliding" that finds an ϵ-accurate solution to (5) with complexity $\mathcal{N}_{\nabla\Lambda} = O(\sqrt{LR^2/\epsilon})$ evaluations of $\nabla\Lambda(x)$ and $\mathcal{N}_{\hat{\nabla}f} = O((GR/\epsilon)^2)$ evaluations of $\hat{\nabla}f(x)$, which are optimal even for each component separately.[6]

In this section, we provide an alternative algorithm that matches the complexity of gradient up to logarithmic factors and is conceptually simple. Our approach, Algorithm 5, is essentially composite AGD [41], where at the kth iteration we compute a proximal point (2) with respect to a partial linearization of Ψ around y_k. In particular, letting $\bar{\Lambda}_k(v) \coloneqq \Lambda(y_k) + \langle\nabla\Lambda(y_k), v - y_k\rangle$ and $\beta_k = \frac{2L}{k}$, we approximate $\mathsf{P}_{\bar{\Lambda}+f,\beta_k}(v_{k-1})$. Similar to Algorithm 4, Algorithm 5 computes two types of approximations: one is an ϵ_k-approximate proximal point $\widetilde{\mathsf{P}}^{\epsilon_k}_{\bar{\Lambda}+f,\beta_k}(v_{k-1})$ as per its definition (4), while the other is our bias-reduced optimum estimator from Algorithm 1. We note, however, that unlike Algorithm 4 which approximates the x_k update, here we approximate v_k, the "mirror descent" update.

Below we state the formal guarantees for Algorithm 5; we defer its proof to Appendix E.

Algorithm 5: Stochastic composite accelerated gradient descent

Input: A problem of the form (5) with Λ, f, $\nabla\Lambda$, $\hat{\nabla}f$.
Parameters: Step size parameters $\beta_k = \frac{2L}{k}$ and $\gamma_k = \frac{2}{k+1}$, iteration number N, approximation parameters $\{\epsilon_k, \delta_k, \sigma_k^2\}$ and $x_0 = v_0$ satisfying $\|x_0 - x_\star\| \le R$.

for $k = 1, 2, \cdots, N$ **do**
- $y_k = (1-\gamma_k)x_{k-1} + \gamma_k\mathsf{Proj}_{\mathcal{X}}(v_{k-1})$
- $\bar{v}_k = \widetilde{\mathsf{P}}^{\epsilon_k}_{\bar{\Lambda}_k+f,\beta_k}(v_{k-1})$ for $\bar{\Lambda}_k(v) \coloneqq \Lambda(y_k) + \langle\nabla\Lambda(y_k), v - y_k\rangle$
- $v_k = \textsc{OptEst}(\hat{\nabla}f, \psi_k, \beta_k, \delta_k, \sigma_k^2, \mathbb{B}_R(v_0)\cap\mathcal{X})$ for $\psi_k(z) = \frac{\beta_k}{2}\|z - v_{k-1}\|^2 + \bar{\Lambda}_k(z)$
- $x_k = (1-\gamma_k)x_{k-1} + \gamma_k\bar{v}_k$

return x_N

Theorem 5. *Given problem (5) with solution $x_\star$, a point x_0 such that $\|x_0 - x_\star\| \le R$ and target accuracy $\epsilon > 0$, Algorithm 5 with $\epsilon_k = LR/2kN$, $\delta_k = R/16N$, $\sigma_k^2 = R^2/4N$, and $N = \Theta(\sqrt{LR^2/\epsilon})$ finds an approximate solution x satisfying $\mathbb{E}\Psi(x) \le \Psi(x_\star) + \epsilon$ and has complexity $\mathcal{N}_{\nabla\Lambda} = O\left(\sqrt{\frac{LR^2}{\epsilon}}\right)$ and $\mathbb{E}\mathcal{N}_{\hat{\nabla}f} = O\left(\big(\frac{GR}{\epsilon}\big)^2\log^2\frac{GR}{\epsilon} + \sqrt{\frac{LR^2}{\epsilon}}\log\big(\frac{GR}{\epsilon}\big)\right)$.*

6 Efficient non-smooth private convex optimization

We conclude the paper with a potential application of our optimum estimator for differentially private stochastic convex optimization (DP-SCO). In this problem we are given n i.i.d. sample

[6] The gradient sliding result holds under a relaxed Lipschitz assumption [see 34, eq. (1.2)]. It is straightforward to extend EPOCHSGD, and hence all of our results, to that assumption as well.

$s_i \sim P$ taking values in a set $\mathbb{S}$, and out objective is to privately minimize the population average $f(x) = \mathbb{E}_{S\sim P}[\hat{f}(x;S)]$, where $\hat{f} : \mathcal{X} \times \mathbb{S} \to \mathbb{R}$, is convex in the first argument and $\mathcal{X} \subset \mathbb{R}^d$ is a convex set, and $\mathbb{S}$ is a population of data points. That is, We wish to find $\hat{x} \in \mathcal{X}$ with small excess loss $f(\hat{x}) - \min_{x\in\mathcal{X}} f(x)$ while preserving differential privacy.

Definition 2 ([22])**.** *A randomized algorithm $\mathcal{A}$ is (α, β)-differentially private ((α, β)-DP) if, for all datasets $\mathcal{S}, \mathcal{S}' \in \mathbb{S}^n$ that differ in a single data element and for every event $\mathcal{O}$ in the output space of $\mathcal{A}$, we have $P[\mathcal{A}(\mathcal{S}) \in \mathcal{O}] \leq e^\alpha P[\mathcal{A}(\mathcal{S}') \in \mathcal{O}] + \beta$.*

DP-SCO has received increased attention over the past few years. Bassily et al. [5] developed (inefficient) algorithms that attain the optimal excess loss $1/\sqrt{n} + \sqrt{d\log(1/\beta)}/n\alpha$. When each function is $O(\sqrt{n})$ smooth, Feldman et al. [23] gave algorithms with optimal excess loss and $O(n)$ gradient query complexity. In the non-smooth setting, however, their algorithms require $O(n^2)$ subgradients. Subsequently, Asi et al. [3] and Kulkarni et al. [32] developed more efficient algorithms for non-smooth functions which need $O(\min(n^2/\sqrt{d}, n^{5/4}d^{1/8}, n^{3/2}/d^{1/8}))$ subgradients which is $O(n^{11/8})$ for the high-dimensional setting $d = n$. Whether a linear gradient complexity is achievable for DP-SCO in the non-smooth setting is still open.

In this section, we develop an efficient algorithm for non-smooth DP-SCO that queries $\widetilde{O}(n)$ subgradients conditional on the existence of an optimum estimator with the following properties.

Definition 3. *Let $F = f + \psi$ be μ-strongly convex with minimizer $x_\star$ and f is G-Lipschitz. For $\delta > 0$, we say that $\mathcal{O}_\delta$ is efficient bounded low-bias estimator if it returns $\hat{x}_\star = \mathcal{O}_\delta(F)$ such that $\|\mathbb{E}[\hat{x}_\star - x_\star]\|^2 \leq \delta^2$, $\|\hat{x}_\star - x_\star\|^2 \leq C_1 G^2 \log(G/\mu\delta)/\mu^2$, and the expected number of gradient queries is $C_2 \log(G/\mu\delta)$.*

Comparing to our MLMC estimator (1) and Proposition 1, we note that the only place our current estimator falls short of satisfying Definition 3 is the probability 1 bound on $\|\hat{x}_\star - x_\star\|^2$, which for (1) holds only in expectation. Indeed, for our estimator, $\|\hat{x}_\star - x_\star\|$ can be as large as $O(G/(\mu\delta))$, meaning that it is heavy-tailed.

It is not clear whether an EBBOE as defined above exists. Nevertheless, assuming access to such estimator, Algorithm 6 solves the DP-SCO problem with a near-linear amount of gradient computations. The algorithm builds on the recent localization-based optimization methods in [23] which iteratively solve regularized minimization problems.

Algorithm 6: Differentially-private stochastic convex optimization via optimum estimation

Input: $(s_1, \ldots, s_n) \in \mathbb{S}^n$, domain $\mathcal{X} \subset \mathbb{B}_R(x_0)$, EBBOE $\mathcal{O}$ (satisfying Definition 3).

Set $k = \lceil \log n \rceil$, $B = 20(\log(\frac{1}{\beta}) + C_2 \log^2 n)$, $\bar{n} = \frac{n}{k}$, $\eta = \frac{R}{G}\min\left\{\frac{1}{\sqrt{n}}, \frac{\alpha}{B\log(n)\sqrt{d\log(\frac{1}{\beta})}}\right\}$

for $i = 1, 2, \cdots, k$ **do**

  Let $\eta_i = 2^{-4i}\eta$, $f_i(x) = \frac{1}{\bar{n}}\sum_{j=1+(k-1)\bar{n}}^{k\bar{n}} \hat{f}(x; s_j)$, $\psi_i(x) = \|x - x_{i-1}\|^2/(\eta_i \bar{n})$

  Let $\tilde{x}_i = \frac{1}{\bar{n}}\sum_{j=1}^{\bar{n}} \mathcal{O}_{\delta_i}(F_i)$ with $F_i = f_i + \psi_i$, $\delta_i^2 = G^2\eta_i^2\bar{n}$

  Set $x_i = \tilde{x}_i + \zeta_i$ where $\zeta_i \sim \mathsf{N}(0, \sigma_i^2 I_d)$ with $\sigma_i = 8B(\sqrt{C_1 \log n} + 2)\eta_i\sqrt{\log(2/\beta)}/\alpha_i$

return x_k

We average multiple draws of the (hypothetical) bounded optimum estimator to solve the regularized problems, and apply private mean estimation procedures to preserve privacy. We defer the proof of the following results Appendix F.

Theorem 6 (conditional)**.** *Given an efficient bounded low-bias estimator $\mathcal{O}_\delta$ satisfying Definition 3 for any $\delta > 0$, then for $\alpha \leq \log(1/\beta)$, $\mathcal{X} \in \mathbb{B}_R(x_0)$, convex and G-Lipschitz $\hat{f}(x; s)$, Algorithm 6 is (α, β)-DP, queries $\widetilde{O}(n)$ subgradients and has (hiding logarithmic factors in n) $\mathbb{E}[f(x_k) - \min_{x\in\mathcal{X}} f(x)] \leq GR \cdot \widetilde{O}\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log^3(1/\beta)}}{n\alpha}\right)$.*

Theorem 6 provides a strong motivation for constructing bounded optimum estimators that satisfy Definition 3 . In Appendix F.3, we discuss the challenges in making our MLMC estimator bounded, as well as some directions to overcome them.

Acknowledgments

HA was supported by ONR YIP N00014-19-2288 and the Stanford DAWN Consortium. YC was supported in part by Len Blavatnik and the Blavatnik Family foundation, and the Yandex Machine Learning Initiative for Machine Learning. YJ was supported by a Stanford Graduate Fellowship. 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

[1] Z. Allen-Zhu. Katyusha: the first direct acceleration of stochastic gradient methods. In *Proceedings of the Forty-Ninth Annual ACM Symposium on the Theory of Computing*, 2017.

[2] Z. Allen-Zhu and L. Orecchia. Linear coupling: An ultimate unification of gradient and mirror descent. In *Proceedings of the 8th Innovations in Theoretical Computer Science*, ITCS '17, 2017.

[3] H. Asi, V. Feldman, T. Koren, and K. Talwar. Private stochastic convex optimization: Optimal rates in ℓ_1 geometry. In *Proceedings of the 38th International Conference on Machine Learning*, 2021.

[4] R. Bassily, A. Smith, and A. Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *55th Annual Symposium on Foundations of Computer Science*, pages 464–473, 2014.

[5] R. Bassily, V. Feldman, K. Talwar, and A. Thakurta. Private stochastic convex optimization with optimal rates. In *Advances in Neural Information Processing Systems*, volume 32, pages 11282–11291, 2019.

[6] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

[7] J. Blanchet and Y. Kang. Semi-supervised learning based on distributionally robust optimization. In *Stochastic Modeling Techniques and Data Analysis International Conference*, 2018.

[8] J. H. Blanchet and P. W. Glynn. Unbiased Monte Carlo for optimization and functions of expectations via multi-level randomization. In *2015 Winter Simulation Conference (WSC)*, pages 3656–3667, 2015.

[9] J. H. Blanchet, P. W. Glynn, and Y. Pei. Unbiased multilevel Monte Carlo: Stochastic optimization, steady-state simulation, quantiles, and other applications. *arXiv:1904.09929 [math.ST]*, 2019.

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

[11] S. Bubeck, Q. Jiang, Y. T. Lee, Y. Li, and A. Sidford. Complexity of highly parallel non-smooth convex optimization. *arXiv:1906.10655 [math.OC]*, 2019.

[12] Y. Carmon, A. Jambulapati, Q. Jiang, Y. Jin, Y. T. Lee, A. Sidford, and K. Tian. Acceleration with a ball optimization oracle. In *Advances in Neural Information Processing Systems*, 2020.

[13] Y. Carmon, A. Jambulapati, Y. Jin, and A. Sidford. Thinking inside the ball: Near-optimal minimization of the maximal loss. In *Conference on Learning Theory*, 2021.

[14] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12:1069–1109, 2011.

[15] K. L. Clarkson, E. Hazan, and D. P. Woodruff. Sublinear optimization for machine learning. *Journal of the ACM (JACM)*, 59(5):1–49, 2012.

[16] M. Cohen, J. Diakonikolas, and L. Orecchia. On acceleration with noise-corrupted gradients. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2018.

[17] D. Davis and D. Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

[18] J. Diakonikolas and C. Guzmán. Lower bounds for parallel and randomized convex optimization. In *Proceedings of the Thirty Second Annual Conference on Computational Learning Theory*, pages 1132–1157, 2019.

[19] D. Drusvyatskiy. The proximal point method revisited. *SIAG/OPT Views and News*, 26(1), 2018.

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

[21] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3 & 4):211–407, 2014.

[22] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Theory of Cryptography Conference*, pages 265–284, 2006.

[23] V. Feldman, T. Koren, and K. Talwar. Private stochastic convex optimization: optimal rates in linear time. In *Proceedings of the 52nd Annual ACM on the Theory of Computing*, pages 439–449, 2020.

[24] R. Frostig, R. Ge, S. Kakade, and A. Sidford. Un-regularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *International Conference on Machine Learning*, 2015.

[25] M. B. Giles. Multilevel Monte Carlo methods. *Acta Numerica*, 24:259–328, 2015.

[26] G. Grimmett and D. Stirzaker. *Probability and random processes*. Oxford university press, 2020.

[27] Y. Gurevich and S. Shelah. Nearly linear time. In *International Symposium on Logical Foundations of Computer Science*, pages 108–118, 1989.

[28] E. Hazan and S. Kale. Beyond the regret minimization barrier: optimal algorithms for stochastic strongly-convex optimization. *Journal of Machine Learning Research*, 15(1):2489–2512, 2014.

[29] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods*. Springer, 1993.

[30] H. Jiang, Y. T. Lee, Z. Song, and S. C. Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proceedings of the Fifty-Second Annual ACM Symposium on the Theory of Computing*, pages 944–953. ACM, 2020.

[31] L. G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademii Nauk*, 244(5):1093–1096, 1979.

[32] J. Kulkarni, Y. T. Lee, and D. Liu. Private non-smooth empirical risk minimization and stochastic convex optimization in subquadratic steps. *arXiv:2103.15352 [cs.LG]*, 2021.

[33] G. Lan. Bundle-level type methods uniformly optimal for smooth and nonsmooth convex optimization. *Mathematical Programming*, 149(1):1–45, 2015.

[34] G. Lan. Gradient sliding for composite optimization. *Mathematical Programming*, 159(1): 201–235, 2016.

[35] D. Levy, Y. Carmon, J. C. Duchi, and A. Sidford. Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems*, 2020.

[36] H. Lin, J. Mairal, and Z. Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, 2015.

[37] M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[38] R. D. C. Monteiro and B. F. Svaiter. An accelerated hybrid proximal extragradient method for convex optimization and its implications to second-order methods. *SIAM J. Optim.*, 23(2): 1092–1125, 2013.

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

[40] A. Nemirovski and D. Yudin. *Problem complexity and method efficiency in optimization*. Wiley-Interscience, 1983.

[41] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[42] Y. Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[43] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3): 123–231, 2013.

[44] A. Rakhlin, O. Shamir, and K. Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *International Coference on International Conference on Machine Learning*, pages 1571–1578, 2012.

[45] S. Shalev-Shwartz and Y. Wexler. Minimizing the maximal loss: How and why? In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

[46] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *Proceedings of the Twenty Second Annual Conference on Computational Learning Theory*, 2009.

[47] Y. Shi and R. Cornish. On multilevel Monte Carlo unbiased gradient estimation for deep latent variable models. In *International Conference on Artificial Intelligence and Statistics*, pages 3925–3933. PMLR, 2021.

[48] N. Z. Shor. Cut-off method with space extension in convex programming problems. *Cybernetics*, 13(1):94–96, 1977.

[49] A. Sinha, H. Namkoong, and J. Duchi. Certifying some distributional robustness with principled adversarial training. In *International Conference on Learning Representations*, 2018.

[50] C. Song, Y. Jiang, and Y. Ma. Unified acceleration of high-order algorithms under Hölder continuity and uniform convexity. *SIAM Journal on Optimization*, 2021.

[51] K. K. Thekumparampil, P. Jain, P. Netrapalli, and S. Oh. Projection efficient subgradient method and optimal nonsmooth frank-wolfe method. In *Advances in Neural Information Processing Systems*, 2020.

[52] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. 2008.

[53] V. N. Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[54] B. Woodworth and N. Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in Neural Information Processing Systems 29*, 2016.

[55] D. B. Yudin and A. S. Nemirovski. Evaluation of the information complexity of mathematical programming problems. *Ekonomika i Matematicheskie Metody*, 12:128–142, 1976.

Appendix

A Additional results and discussion

Here we provide additional discussion of three topics pertinent to our results: a zero-bias optimum estimator, the parallel depth of our estimator, turning expected complexity bounds into deterministic ones, and a justification for the adjective "optimal" in Definition 1 of the "optimal distance convergence" property. We recommend reading Sections 1 to 3 before the subsections below.

A.1 Zero-bias optimum estimation given exact gradients

The main tool developed in this paper is an optimum estimator with bias δ whose expected query complexity is $O(\log(G/(\mu\delta)))$. In this section, we show how to obtain a completely unbiased optimum estimator when, in addition to a stochastic subgradient oracle, we assume access to a first-order oracle, i.e., one which outputs the functions exact value and subgradient at the query point.

To be concrete, assume that the domain is a ball of radius R in $\mathbb{R}^d$, and that the objective $F : \mathbb{B}_R(x_0) \to \mathbb{R}$ is μ-strongly-convex and of the form $F(x) = \frac{1}{n}\sum_{i\in[n]} \hat{F}(x;i)$, where each $\hat{F}(\cdot;i)$ is G-Lipschitz and given by a first-order oracle. In this case, we can compute an unbiased subgradient estimator with single oracle query by sampling $i \sim \mathsf{Unif}([n])$ and taking $\hat{\nabla}F \in \partial\hat{F}(x;i)$. Further, value and subgradient evaluations of F can be implemented at n-times the cost by querying each each F_i. In this setting, we design an unbiased estimator of $x_\star = \operatorname{argmin}_{x\in\mathbb{B}_R(0)} F(x)$ with variance $O((G^2/\mu^2)\log(nd))$, expected query complexity $O(\log(nd))$ and expected runtime $O(d\log(nd))$.

To obtain this result, we leverage that first-order methods can compute the minimizer of a convex function with a number of queries and runtime that depends polynomially on dimension and logarithmically on regularity parameters and the desired accuracy. In fact, any polynomial bound suffices for our purposes and effect only constants factors in our expected complexity bounds. For concreteness, we use the classic ellipsoid method [55, 48, 31] whose complexity we describe in the following lemma. (We remark, however, that improved query complexities and runtimes are achievable; see [30] for the state-of-the-art).

Lemma 2 (Ellipsoid method)**.** *There is an algorithm,* $\textsc{Ellipsoid}(x_0, f, T)$*, which given* $x_0 \in \mathbb{R}^d$*, a first order oracle for* G*-Lipschitz,* μ*-strongly-convex* $f : \mathbb{B}_R(x_0) \to \mathbb{R}$*, and query budget* $T \geq 0$*, runs in* $O(d^2T)$ *time, makes at most* T *queries, and outputs* $\hat{x}_\star \in \mathbb{B}_R(x_0)$ *with* $\|\hat{x}_\star - x_\star\|_2^2 \leq (8G^2/\mu^2)\exp(-T/(2d^2))$ *for* $x_\star \coloneqq \operatorname{argmin}_{x\in\mathbb{B}_R(x_0)} f(x)$*.*

Proof. Since f is G-Lipschitz for all $x \in \mathbb{B}_R(x_0)$ we have $|f(x) - f(x_0)| \leq GR$. Consequently, the ellipsoid method applied to $f(x) - f(x_0)$ can compute $\hat{x}_\star \in \mathbb{B}_R(x_0)$ with $f(\hat{x}_\star) - f(x_\star) \leq 2GR\exp(-T/(2d^2))$ with $O(T)$ queries and $O(d^2T)$ time [see, e.g., 10, Theorem 2.4]. Since by strong convexity $\|\hat{x}_\star - x_\star\|_2^2 \leq \frac{2}{\mu}[f(\hat{x}_\star) - f(x_\star)]$ this implies that $\|\hat{x}_\star - x_\star\|_2^2 \leq (4GR/\mu)\exp(-T/(2d^2))$. Further, since f is G-Lipschitz and μ-strongly-convex we know that for all $y \in \mathbb{B}_R(x_0)$ we have $G\|y - x_\star\| \geq f(y) - f(x_\star) \geq \frac{\mu}{2}\|y - x_\star\|^2$ and since $\mathbb{B}_R(x_0)$ contain a point y with $\|y - x_\star\| \geq R$ this implies $R \leq 2G/\mu$. Combining yields the result. □

Combining the ellipsoid method with an ODC algorithm (see Definition 1) we obtain our unbiased optimum estimator, which we formally describe in Algorithm 7. The procedure is similar to the

Algorithm 7: Unbiased optimum estimator

Input: Initialization $x_0 \in \mathbb{R}^d$, first-order oracles for $\hat{F}(x;i)$ for all $i \in [n]$ and ODC algorithm ODC.

Let $J_0 \coloneqq \lceil 4\log_2(14(nd^2 + d^4))\rceil$

For all $j > 1$ let $x_j \coloneqq \begin{cases} \textsc{ODC}(\nabla\hat{F}(\cdot;i), 0, 2^j) & \text{if } j \leq J_0 \\ \textsc{Ellipsoid}(x_0, F, \lceil 2^{j/2}\rceil) & \text{if } j > J_0 \end{cases}$

Draw $J \sim \mathsf{Geom}\left(\frac{1}{2}\right)$

return $x_0 + 2^J(x_{J-1} - x_J)$ ▷ Only x_0, x_{J-1}, and x_J are computed explicitly by the algorithm.

MLMC estimator (1), with the key difference that when (1) would output x_0, we instead apply the ellipsoid method. The following theorem establishes the performance of our algorithm.

Theorem 7 (Unbiased optimum estimator)**.** *Let $F : \mathcal{X} \to \mathbb{R}$ be μ-strongly convex with, $\mathcal{X} = \mathbb{B}_R(x_0)$, $F(x) = \frac{1}{n}\sum_{i\in[n]} \hat{F}(x;i)$ for all $x \in \mathcal{X}$ and $\mathbb{E}\|\nabla F(x;i)\|^2 \le G^2$ for all $x \in \mathcal{X}$ and $i \sim \mathsf{Unif}([n])$. Algorithm 7 outputs $\hat{x}_\star$ with $\mathbb{E}\hat{x}_\star = x_\star = \operatorname{argmin}_{x\in\mathbb{B}_R(x_0)} F(x)$ and $\mathbb{E}\|x - x_\star\|_2^2 = O(\frac{G^2}{\mu^2}\log(nd))$ with expected $O(\log(nd))$ queries to $(\hat{F}(x;i), \nabla\hat{F}(x;i))$ and expected $O(d\log(nd))$ time.*

Proof. Note that

$$\begin{aligned}
\mathbb{E}\|\hat{x}_\star - x_\star\|^2 &= \sum_{j=1}^{\infty} \frac{1}{2^j}\cdot \mathbb{E}\|x_0 - x_\star + 2^j(x_{j-1} - x_j)\|^2 \\
&\le \sum_{j=1}^{\infty} \frac{2}{2^j}\mathbb{E}\left[\|x_0 - x_\star\|^2 + 2^{2j}\|x_{j-1} - x_j\|^2\right] \\
&\le 2\mathbb{E}\|x_0 - x_\star\|^2 + 4\sum_{j=1}^{\infty} 2^j\mathbb{E}\left[\|x_{j-1} - x_\star\|^2 + \|x_j - x_\star\|^2\right] \\
&= 10\|x_0 - x_\star\|^2 + 4\sum_{j=1}^{\infty}(2^j + 2^{j+1})\mathbb{E}\|x_j - x_\star\|^2 \le 12\sum_{j=0}^{\infty} 2^j\mathbb{E}\|x_j - x_\star\|^2.
\end{aligned}$$

Further, by definition of ODC we have that $\mathbb{E}\|x_j - x_\star\|_2^2 \le (cG^2/\mu^2)2^{-j/2}$ for all $j \le J_0$ where c is the constant in Definition 1. Also, by Lemma 2 we have $\|x_j - x_\star\|_2^2 \le (8G^2/\mu^2)\exp(-\lceil 2^{j/2}\rceil/(2d^2))$ for all $j > J_0$ (since by assumption and Jensen's inequality for $i \sim \mathsf{Unif}([n])$ we have $\|\nabla F(x)\|^2 = \|\mathbb{E}\nabla F(x;i)\|^2 \le \mathbb{E}\|\nabla F(x,i)\|^2 \le G^2$ for alll $x \in \mathcal{X}$ and therefore F is G-Lipschitz). Note that $j \le 5\cdot 2^{j/4}$ for all $j \ge 1$ and $5\cdot 2^{j/4}\ln 2 \le 2^{j/2}/(4d^2)$ for all $j \ge 4\log_2(14d^2)$. Consequently, $\frac{\lceil 2^{j/2}\rceil}{2d^2} \ge 2j\ln 2$ and $\|x_j - x_\star\|_2^2 \le (8G^2/\mu^2)2^{-2j}$ for all $j > J_0$. Therefore,

$$\mathbb{E}\|\hat{x}_\star - x_\star\|^2 \le 12\sum_{j=0}^{J_0}\frac{cG^2}{\mu^2} + 12\sum_{j=J_0+1}^{\infty}\frac{8G^2}{\mu^2}2^{-j} \le 12(cJ_0 + 8)\frac{G^2}{\mu^2} = O\left(\frac{G^2}{\mu^2}\log(nd)\right).$$

Further,

$$\mathbb{E}x_\star = \sum_{j=1}^{\infty} 2^{-j}[x_0 + 2^j(x_j - x_{j-1})] = x_0 + \sum_{j=1}^{\infty}(x_j - x_{j-1}) = \lim_{j\to\infty} x_j = x_\star.$$

Now, note that when $J \le J_0$ the algorithm makes 2^J subgradient queries and runs in time $O(d2^J)$. Further, when $J > J_0$ by Lemma 2 the algorithm makes $\lceil 2^{J/2}\rceil \le 2^{1+(J/2)}$ first-order oracle queries, costing $O(n2^{(J/2)})$ sub-gradient in total, and runs in time $O((nd^2 + d^4)2^{J/2})$. Consequently, the expected number of subgradient queries is upper bounded by

$$\sum_{j\in[J_0]}\frac{1}{2^j}\cdot 2^j + \sum_{j>J_0}^{\infty}\frac{1}{2^j}\cdot n2^{1+(j/2)} = J_0 + \frac{2n}{2^{J_0/2}}\sum_{j=1}^{\infty}\frac{1}{2^{j/2}} = O(J_0) = O(\log(nd))$$

where in the last step we used that $J_0 = \Omega(\log(n))$ and $\sum_{j=1}^{\infty}\frac{1}{2^{j/2}} = O(1)$. Similarly, since $J_0 \ge \log_2(nd^2 + d^4)$ the expected runtime is at most

$$\begin{aligned}
&\sum_{j\in[J_0]}\frac{1}{2^j}\cdot O(d2^j) + \sum_{j>J_0}^{\infty}\frac{1}{2^j}\cdot O\left((nd^2 + d^4)2^{j/2}\right) \\
&\quad = O(J_0\cdot d) + O\left(2^{-J_0}(nd^2 + d^4)\right)\cdot\sum_{j=1}^{\infty}\frac{1}{2^{j/2}} = O(J_0\cdot d) = O(d\log(nd)).
\end{aligned}$$

□

A.2 The sequential depth of our optimum estimator

Let us discuss the implications of our development—or more precisely, the lack thereof—on the parallel complexity of non-smooth optimization. Following the standard setting for this problem, consider the task of minimizing a G-Lipschitz convex function f in a domain of diameter R in $\mathbb{R}^d$ given the ability to query a subgradient oracle for f in batches of B parallel queries. That is, at round t we query points $x_t^{(1)}, \ldots, x_t^{(B)}$ and observe subgradients $g_t^{(i)} \in \partial f(x_t^{(i)})$ for $i \in [B]$. In sufficiently high dimension, the ability to query B points in parallel does not improve worst-case complexity: for required accuracy ϵ and algorithm with batch size $B = \mathsf{poly}(1/\epsilon)$, there exists a problem instance in dimension $d = O((\frac{GR}{\epsilon})^4 \log \frac{GR}{\epsilon})$ for which the algorithm must make $T = \Omega((\frac{GR}{\epsilon})^2)$ queries in sequence in order to find an ϵ-accurate solution [11].

At first glance, our algorithms—and Algorithm 3 in particular—seem to contradict the lower bound described above. Indeed, the algorithm performs $O(\frac{GR}{\epsilon})$ iterations, where each iterations consists of averaging $\widetilde{O}(\frac{GR}{\epsilon})$ copies of the optimum estimator (1). Since we can compute copies of the estimator in parallel, the sequential depth of the algorithm appears to be only $O(\frac{GR}{\epsilon})$. To resolve the apparent contradiction, recall that each evaluation of (1) itself involves a sequential computation. In particular, while an evaluation of (1) has depth $\widetilde{O}(1)$ on average, it also has depth $\Omega(\frac{GR}{\epsilon})$ with probability $\Omega(\frac{\epsilon}{GR})$. Therefore, for a batch of $O(\frac{GR}{\epsilon})$ copies of the estimator, one of them would have depth $\Omega(\frac{GR}{\epsilon})$ with constant probability, implying an overall bound of $\Omega((\frac{GR}{\epsilon})^2)$ on the sequential depth of Algorithm 3.

Viewed another way, the parallelism lower bound implies a limitation on the sequential depth *distribution* of any lower bias optimum estimator. More specifically, let $\hat{T}$ be a random variable representing the sequential depth of a single copy of a low-bias optimum estimator and let $\hat{T}_1, \ldots, \hat{T}_{BK}$ be i.i.d. copies of that random variable, with B and K denoting batch size and AGD depth respectively. Then, when setting $B = K = O(\frac{GR}{\epsilon})$ we must have

$$\sum_{k \in [K]} \max_{b \in [B]} \left\{ \hat{T}_{b+(k-1)B} \right\} = \Omega\left(\left(\frac{GR}{\epsilon} \right)^2 \right)$$

with high probability. In particular, it is impossible to create a low-bias optimum estimator whose depth is $\widetilde{O}(1)$ with high probability. This fact might serve as a useful sanity check when designing new optimum estimators.

A.3 Obtaining deterministic complexity bounds

This paper measures complexity via $\mathcal{N}_{\hat{\nabla} f}$, the number of gradient estimator evaluations by the algorithm. The performance guarantees of our algorithms bound the *expected complexity* while guaranteeing correctness with constant probability. In particular our guarantees in Sections 3 to 5 have the following general form: the algorithm outputs x such that $f(x) - \min_{z \in \mathcal{X}} f(z) \leq \epsilon$ with probability at least p, and $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} \leq \mathsf{C}(\epsilon)$. To guarantee a probability 1 bound on $\mathcal{N}_{\hat{\nabla} f}$, we may terminate the algorithm and output an arbitrary point whenever $\mathcal{N}_{\hat{\nabla} f}$ exceeds $\frac{2}{p}\mathsf{C}(\epsilon)$. By Markov's inequality such termination occurs with probability at most $p/2$ and therefore by the union bound we will output a correct x (satisfying $f(x) - \min_{z \in \mathcal{X}} f(z) \leq \epsilon$) with probability at least $p/2$.

In Section 6 we describe a differentially-private algorithm with bounded expected error and expected gradient estimation complexity. Here too, we may terminate the algorithm if the number of gradient estimations exceeds the bound on the expectation by more than a constant, and maintain a constant probability bound on the error. Since the random amount of gradient estimations in this algorithm is independent of the input (and in fact can be computed ahead of the algorithm's execution), the termination strategy described above does not affect the algorithm's privacy guarantee.

A.4 The optimal distance convergence rate

Definition 1 of an optimal-distance-convergence (ODC) algorithm implies a claim on the optimal rate of convergence (in Euclidean norm) to the minimizer of strongly-convex and Lipschitz functions. Lemma 1 shows that this rate is achievable, and here we sketch a matching lower bound, showing that this rate is not improvable and therefore optimal. More precisely, we exhibit a function F that is G-Lipschitz, μ strongly-convex, has minimizer $x_\star$ and satisfies the following: for every algorithm

that queries points in the span of previously observed subgradients and outputs x_T after T queries, we have $\|x_T - x_\star\| \geq \Omega(G/(\mu\sqrt{T}))$. The restriction of queries to the span of previous gradients is a standard simplifying assumptions [42], and we can extend the claim to any randomized algorithm by choosing a random coordinate system [54, 11].

Let us describe our hard instance construction for algorithms that execute T steps, which we denote by F. The function $F : \mathbb{R}^{2T} \to \mathbb{R}$ is a strongly-convex variant of Nemirovski's function [40, 39, 18, 11], defined as follows

$$F(x) \coloneqq \frac{G}{2} \max_{i \in [2T]} \{x_{[i]}\} + \frac{\mu}{2}\|x\|^2.$$

Note that the function is μ-strongly-convex, and—when constrained to a ball of radius $G/(2\mu)$ around the origin—is G-Lipschitz as required. It is also easy to verify that the minimizer of the function is

$$x_\star = -\frac{G}{4\mu T}\mathbf{1},$$

where $\mathbf{1}$ denotes to the all-ones vector in $\mathbb{R}^{2T}$, since a calculation shows that $0 \in \partial F(x_\star)$.

To establish our claimed lower bound, consider a subgradient oracle for $\max_{i\in[2T]}\{x_{[i]}\}$ which only outputs 1-sparse subgradients of F (it is also possible to design differentiable hard instances via Moreau-Yoshida smoothing, see, e.g., [18]). Then, the query x_T at iteration T is in the span of T 1-sparse vectors, which means that at least T of its coordinates are zero. Recalling the expression of $x_\star$, this implies the claim that

$$\|x_T - x_\star\| \geq \frac{G}{4\mu T}\sqrt{T} = \Omega\left(\frac{G}{\mu\sqrt{T}}\right).$$

B Proofs and additional results from Section 2

B.1 Analysis of EpochSGD

Algorithm 8 is a composite variant of the "epoch SGD" algorithm of Hazan and Kale [28]. We note that when $\psi(x) = \frac{\mu}{2}\|x - z\|^2$ (as it is in all of our applications), the gradient step in line 5 of the algorithm is simply

$$x_k^{t+1} = \mathsf{Proj}_{\mathcal{X}}\left(\frac{1}{1+\mu\eta_k}\left[x_k^t + \mu\eta_k z - \eta_k \hat{\nabla} f(x_k^t)\right]\right),$$

where $\mathsf{Proj}_{\mathcal{X}}$ is the Euclidean projection to $\mathcal{X}$. To analyze Algorithm 8, we first prove the following standard single-epoch optimization guarantee. Below, we let $V_x(x') \coloneqq \frac{1}{2}\|x' - x\|_2^2$ denote the Bregman divergence induced by $\frac{1}{2}\|\cdot\|_2^2$.

Algorithm 8: EpochSGD$(\hat{\nabla} f, \psi, \mu, \mathcal{X}, T)$

Input: A μ-strongly-convex function $F = f + \psi : \mathcal{X} \to \mathbb{R}$ with f satisfying Assumption 1, iteration budget T.

Parameters : Initial step size $\eta_1 = 1/(4\mu)$ and epoch length $T_1 = 16$.

1 Initialize $x_1^0 \in \arg\min_{x\in\mathcal{X}} \psi(x)$, and set $k = 1$

2 **while** $\sum_{i\in[k]} T_i \leq T$ **do**

3 $\quad x_k^1 = \arg\min_{x\in\mathcal{X}} \left(\eta_k\psi(x) + \frac{1}{2}\|x - x_k^t\|^2\right)$

4 $\quad$ **for** $t = 1, 2, \cdots T_k - 1$ **do**

5 $\quad\quad x_k^{t+1} = \arg\min_{x\in\mathcal{X}} \left(\eta_k\left(\langle\hat{\nabla} f(x_k^t), x\rangle + \psi(x)\right) + \frac{1}{2}\|x - x_k^t\|^2\right)$

6 $\quad$ Set $x_{k+1}^0 = \frac{1}{T_k}\sum_{t\in[T_k]} x_k^t$, update $T_{k+1} = 2T_k$, $\eta_{k+1} = \eta_k/2$ and $k \leftarrow k+1$

7 **return** $x = x_k^0$

Lemma 3. *Let $f : \mathcal{X} \to \mathbb{R}$ and $\hat{\nabla} f$ satisfy Assumption 1. For any $k \geq 1$, $T \geq 1$ and $u \in \mathcal{X}$, the iterates of Algorithm 8 satisfy*

$$\mathbb{E}\left[F\left(\frac{1}{T}\sum_{t\in[T]} x_k^t\right)\right] - F(u) \leq \frac{V_{x_k^0}(u)}{\eta T} + \frac{\eta}{2}G^2.$$

Proof. $x_k^t \equiv x^t$, $\eta_k \equiv \eta$, and $T_k \equiv T$. We furthermore let $x^{T+1} \equiv u$, and let $g^t \coloneqq \hat{\nabla} f(x^t)$ for $t \geq 1$ and $g^0 \coloneqq 0$. By the optimality conditions of the minimization in line 3 and line 5, we have

$$\langle \eta \left(g^{t-1} + \nabla\psi(x^t)\right) + x^t - x^{t-1}, x^t - u\rangle \leq 0 \text{ for all } t \in [T],$$

and consequently

$$\langle g^{t-1} + \nabla\psi(x^t), x^t - u\rangle \leq \frac{1}{\eta}\big(V_{x^{t-1}}(u) - V_{x^t}(u) - V_{x^{t-1}}(x^t)\big) \text{ for all } t \in [T].$$

Using the convexity of ψ and the bound above, we obtain

$$\begin{aligned}
&\sum_{t\in[T]} \langle g^{t-1}, x^t - u\rangle + \sum_{t\in[T]} \big(\psi(x^t) - \psi(u)\big) \\
&\qquad \leq \sum_{t\in[T]} \langle g^{t-1} + \nabla\psi(x^t), x^t - u\rangle \\
&\qquad \leq \frac{1}{\eta} \sum_{t\in[T]} \big(V_{x^{t-1}}(u) - V_{x^t}(u) - V_{x^{t-1}}(x^t)\big) \\
&\qquad \leq \frac{1}{\eta} V_{x^0}(u) - \frac{1}{\eta}\sum_{t=0}^{T} V_{x_t}(x^{t+1}).
\end{aligned}$$

Adding $\sum_{t\in[T]}\langle g^t, x^t - x^{t+1}\rangle$ to both sides, recalling that $x^{T+1} \equiv u$ and $g^t = \hat{\nabla} f(x^t)\mathbb{1}_{\{t>0\}}$, and rearranging terms, we have

$$\begin{aligned}
&\sum_{t\in[T]} \langle \hat{\nabla} f(x^t), x^t - u\rangle + \sum_{t\in[T]} \big(\psi(x^t) - \psi(u)\big) \\
&\qquad \leq \frac{1}{\eta} V_{x^0}(u) - \frac{1}{\eta}\sum_{t=0}^{T} V_{x_t}(x^{t+1}) + \sum_{t\in[T]} \langle \hat{\nabla} f(x^t), x^t - x^{t+1}\rangle \\
&\qquad \leq \frac{1}{\eta} V_{x^0}(u) + \sum_{t\in[T]} \frac{\eta}{2}\|\hat{\nabla} f(x^t)\|^2,
\end{aligned}$$

where in the last transition we used $\langle g, x - y\rangle \leq \frac{1}{\eta}V_y(x) + \frac{\eta}{2}\|g\|^2$. Taking expectation, applying Assumption 1 and using convexity of f, we have

$$\mathbb{E}\sum_{t\in[T]} \big(F(x^t) - F(u)\big) \leq \frac{1}{\eta} V_{x^0}(u) + \frac{T}{2}\eta G^2.$$

Dividing by T and applying Jensen's inequality to bound $F\left(\frac{1}{T}\sum_{t\in[T]} x^t\right) \leq \frac{1}{T}\sum_{t\in[T]} F(x^t)$ yields the claimed bound. □

We now are ready to prove the main guarantee of Algorithm 8 (see also Lemma 8, Theorem 5 in Hazan and Kale [28]), which implies Lemma 1.

Proposition 3. *Let $F : \mathcal{X} \to \mathbb{R}$ by a μ-strongly-convex function of the form $F = f + \psi$, such that f satisfies Assumption 1 and $x_\star = \operatorname{argmin}_{x\in\mathcal{X}} F(x)$. Then, for any $T \geq 1$, we have that $x = \text{EPOCHSGD}(\hat{\nabla} f, \psi, \mu, \mathcal{X}, T)$ satisfies*

$$\mathbb{E}F(x) - F(x_\star) \leq \frac{16G^2}{\mu T} \quad \textit{and} \quad \mathbb{E}\|x - x_\star\|^2 \leq \frac{32G^2}{\mu^2 T}.$$

Consequently, EPOCHSGD *is an ODC algorithm with constant $c = 32$.*

Proof. First we claim that $F(x_1^0) - F(x_\star) \leq \frac{G^2}{2\mu}$. To see this we have by μ-strong-convexity of F that

$$\begin{aligned}
F(x_\star) &\geq F(x_1^0) + \langle \nabla f(x_1^0), x_\star - x_1^0\rangle + \langle \nabla\psi(x_1^0), x_\star - x_1^0\rangle + \frac{\mu}{2}\|x_1^0 - x_\star\|^2 \\
&\geq F(x_1^0) + \langle \nabla f(x_1^0), x_\star - x_1^0\rangle + \frac{\mu}{2}\|x_1^0 - x_\star\|^2,
\end{aligned}$$

where we use the definition that $x_1^0 \in \arg\min_{x\in\mathcal{X}} \psi(x)$ and its first-order optimality condition for the second inequality. Rearranging terms gives

$$F(x_1^0) - F(x_\star) \le -\langle \nabla f(x_1^0), x_\star - x_1^0\rangle - \frac{\mu}{2}\|x_1^0 - x_\star\|^2$$
$$\le \max_x \left(-\langle \nabla f(x_1^0), x - x_1^0\rangle - \frac{\mu}{2}\|x_1^0 - x\|^2\right) = \frac{\|\nabla f(x_1^0)\|^2}{2\mu} \le \frac{G^2}{2\mu}.$$

For $x_\star = \operatorname{argmin}_{x\in\mathcal{X}} F(x)$, we so define the potential $\Delta_k = F(x_k^0) - F(x_\star)$ and use induction to prove that $\mathbb{E}\Delta_k \le \frac{G^2}{2^k\mu}$ for all k, with the base case $k = 1$ established above. Suppose that $\mathbb{E}\Delta_k \le \frac{G^2}{2^k\mu}$ for a fixed k. Then for $k+1$ Lemma 3 yields

$$\mathbb{E}\Delta_{k+1} \le \frac{\mathbb{E}V_{x_k^0}(x_\star)}{\eta_k T_k} + \frac{\eta_k}{2}G^2 \overset{(i)}{\le} \frac{\mathbb{E}\Delta_k}{\mu\eta_k T_k} + \frac{\eta_k}{2}G^2 \overset{(ii)}{=} \frac{\mathbb{E}\Delta_k}{4} + \frac{G^2}{2^{k+2}\mu} \overset{(iii)}{\le} \frac{G^2}{2^{k+1}\mu},$$

with the transitions above following from (i) strong convexity of F, which implies that $V_{x_k^0}(x_\star) = \frac{1}{2}\|x_k^0 - x_\star\|^2 \le \frac{1}{\mu}\Delta_k$; (ii) the choice of parameters ensures $\eta_k T_k = \frac{4}{\mu}$ and $\eta_k = \frac{1}{2^{k+1}\mu}$; and (iii) the inductive hypothesis $\mathbb{E}\Delta_k \le \frac{G^2}{2^k\mu}$. This completes the induction.

Let K be such that the algorithm outputs $x = x_K^0$, and note that $T \le 16\cdot(2^K - 1) - 1$. Therefore, we have

$$\mathbb{E}F(x) - F(x_\star) = \mathbb{E}\Delta_K \le \frac{G^2}{2^K\mu} \le \frac{16G^2}{\mu T},$$

and

$$\mathbb{E}\|x - x_\star\|^2 \le \frac{2}{\mu}(\mathbb{E}F(x) - F(x_\star)) \le \frac{32G^2}{\mu^2 T}.$$

Recalling Definition 1, we conclude that EPOCHSGD is an ODC algorithm with constant $c = 32$. □

Remark 1 (Using EPOCHSGD for optimum estimation)**.** *When using* EPOCHSGD *as the ODC algorithm in our MLMC optimum estimator* (1)*, we need only call once with $T = 2^J$ and take x_0, x_{J-1} and x_J to be the iterates x_1^0, x_{K-1}^0 and x_K^0 of* EPOCHSGD*, for K the last value of k that* EPOCHSGD *reaches.*

B.2 Proof of Theorem 1

Theorem 1. *Let f and $\hat{\nabla} f$ satisfy Assumption 1, $F = f + \psi$ be μ-strongly convex with minimizer $x_\star \in \mathcal{X}$, and $\delta, \sigma > 0$. The function* OPTEST$(\hat{\nabla} f, \psi, \mu, \delta, \sigma^2, \mathcal{X})$ *outputs $\hat{x}_\star$ satisfying*

$$\|\mathbb{E}\hat{x}_\star - x_\star\| \le \delta \quad \textit{and} \quad \mathbb{E}\|\hat{x}_\star - x_\star\|^2 \le \sigma^2$$

using $\mathcal{N}_{\hat{\nabla} f}$ stochastic gradient computations, where

$$\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\left(\frac{G^2}{\mu^2\sigma^2}\log^2\left(\frac{G}{\mu\min\{\delta,\sigma\}}\right) + \log\left(\frac{G}{\mu\min\{\delta,\sigma\}}\right)\right).$$

Proof. Write the algorithm's output as $\hat{x}_\star = \frac{1}{N}\sum_{i=1}^N \hat{x}_\star^{(i)}$ where $\hat{x}_\star^{(1)}, \ldots, \hat{x}_\star^{(N)}$ are independent draws of the estimator (1), with

$$T_{\max} = \left\lceil \frac{(2c)^2G^2}{\mu^2\min\{\delta^2, \frac{1}{2}\sigma^2\}} \right\rceil \text{ and } N = \left\lceil \frac{2(4c)^2G^2}{\mu^2\sigma^2}\log(T_{\max}) \right\rceil$$

as in Algorithm 1. Then, Proposition 1 implies that

$$\|\mathbb{E}\hat{x}_\star^{(1)} - x_\star\| \le \min\left\{\delta, \frac{1}{\sqrt{2}}\sigma\right\} \text{ and } \mathbb{E}\|\hat{x}_\star^{(1)} - \mathbb{E}\hat{x}_\star^{(1)}\|^2 \le \frac{N}{2}\sigma^2.$$

Noting that $\mathbb{E}\hat{x}_\star = \mathbb{E}\hat{x}_\star^{(1)}$ and

$$\mathbb{E}\|\hat{x}_\star - x_\star\|^2 = \frac{1}{N}\mathbb{E}\|\hat{x}_\star^{(1)} - \mathbb{E}\hat{x}_\star^{(1)}\|^2 + \|\mathbb{E}\hat{x}_\star^{(1)} - x_\star\|^2,$$

we obtain the claimed bias and error bounds. Finally, Proposition 1 guarantees that $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O(N\cdot\log(T_{\max}))$, giving the claimed bound on the number of evaluations. □

B.3 Properties of the proximal operator and Moreau envelope

For a convex function $f : \mathcal{X} \to \mathbb{R}$ we recall the definitions of

$$\text{the proximal operator } \mathsf{P}_{f,\lambda}(x) \coloneqq \operatorname*{argmin}_{y\in\mathcal{X}}\left\{f(y) + \tfrac{\lambda}{2}\|y-x\|^2\right\}$$
$$\text{and the Moreau envelope } f_\lambda(x) \coloneqq \min_{y\in\mathcal{X}}\left\{f(y) + \tfrac{\lambda}{2}\|y-x\|^2\right\}.$$

Below, we collect several well-known properties that we use throughout the paper.

Fact 1. *Given a convex function $f : \mathcal{X} \to \mathbb{R}$, and $\lambda > 0$ defined on a closed convex set $\mathcal{X}$, the following properties of the Moreau envelope $f_\lambda : \mathbb{R}^d \to \mathbb{R}$ and the proximal operator $\mathsf{P}_{f,\lambda} : \mathcal{X} \to \mathcal{X}$ hold for all $x \in \mathcal{X}$*

1. Convexity*: f_λ is convex.*
2. Differentiablility*: f_λ is λ-smooth and $\nabla f_\lambda(x) = \lambda(x - \mathsf{P}_{f,\lambda}(x))$.*
3. Approximation*: If f is G-Lipschitz then $f(x) - \frac{G^2}{2\lambda} \le f_\lambda(x) \le f(x)$.*
4. Subgradient*: $\nabla f_\lambda(x) \in \partial f(\mathsf{P}_{f,\lambda}(x))$,*
5. Three point inequality*: for all $u \in \mathcal{X}$:*

$$\langle \nabla f_\lambda(x), \mathsf{P}_{f,\lambda}(x) - u\rangle \le \frac{\lambda}{2}\|u-x\|^2 - \frac{\lambda}{2}\|u - \mathsf{P}_{f,\lambda}(x)\|^2 - \frac{\lambda}{2}\|x - \mathsf{P}_{f,\lambda}(x)\|^2 .$$

See [29, Section 4.1] as well as [13, Lemma 1] and [51, Lemma 1] for proofs and additional background and properties.

C Proofs from Section 3

In this section, we give a proof of Theorem 3. Before we give the technical details, we briefly comment on our algorithm and its analysis. Algorithm 3 is at its core an instantiation of Nesterov's accelerated gradient method applied to the Moreau envelope $f_\lambda(x) = \min_{y\in\mathbb{B}_R(0)}\left\{f(y) + \frac{\lambda}{2}\|y-x\|^2\right\}$. We compute stochastic gradient estimates of f_λ via Algorithm 2, and apply techniques from [1, 2] to bound the accumulated error.

Based on the iterates $\{x_k, v_k\}$ of Algorithm 3, we define

$$E_k = f_\lambda(x_k) - f_\lambda(u), R_k = \frac{1}{2}\|v_k - u\|^2, \text{ and } P_k = k(k+1)E_k + 12\lambda R_k$$

for any fixed $u \in \mathbb{B}_R(0)$. We first prove that (conditioned on the iterates x_{k-1}, v_{k-1}) the potential P_k cannot increase significantly in expectation.

Lemma 4. *Consider an execution of Algorithm 3 with parameters given by Theorem 3. Fix any $u \in \mathbb{B}_R(0)$. For any $k \ge 1$ we have $y_{k-1} \in \mathbb{B}_R(0)$ and*

$$\mathbb{E}\left[P_k | x_{k-1}, v_{k-1}\right] \le P_{k-1} + \epsilon k .$$

Proof. We first remark that $x_{k-1} \in \mathcal{X} \subseteq \mathbb{B}_R(0)$ and $v_{k-1} \in \mathbb{B}_R(0)$ by construction. As a result, $y_{k-1} \in \mathbb{B}_R(0)$ as well. Following [1, 2], we define the function

$$\mathsf{Prog}(y; g) \coloneqq \min_{x\in\mathcal{X}}\left\{\frac{3\lambda}{2}\|x-y\|^2 + \langle g, x-y\rangle\right\}.$$

We observe

$$\begin{aligned}
\mathsf{Prog}(y_{k-1};g_k) &= \min_{x\in\mathcal{X}}\left\{\frac{3\lambda}{2}\|x-y_{k-1}\|^2+\langle g_k, x-y_{k-1}\rangle\right\} \\
&\overset{(i)}{=} \frac{3\lambda}{2}\|x_k-y_{k-1}\|^2+\langle g_k, x_k-y_{k-1}\rangle \\
&= \left(\frac{\lambda}{2}\|x_k-y_{k-1}\|^2+\langle \nabla f_\lambda(y_{k-1}), x_k-y_{k-1}\rangle\right) \\
&\quad +\lambda\|x_k-y_{k-1}\|^2+\langle g_k-\nabla f_\lambda(y_{k-1}), x_k-y_{k-1}\rangle \\
&\overset{(ii)}{\geq} f_\lambda(x_k)-f_\lambda(y_{k-1})+\lambda\|x_k-y_{k-1}\|^2+\langle g_k-\nabla f_\lambda(y_{k-1}), x_k-y_{k-1}\rangle \\
&\overset{(iii)}{\geq} f_\lambda(x_k)-f_\lambda(y_{k-1})-\frac{1}{4\lambda}\|g_k-\nabla f_\lambda(y_{k-1})\|^2.
\end{aligned} \tag{6}$$

Here, we use (i) the definition of x_k, (ii) smoothness of f_λ (Item 2 of Fact 1), and (iii) Young's inequality $\langle a,b\rangle+\frac{1}{2}\|b\|^2\geq-\frac{1}{2}\|a\|^2$ with $a=\frac{1}{2\lambda}(g_k-\nabla f_\lambda(y_{k-1}))$ and $b=2\lambda(x_k-y_{k-1})$. Define the point

$$\widetilde{y}_{k-1}=\frac{k-1}{k+1}x_{k-1}+\frac{2}{k+1}v_k.$$

We observe that

$$y_{k-1}-\widetilde{y}_{k-1}=\left(\frac{k-1}{k+1}x_{k-1}+\frac{2}{k+1}v_{k-1}\right)-\left(\frac{k-1}{k+1}x_{k-1}+\frac{2}{k+1}v_k\right)=\frac{2}{k+1}\left(v_{k-1}-v_k\right).$$

Consequently, we have

$$\begin{aligned}
\frac{k}{6\lambda}\langle g_k, v_{k-1}-u\rangle &= \frac{k}{6\lambda}\langle g_k, v_{k-1}-v_k\rangle+\frac{k}{6\lambda}\langle g_k, v_k-u\rangle \\
&\overset{(i)}{\leq} \frac{k}{6\lambda}\langle g_k, v_{k-1}-v_k\rangle+\frac{1}{2}\left(\|v_{k-1}-u\|^2-\|v_k-u\|^2-\|v_{k-1}-v_k\|^2\right) \\
&\overset{(ii)}{=} \frac{k(k+1)}{12\lambda}\langle g_k, y_k-\widetilde{y}_{k-1}\rangle-\frac{(k+1)^2}{8}\|y_k-\widetilde{y}_{k-1}\|^2+R_{k-1}-R_k \\
&\overset{(iii)}{\leq} \frac{k(k+1)}{12\lambda}\left(\langle g_k, y_{k-1}-\widetilde{y}_{k-1}\rangle-\frac{3\lambda}{2}\|y_k-\widetilde{y}_{k-1}\|^2\right)+R_{k-1}-R_k \\
&\overset{(iv)}{\leq} -\frac{k(k+1)}{12\lambda}\mathsf{Prog}(y_{k-1};g_k)+R_{k-1}-R_k \\
&\overset{(v)}{\leq} \frac{k(k+1)}{12\lambda}\left(f_\lambda(y_{k-1})-f_\lambda(x_k)+\frac{1}{4\lambda}\|g_k-\nabla f_\lambda(y_{k-1})\|^2\right)+R_{k-1}-R_k.
\end{aligned} \tag{7}$$

Here we use (i) the proximal three-point inequality (Item 5 of Fact 1), (ii) the definition of $\widetilde{y}_{k-1}$, (iii) $\frac{(k+1)^2}{8}\geq\frac{3\lambda}{2}\cdot\frac{k(k+1)}{12\lambda}$ and $\|y_{k-1}-\widetilde{y}_{k-1}\|^2\geq 0$, (iv) the definition of Prog, and (v) Equation (6). Thus,

$$\begin{aligned}
\frac{k}{6\lambda}\left(f_\lambda(y_{k-1})-f_\lambda(u)\right) &\leq \frac{k}{6\lambda}\langle\nabla f_\lambda(y_{k-1}), y_{k-1}-u\rangle \\
&\leq \frac{k}{6\lambda}\langle\nabla f_\lambda(y_{k-1}), y_{k-1}-v_{k-1}\rangle+\frac{k}{6\lambda}\langle\nabla f_\lambda(y_{k-1}), v_{k-1}-u\rangle \\
&\overset{(i)}{=} \frac{k(k-1)}{12\lambda}\langle\nabla f_\lambda(y_{k-1}), x_{k-1}-y_{k-1}\rangle+\frac{k}{6\lambda}\langle\nabla f_\lambda(y_{k-1}), v_{k-1}-u\rangle \\
&\leq \frac{k(k-1)}{12\lambda}\left(f_\lambda(x_{k-1})-f_\lambda(y_{k-1})\right)+\frac{k}{6\lambda}\langle\nabla f_\lambda(y_{k-1}), v_{k-1}-u\rangle \\
&\overset{(ii)}{\leq} \frac{k(k-1)}{12\lambda}\left(f_\lambda(x_{k-1})-f_\lambda(y_{k-1})\right)+R_{k-1}-R_k \\
&\quad+\frac{k(k+1)}{12\lambda}\left(f_\lambda(y_{k-1})-f_\lambda(x_k)+\frac{1}{4\lambda}\|g_k-\nabla f_\lambda(y_{k-1})\|^2\right) \\
&\quad+\frac{k}{6\lambda}\langle\nabla f_\lambda(y_{k-1})-g_k, v_{k-1}-u\rangle,
\end{aligned}$$

where we use (i) $y_{k-1} - v_{k-1} = \frac{k-1}{2}(x_{k-1} - y_{k-1})$ and (ii) Equation (7). Rearranging, we obtain

$$\begin{aligned}\frac{1}{12\lambda}\left(P_k - P_{k-1}\right) &= \frac{k(k+1)}{12\lambda}E_k + R_k - \frac{k(k-1)}{12\lambda}E_{k-1} - R_{k-1} \\ &\leq \frac{k(k+1)}{48\lambda^2}\|g_k - \nabla f_\lambda(y_{k-1})\|^2 + \frac{k}{6\lambda}\left\langle \nabla f_\lambda(y_{k-1}) - g_k, v_{k-1} - u\right\rangle \end{aligned} \tag{8}$$

Applying Corollary 2, we observe

$$\mathbb{E}\left[\|g_k - \nabla f_\lambda(y_k)\|^2 | x_{k-1}, v_{k-1}\right] \leq \sigma_k^2 = \frac{2\epsilon\lambda}{k+1}$$

and

$$\mathbb{E}\left[\left\langle \nabla f_\lambda(y_{k-1}) - g_k, v_{k-1} - u\right\rangle | x_{k-1}, v_{k-1}\right] \leq \|\mathbb{E}\left[g_k\right] - \nabla f_\lambda(y_{k-1})\|\|v_{k-1} - u\| \leq 2R\delta_k = \frac{\epsilon}{4}$$

by the Cauchy-Schwarz inequality, the constraint that $u, v_k \in \mathbb{B}_R(0)$, and the choice of parameters σ_k, δ_k. Taking expectations and applying these to Equation (8), we obtain

$$\frac{1}{12\lambda}\left(\mathbb{E}\left[P_k | x_{k-1}, v_{k-1}\right] - P_{k-1}\right) \leq \frac{\epsilon k}{24\lambda} + \frac{\epsilon k}{24\lambda} = \frac{\epsilon k}{12\lambda}.$$

Multiplying both sides by 12λ yields the claim. □

With Lemma 4 in hand, we complete the proof of Theorem 3.

Theorem 3. *Let $f : \mathbb{B}_R(0) \to \mathbb{R}$ and $\hat{\nabla} f$ satisfy Assumption 1. Let $\mathcal{X} \subseteq \mathbb{B}_R(0)$ be a convex set admitting a projection oracle $\mathsf{Proj}_{\mathcal{X}}$. Let $x_0 \in \mathcal{X}$ be an initial point with $\|x - x_\star\| \leq D$ for some $x_\star \in \mathcal{X}$. With $\lambda = \frac{2G^2}{\epsilon}$, $\delta_k = \frac{\epsilon}{8R}$, $\sigma_k^2 = \frac{2\epsilon\lambda}{k+1}$, and $T = \frac{7GD}{\epsilon}$ Algorithm 3 computes $x \in \mathcal{X}$ with $\mathbb{E}\left[f(x)\right] \leq f(x_\star) + \epsilon$ with complexity $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\left(\frac{G^2D^2}{\epsilon^2}\log^2\left(\frac{GR}{\epsilon}\right)\right)$ and $O\left(\frac{GD}{\epsilon}\right)$ calls to $\mathsf{Proj}_{\mathcal{X}}$.*

Proof of Theorem 3. Applying the law of total probability and inductively applying Lemma 4, we obtain

$$\mathbb{E}\left[P_T\right] \leq P_0 + \epsilon\sum_{k=1}^{T} k = P_0 + \frac{\epsilon}{2}T(T+1).$$

We choose $u = x_\star$ and observe $P_T = T(T+1)E_T + 12\lambda R_T \geq T(T+1)\left(f_\lambda(x_T) - f_\lambda(x_\star)\right)$ and $P_0 = 12\lambda R_0 \leq 6\lambda D^2$. Plugging these in, we have

$$\mathbb{E}\left[f_\lambda(x_T)\right] - f_\lambda(x_\star) \leq \frac{6\lambda D^2}{T(T+1)} + \frac{\epsilon}{2}.$$

As f is G-Lipschitz, we apply Item 1 of Fact 1 and our choices of λ and T: this gives

$$\mathbb{E}\left[f(x_T)\right] - f(x_\star) \leq \frac{G^2}{2\lambda} + \frac{6\lambda D^2}{T(T+1)} + \frac{\epsilon}{2} \leq \frac{\epsilon}{4} + \frac{12G^2D^2}{\epsilon T^2} + \frac{\epsilon}{2} \leq \frac{\epsilon}{4} + \frac{12\epsilon}{49} + \frac{\epsilon}{2} < \epsilon$$

as desired.

To finish, we bound the number of oracle queries. The bound on the number of projection oracle calls is immediate since we only call it once per iteration of the algorithm. To bound the number of stochastic gradients needed, we apply Corollary 2 together with the fact that $y_k \in \mathbb{B}_R(0)$ at all times. Thus, we need

$$O\left(\sum_{k=0}^{T-1}\frac{G^2}{\sigma_k^2}\log^2\left(\frac{G}{\delta_k}\right)\right) = O\left(\sum_{k=0}^{T-1}\frac{G^2(k+1)}{\epsilon\lambda}\log^2\left(\frac{GR}{\epsilon}\right)\right) = O\left(\frac{G^2D^2}{\epsilon^2}\log^2\left(\frac{GR}{\epsilon}\right)\right)$$

subgradient computations as desired. □

D Proofs from Section 4

D.1 Analysis of the stochastic accelerated proximal method

In this section we provide a complete analysis of the stochastic accelerated proximal method. We first prove Lemma 5, which shows potential decrease in (conditional) expectation for the iterates of Algorithm 4. Then we give Lemma 6 which provides an in-expectation bound on the potential when the algorithm terminates. In Lemma 7 we give a deterministic error bound resulting from the growth of the A_k sequence. We then combine these ingredients to prove Proposition 2.

Notation. Define the filtration

$$\mathcal{F}_k = \sigma(x_1, v_1, A_1, \zeta_1 \ldots, x_k, v_k, A_k, \zeta_k)$$

where ζ_i is the internal randomness in NEXTLAMBDA(x_i, v_i, A_i). Throughout, we let

$$\hat{x}_k = \mathsf{P}_{f,\lambda_k}(y_{k-1})$$

denote the exact proximal mapping which iteration x_k of the algorithm approximate. We note that $A_{k+1}, y_k, \hat{x}_{k+1}, \in \mathcal{F}_k$, i.e., they are deterministic when conditioned on x_k, v_k, A_k, ζ_k.

For each iteration of Algorithm 4, we obtain the following bound on potential decrease.

Lemma 5. *Let $f : \mathcal{X} \to \mathbb{R}$ satisfy Assumption 1. If $\mathcal{X} \subseteq \mathbb{B}_R(x_0)$, we have*

$$\begin{aligned}\mathbb{E}\big[A_{k+1}(f(x_{k+1}) - f(x_\star)) + \|v_{k+1} - x_\star\|^2 \mid \mathcal{F}_k\big] \\ \leq A_k(f(x_k) - f(x_\star)) + \|v_k - x_\star\|^2 - \frac{1}{6}\lambda_{k+1}A_{k+1}\|\hat{x}_{k+1} - y_k\|^2 \\ + \lambda_{k+1}a_{k+1}^2\varphi_{k+1} + a_{k+1}^2\sigma_{k+1}^2 + 2Ra_{k+1}\delta_{k+1}.\end{aligned}$$

Proof. We let

$$\hat{g}_k = \nabla f_{\lambda_k}(y_{k-1}) = \lambda_k(y_{k-1} - \hat{x}_k)$$

and bound from both sides the quantity $a_{k+1}\langle \hat{g}_{k+1}, v_k - x_\star\rangle$. First, note that

$$v_k - x_\star = \hat{x}_{k+1} - x_\star + \frac{A_k}{a_{k+1}}(\hat{x}_{k+1} - x_k) - \frac{A_{k+1}}{a_{k+1}}(\hat{x}_{k+1} - y_k).$$

Since $\hat{g}_{k+1} \in \partial f(\hat{x}_{k+1})$ (see Item 4 in Fact 1), f is convex and $\langle \hat{g}_{k+1}, \hat{x}_{k+1} - y_k\rangle = -\lambda_{k+1}\|\hat{x}_{k+1} - y_k\|^2$, we have that

$$\begin{aligned}\langle \hat{g}_{k+1}, v_k - x_\star\rangle &= \langle \hat{g}_{k+1}, \hat{x}_{k+1} - x_\star\rangle + \frac{A_k}{a_{k+1}}\langle \hat{g}_{k+1}, \hat{x}_{k+1} - x_k\rangle - \frac{A_{k+1}}{a_{k+1}}\langle \hat{g}_{k+1}, \hat{x}_{k+1} - y_k\rangle \\ &\geq f(\hat{x}_{k+1}) - f(x_\star) + \frac{A_k}{a_{k+1}}(f(\hat{x}_{k+1}) - f(x_k)) - \frac{A_{k+1}}{a_{k+1}}\langle \hat{g}_{k+1}, \hat{x}_{k+1} - y_k\rangle \\ &= \frac{A_{k+1}}{a_{k+1}}(f(\hat{x}_{k+1}) - f(x_\star)) - \frac{A_k}{a_{k+1}}(f(x_k) - f(x_\star)) + \frac{\lambda_{k+1}A_{k+1}}{a_{k+1}}\|\hat{x}_{k+1} - y_k\|^2.\end{aligned}$$

Moreover, by definition of x_{k+1} we have that

$$\mathbb{E}[f(x_{k+1}) \mid \mathcal{F}_k] \leq \mathbb{E}\left[f(x_{k+1}) + \frac{\lambda_{k+1}}{2}\|x_{k+1} - y_k\|^2 \;\middle|\; \mathcal{F}_k\right] \leq f(\hat{x}_{k+1}) + \frac{\lambda_{k+1}}{2}\|\hat{x}_{k+1} - y_k\|^2 + \varphi_{k+1}.$$

Substituting back, we have

$$\begin{aligned}a_{k+1}\langle \hat{g}_{k+1}, v_k - x_\star\rangle \geq &A_{k+1}(\mathbb{E}[f(x_{k+1}) \mid \mathcal{F}_k] - f(x_\star)) - A_k(f(x_k) - f(x_\star)) \\ &+ \frac{\lambda_{k+1}A_{k+1}}{2}\|\hat{x}_{k+1} - y_k\|^2 - A_{k+1}\varphi_{k+1}. \qquad (9)\end{aligned}$$

To upper bound $a_{k+1}\langle \hat{g}_{k+1}, v_k - x_\star\rangle$, note that, since $x_\star \in \mathcal{X}$,

$$\|v_{k+1} - x_\star\|^2 \leq \left\|v_k - \frac{1}{2}a_{k+1}g_{k+1} - x_\star\right\|^2 = \|v_k - x_\star\|^2 - a_{k+1}\langle g_{k+1}, v_k - x_\star\rangle + \frac{a_{k+1}^2}{4}\|g_{k+1}\|^2.$$

Our Moreau Envelope gradient estimtor (see Corollary 2) guarantees that

$$\begin{aligned}\mathbb{E}[\langle g_{k+1}, v_k - x_\star\rangle \mid \mathcal{F}_k] &\geq \langle \hat{g}_{k+1}, v_k - x_\star\rangle - \|\mathbb{E}[g_{k+1} \mid \mathcal{F}_k] - \hat{g}_{k+1}\|\,\|v_k - x_\star\| \\ &\geq \langle \hat{g}_{k+1}, v_k - x_\star\rangle - 2R\delta_{k+1},\end{aligned}$$

and moreover

$$\begin{aligned}\mathbb{E}\left[\|g_{k+1}\|^2 \;\middle|\; \mathcal{F}_k\right] &= \left(1 + \frac{1}{3}\right)\mathbb{E}\|\hat{g}_{k+1}\|^2 + (1+3)\,\mathbb{E}\left[\|g_{k+1} - \hat{g}_{k+1}\|^2 \;\middle|\; \mathcal{F}_k\right] \\ &\leq \frac{4}{3}\|\hat{g}_{k+1}\|^2 + 4\sigma_{k+1}^2.\end{aligned}$$

Combining the last three displays and rearranging, we obtain

$$a_{k+1}\langle \hat{g}_{k+1}, v_k - x_\star\rangle \le \|v_k - x_\star\|^2 - \mathbb{E}\Big[\|v_{k+1} - x_\star\|^2 \;\Big|\; \mathcal{F}_k\Big] + \frac{\lambda_{k+1}^2 a_{k+1}^2}{3}\|\hat{x}_{k+1} - y_k\|^2 + a_{k+1}^2\sigma_{k+1}^2 + 2Ra_{k+1}\delta_{k+1} \tag{10}$$

Combining (9) and (10) and simplifying using $A_{k+1} = \lambda_{k+1}a_{k+1}^2$, we obtain the claimed bound. □

Combining Lemma 5 with the optional stopping theorem, one obtains the following bound on the potential at the final iteration K of the algorithm.

Lemma 6. *Let $K \le K_{\max}$ be the iteration in which Algorithm 4 returns and let*

$$\bar{\varepsilon} \ge \max_{k \le K_{\max}} \left\{\lambda_k a_k \varphi_k + a_k\sigma_k^2 + 2R\delta_k\right\}$$

with probability 1. Then, under the assumptions of Lemma 5, we have

$$\mathbb{E}\left[A_K(f(x_K) - f(x_\star) - \bar{\varepsilon}) + \frac{1}{6}\sum_{i\le K}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2\right] \le A_0(f(x_0) - f(x_\star)) + R^2.$$

Proof. Define $M_k = A_k(f(x_k) - f(x_\star) - \bar{\varepsilon}) + \frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2 + \|v_k - x_\star\|^2$ for all $k \in [K]$. We argue that it is a supermartingale adapted to filtration $\mathcal{F}_k$. Clearly, $\mathbb{E}[|M_k|] < \infty$ for each k due to boundedness of f, K, λ_i and A_i. It therefore suffices to show that $\mathbb{E}[M_{k+1}|F_k] \le M_k$ for all $k+1 \in [K]$. By Lemma 5 we have

$$\begin{aligned}\mathbb{E}\left[M_{k+1}|F_k\right] &\le A_k(f(x_k) - f(x_\star)) + \|v_k - x_\star\|^2 - \frac{1}{6}\lambda_{k+1}A_{k+1}\|\hat{x}_{k+1} - y_k\|^2 \\ &\quad + \lambda_{k+1}a_{k+1}^2\varphi_{k+1} + a_{k+1}^2\sigma_{k+1}^2 + 2Ra_{k+1}\delta_{k+1} - A_{k+1}\bar{\varepsilon} + \frac{1}{6}\sum_{i\le k+1}\lambda_i A_i\|\hat{x}_{i+1} - y_i\|^2 \\ &\le A_k(f(x_k) - f(x_\star) - \bar{\varepsilon}) + \|v_k - x_\star\|^2 + \frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_{i+1} - y_i\|^2 = M_k,\end{aligned}$$

where the second inequality used the definition of $\bar{\varepsilon}$ and $A_{k+1} = A_k + a_{k+1}$ for the second inequality. This completes the proof that M_k being a supermartingale adapted to filtration $\mathcal{F}_k$.

Now note K is a stopping time adapted to $\mathcal{F}_k$ as it only depends on A_{k+1}. Also, K as a random variable is finitely bounded by $K_{\max}$ with probability 1. Thus, by optional stopping theorem for supermartingale [26], we have

$$\mathbb{E}M_K \le M_0 = A_0(f(x_0) - f(x_\star) - \bar{\varepsilon}) + \|v_0 - x_\star\|^2 \le A_0(f(x_0) - f(x_\star)) + R^2.$$

□

Further, following a similar argument to Carmon et al. [12, 13], we obtain a deterministic growth bound on the coefficients A_k.

Lemma 7. *Fix $k > 0$ and let*

$$T_\lambda = \sum_{i\le k}\mathbb{1}_{\{\lambda_i < 2\lambda_{\min}\}} \;\text{ and }\; T_r = \sum_{i\le k}\mathbb{1}_{\{\|\hat{x}_i - y_{i-1}\| \ge 3r/4\}}$$

count the number of times $\lambda_i < 2\lambda_{\min}$ and $\|\hat{x}_i - y_{i-1}\| \ge 3r/4$, respectively. Then, the following holds with probability 1,

$$\frac{1}{A_k}\left(9R^2 - \frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2\right) \le O\left(\min\left\{\frac{\lambda_{\min}R^2}{T_\lambda^2}, \frac{R^2}{A_0}\exp\left(-\Omega(1)\frac{r^{2/3}}{R^{2/3}}T_r\right)\right\}\right).$$

Proof. When $9R^2 - \frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2 \le 0$, the inequality holds true trivially. Thus, we only consider the case when $\frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2 \le 9R^2$. Consider the following iterate index subsets

$$\mathcal{I}_\lambda \coloneqq \{i \le k : \lambda_i < 2\lambda_{\min}\}$$

and, for $t \le k$

$$\mathcal{I}_{r,t} \coloneqq \{i \le t : \|\hat{x}_i - y_{i-1}\| \ge 3r/4\}.$$

We first show that

$$\frac{1}{A_k}\left(9R^2 - \frac{1}{6}\sum_{i\le k}\lambda_i A_i \|\hat{x}_i - y_{i-1}\|^2\right) \le O\Big(\frac{R^2}{A_0}\exp\Big(-\Omega(1)\frac{r^{2/3}}{R^{2/3}}T_r\Big)\Big). \tag{11}$$

To see this, observe that for any $t \le k$ by definition of $\mathcal{I}_{r,t}$,

$$\frac{1}{6}\sum_{i\in\mathcal{I}_{r,t}}\lambda_i A_i \cdot \left(\frac{9}{16}r^2\right) \le \frac{1}{6}\sum_{i\le t}\lambda_i A_i \|\hat{x}_i - y_{i-1}\|^2 \le 9R^2,$$

which by rearranging terms implies

$$\sum_{i\in\mathcal{I}_{r,t}}\lambda_i A_i \le \frac{96R^2}{r^2}. \tag{12}$$

Note the reverse Hölder's inequality with $p = 2/3$ states that for any $u, v \in \mathbb{R}^d_{>0}$,

$$\langle u, v\rangle \ge \left(\sum_{i\in[d]} u_i^{2/3}\right)^{3/2} \cdot \left(\sum_{i\in[d]} v_i^{-2}\right)^{-1/2}.$$

We have

$$\begin{aligned}\sqrt{A_t} &\overset{(i)}{\ge} \frac{1}{2}\sum_{i\in\mathcal{I}_{r,t}}\frac{1}{\sqrt{\lambda_i}} \overset{(ii)}{\ge} \frac{1}{2}\left(\sum_{i\in\mathcal{I}_{r,t}}\left(\sqrt{A_i}\right)^{2/3}\right)^{3/2} \cdot \left(\sum_{i\in\mathcal{I}_{r,t}}\left(\frac{1}{\sqrt{A_i\lambda_i}}\right)^{-2}\right)^{-1/2} \\ &\overset{(iii)}{\ge} \frac{r}{8\sqrt{6}R} \cdot \left(\sum_{i\in\mathcal{I}_{r,t}}\left(\sqrt{A_i}\right)^{2/3}\right)^{3/2},\end{aligned}$$

where we used (i) Lemma 23 of [12] and $\mathcal{I}_{r,t} \subseteq [t]$, (ii) the reverse Hölder's inequality with $u_i = \sqrt{A_i}$, and $v_i = 1/\sqrt{A_i\lambda_i}$, and (iii) the bound (12). Rearranging, we have

$$A_t^{1/3} \ge \frac{r^{2/3}}{4\sqrt[3]{6}R^{2/3}}\left(\sum_{i\in\mathcal{I}_{r,t}} A_i^{1/3}\right), \quad \text{for all } \ t \le k, \tag{13}$$

which by applying Lemma 32 of [12] and noting that $T_r = |\mathcal{I}_{r,k}|$ gives

$$A_k^{1/3} \ge \exp\left(\frac{r^{2/3}}{4\sqrt[3]{6}R^{2/3}}T_r\right)A_0^{1/3},$$

and thus

$$\frac{1}{A_k}\left(9R^2 - \frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2\right) \le \frac{9R^2}{A_k} \le O\Big(\frac{R^2}{A_0}\exp\Big(-\Omega(1)\frac{r^{2/3}}{R^{2/3}}T_r\Big)\Big).$$

Next, we show that

$$\frac{1}{A_k}\left(9R^2 - \frac{1}{6}\sum_{i\le k}\lambda_i A_i\|\hat{x}_i - y_{i-1}\|^2\right) \le O\Big(\frac{\lambda_{\min}R^2}{T_\lambda^2}\Big). \tag{14}$$

Using Lemma 23 of [12] again, along with $\mathcal{I}_\lambda \subseteq [k]$ and $|\mathcal{I}_\lambda| = T_\lambda$, we have

$$\sqrt{A_k} \ge \frac{1}{2}\sum_{i\in\mathcal{I}_\lambda}\frac{1}{\sqrt{\lambda_i}} \ge \frac{T_\lambda}{2\sqrt{2\lambda_{\min}}}.$$

Rearranging the terms, we see that $1/A_k \le O(\lambda_{\min}/T_\lambda^2)$ as desired.

Combining Equations (11) and (14) we obtain the claimed bound. □

Putting these pieces together gives Proposition 2, which we prove below.

Proposition 2. *Let $f : \mathcal{X} \to \mathbb{R}$ and $\hat{\nabla} f$ satisfy Assumption 1, and let $\mathcal{X} \subseteq \mathbb{B}_R(x_0)$. For a target accuracy $\epsilon \leq GR$ let $\varphi_{k+1} = \frac{\epsilon}{60\lambda_{k+1}a_{k+1}}$, $\delta_{k+1} = \frac{\epsilon}{120R}$, $\sigma_{k+1}^2 = \frac{\epsilon}{60a_{k+1}}$, $A_0 = \frac{R}{G}$, and $A_{\max} = \frac{9R^2}{\epsilon}$. If $\lambda_k \geq \lambda_{\min} \geq \frac{1}{A_{\max}} = \Omega(\frac{\epsilon}{R^2})$ for all $k \leq K_{\max}$, then lines 4 and 5 of Algorithm 4 have total complexity $\mathbb{E}\mathcal{N}_{\hat{\nabla} f} = O\Big(K_{\max} \log \frac{GR}{\epsilon} + \frac{G^2R^2}{\epsilon^2} \log^2 \frac{GR}{\epsilon}\Big)$. If in addition $\|\mathsf{P}_{f,\lambda_k}(y_{k-1}) - y_{k-1}\| \geq 3r/4$ whenever $\lambda_k \geq 2\lambda_{\min}$ then for $K_{\max} = O\Big(\big(\frac{R}{r}\big)^{2/3} \log\big(\frac{GR}{\epsilon}\big) + \sqrt{\frac{\lambda_{\min}R^2}{\epsilon}}\Big)$, the algorithm's output x_K satisfies $f(x_K) - f(x_\star) \leq \epsilon$ with probability at least $\frac{2}{3}$.*

Proof. First, let us prove correctness of the algorithm. The settings of φ_k, δ_k and σ_k in the proposition guarantee that

$$\max_{k \leq K_{\max}} \big\{\lambda_k a_k \varphi_k + a_k \sigma_k^2 + 2R\delta_k\big\} = \frac{\epsilon}{20}.$$

Therefore, Lemma 6 with $\bar{\varepsilon} = \epsilon/20 \leq R^2/(2.2A_{\max})$ yields

$$\mathbb{E}\left[A_K(f(x_K) - f(x_\star)) + \frac{1}{6}\sum_{i \leq K} \lambda_i A_i \|\hat{x}_i - y_{i-1}\|^2\right] \leq R^2 + \bar{\varepsilon} \cdot \mathbb{E}A_K + A_0(f(x_0) - f(x_\star)).$$

Note that $A_{K-1} \leq A_{\max}$ by definition. Therefore, $\lambda_{\min} \geq \frac{1}{A_{\max}}$ implies that

$$a_K = \frac{1}{2}\sqrt{\frac{1}{\lambda_K^2} + \frac{4A_{K-1}}{\lambda_K}} \leq \frac{\sqrt{5}}{2} A_{\max},$$

and therefore $A_K \leq 2.2A_{\max} \leq R^2/\bar{\varepsilon}$ with probability 1. Moreover the choice of $A_0 = R/G$ and the fact that f is G Lipschitz imply that $A_0(f(x_0) - f(x_\star)) \leq R^2$. Therefore, $\mathbb{E}\Big[A_K(f(x_K) - f(x_\star)) + \frac{1}{6}\sum_{i \leq K} \lambda_i A_i \|\hat{x}_i - y_{i-1}\|^2\Big]$ is at most $3R^2$. Since the term in the expectation is non-negative, we conclude that with probability at least $2/3$ it is bounded by $9R^2$, which implies

$$f(x_K) - f(x_\star) \leq \frac{1}{A_K}\left(9R^2 - \frac{1}{6}\sum_{i \leq K} \lambda_i A_i \|\hat{x}_i - y_{i-1}\|^2\right).$$

If $A_K \geq A_{\max} = 9R^2/\epsilon$ we are done. Otherwise, $K = K_{\max}$ and by the assumption on NEXTLAMBDA we have $T_\lambda + T_r \geq K_{\max}$ for T_λ and T_r defined in Lemma 7. Therefore, either $T_r \geq K_{\max}/2$ or $T_\lambda \geq K_{\max}/2$, and in either case taking $K_{\max} = O\left(\big(\frac{R}{r}\big)^{2/3} \log\big(\frac{GR}{\epsilon}\big) + \sqrt{\frac{\lambda_{\min}R^2}{\epsilon}}\right)$ and applying Lemma 7 yields $f(x_K) - f(x_\star) \leq \epsilon$ and establishing correctness.

Next, let us prove the stated complexity bound. We note each step of computing x_k in Line 4 requires $O(G^2/\lambda_k\varphi_k)$ queries via Proposition 3 and the definition (4) of the approximate proximal mapping. Moreover, by Corollary 2 computing g_k in Line 5 requires

$$O\left(\log\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right) + \frac{G^2}{\sigma_k^2}\log^2\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right)\right)$$

queries in expectation. Summing over $k \in [K]$ and substituting $\varphi_k, \delta_k, \sigma_k$, we obtain

$$\begin{aligned}
\mathbb{E}\mathcal{N}_{\hat{\nabla} f} &= \sum_{k \in [K]} O\left(\frac{G^2}{\lambda_k \varphi_k}\right) + \sum_{k \in [K]} O\left(\log\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right) + \frac{G^2}{\sigma_k^2}\log^2\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right)\right) \\
&= \sum_{k \in [K]} O\left(\log\left(\frac{GR}{\epsilon}\right) + \frac{a_k G^2}{\epsilon}\log^2\left(\frac{GR}{\epsilon}\right)\right) = O\left(\log\left(\frac{GR}{\epsilon}\right) \cdot K + \frac{A_K G^2}{\epsilon}\log^2\left(\frac{GR}{\epsilon}\right)\right) \\
&= O\left(K_{\max}\log\left(\frac{GR}{\epsilon}\right) + \frac{G^2R^2}{\epsilon^2}\log^2\left(\frac{GR}{\epsilon}\right)\right),
\end{aligned}$$

where we have used $A_K = O(A_{\max}) = O(R^2/\epsilon)$ once more. □

D.2 Minimizing the maximum of N functions

In this section, we first revisit the problem setup of minimizing the maximum of N functions and reintroduce key notation. Then we provide the procedure of estimating the gradient of the softmax using rejection sampling in Algorithm 9 and prove its guarantees in Lemma 8. Next, we bound the query complexity of lines 4 and 5 of Algorithm 4 in Lemmas 9 and 10 respectively. Citing [13], we provide a bisection procedure in Algorithm 10 and state its guarantee in Lemma 4. For this procedure we use the Ball Regularization Optimization Oracle (BROO) implementation of [13]; see Definition 4 and Lemma 11. Combining these components with the developments of the previous subsection, we prove Theorem 4.

Notation. Consider the problem of approximately minimizing the maximum of N convex functions: given $f_{(i)}$ such that for every $i \in [N]$ the function $f_{(i)} : \mathbb{R}^d \to \mathbb{R}$ is convex, G-Lipschitz, with a subgradient oracle $\nabla f_{(i)}$ and a target accuracy ϵ we wish to

$$\text{find a point } x \text{ such that } \; f_{\max}(x) - \inf_{x_\star \in \mathbb{R}^d} f_{\max}(x_\star) \le \epsilon \;\text{ where }\; f_{\max}(x) \coloneqq \max_{i \in [N]} f_{(i)}(x) \,. \tag{15}$$

A common approach to solving this problem is to consider the following "softmax" approximation of $f_{\max}$,

$$f_{\mathrm{smax}}(x) \coloneqq \epsilon' \log\left(\sum_{i \in [N]} e^{f_{(i)}(x)/\epsilon'}\right), \;\text{ where } \epsilon' = \frac{\epsilon}{2 \log N}. \tag{16}$$

It is straightforward to show that $0 \le f_{\mathrm{smax}}(x) - f_{\max}(x) \le \frac{\epsilon}{2}$ for all $x \in \mathbb{R}^d$, and that the subgradients of f_{smax} are of the form

$$\nabla f_{\mathrm{smax}}(x) = \sum_{i \in [N]} p_i(x) \nabla f_{(i)}(x) \;\text{ where }\; p_i(x) = \frac{e^{f_{(i)}(x)/\epsilon'}}{\sum_{j \in [N]} e^{f_{(j)}(x)/\epsilon'}} \tag{17}$$

for $\nabla f_{(i)}(x) \in \partial f_{(i)}(x)$ for all $i \in [N]$. The small radius

$$r_\epsilon \coloneqq \frac{\epsilon'}{G} = \frac{\epsilon}{2G \log N}$$

plays a key role in our analysis, since—as we now discuss in detail—this is a domain size where we can efficiently minimize f_{smax} using stochastic gradient methods.

D.2.1 Gradient estimation via rejection sampling

We first construct the gradient estimator of $f_{\mathrm{smax}}(x)$ using rejection sampling. The high-level idea of the technique is as follows. Given a ball $\mathbb{B}_{r_\epsilon}(\bar{x})$ where $r_\epsilon = \epsilon'/G$, Lipschitz continuity of $f_{(i)}$ implies

$$\frac{\left|f_{(i)}(x) - f_{(i)}(\bar{x})\right|}{\epsilon'} \le \frac{G r_\epsilon}{\epsilon'} = 1. \tag{18}$$

As a result, we can perform a full data pass *once* to compute $p(\bar{x})$, and use it to sample from $p(x)$ at nearby points $x \in \mathbb{B}_{r_\epsilon}(\bar{x})$ via rejection sampling. In particular, we draw i from $p(\bar{x})$ and accept it with probability $q_{\mathrm{accept}} = \exp(f_{(i)}(x)/\epsilon' - f_{(i)}(\bar{x})/\epsilon' - 1)$, and otherwise repeat the process. The the bound (18) guarantees that $q_{\mathrm{accept}} < 1$ (so it is indeed a probably), and therefore the output i has distribution p. The bound (18) also guarantees that $q_{\mathrm{accept}} = \Omega(1)$ and consequently that the query complexity of the procedure is $O(1)$. We sate the procedure formally in Algorithm 9 and give its guarantees in Lemma 8.

Lemma 8 (Rejection sampling)**.** *Given G-Lipschitz functions $f_{(i)}$ and $\bar{p} = p(\bar{x})$, $\forall i \in [N]$, the procedure* SoftmaxGradEst *with input $x \in \mathbb{B}_{r_\epsilon}(\bar{x})$ returns a vector $\hat{\nabla} f_{\mathrm{smax}}(x)$ such that $\mathbb{E}\hat{\nabla} f_{\mathrm{smax}}(x) \in \partial f_{\mathrm{smax}}(x)$ and $\|\hat{\nabla} f_{\mathrm{smax}}(x)\| \le G$. The procedure has complexity $\mathbb{E}\mathcal{N}_{f_{(i)}} = O(1)$ and $\mathcal{N}_{\partial f_{(i)}} = 1$.*

Proof. We first prove correctness. Note that G-Lipschitz continuity of the $f_{(i)}$'s along with $\|x - \bar{x}\| \le r_\epsilon = \epsilon'/G$ guarantees that, $f_{(i)}(x)\epsilon' - f_{(i)}(\bar{x})/\epsilon' \le 1$ and therefore $q_{\mathrm{accept}} \le 1$ is a valid probability of every value of i. Therefore, the probability of sample and accepting i is proportional to

$$\bar{p}_i \cdot \exp\left(\frac{f_{(i)}(x)}{\epsilon'} - \frac{f_{(i)}(\bar{x})}{\epsilon'}\right) \propto \exp\left(\frac{f_{(i)}(x)}{\epsilon'}\right) \propto p_i(x),$$

Algorithm 9: SOFTMAXGRADEST$(\{f_{(i)}\}, \{\bar{p}_i\}, \bar{x}, x)$

Input: Functions $f_{(i)}$, pre-computed $f_{(i)}(\bar{x})$ and $\bar{p}_i = p_i(\bar{x})$ for $i \in [N]$, query point $x \in \mathbb{B}_{r_\epsilon}(\bar{x})$.

Output: An unbiased estimator for $\nabla f_{\max}$ with norm at most G.

Loop
  Sample i from $\bar{p}$
  Let $q_{\text{accept}} = \exp(f_{(i)}(x)/\epsilon' - f_{(i)}(\bar{x})/\epsilon' - 1)$
  Draw $A \sim \mathsf{Bernoulli}(q_{\text{accept}})$
  if $A = 1$ **then return** $\nabla f_{(i)}(x)$

which proves the correctness of the sampling distribution for the output i and, via eq. (17), the unbiasedness of the gradient estimator. The norm bound on the output of the procedure is immediate from Lipschitzness of $f_{(i)}$.

Next, we prove the complexity bound. Clearly, the algorithm only queries a single subgradient at termination. To bound the number of function value queries, note that Lipschitz continuity and the ball radius imply $f_{(i)}(x)\epsilon' - f_{(i)}(\bar{x})/\epsilon' \geq -1$, and therefore the probability of acceptance is at least e^{-2}. Consequently, the expected number of iterations before accepting a sample is at most $e^2 = O(1)$. □

D.2.2 Estimating the proximal mapping and Moreau envelope gradient

Using gradient estimator for $\hat{\nabla} f_{\max}$ developed above, we can implement lines 4 and 5 in Algorithm 4, provided that the true proximal bound $\hat{x} = \mathsf{P}_{\lambda, f_{\max}}(y)$ satisfies $\|\hat{x} - y\| \leq r = r_\epsilon$. We begin the implementation of the approximate proximal step in line 4, which we obtain by directly applying EPOCHSGD. The following is an immediate consequence of Lemma 8 and Proposition 3.

Lemma 9. *Let $f_{(i)}$ be convex and G-Lipschitz for all $i \in [N]$, let $\epsilon, \varphi > 0$ and $r_\epsilon = \epsilon/(2 \log GN)$. For any $\bar{x} \in \mathbb{R}^d$ and $\lambda > 0$, if $\mathsf{P}_{f_{\max},\lambda}(\bar{x}) \in \mathbb{B}_{r_\epsilon}(\bar{x})$ then* EPOCHSGD$(\hat{\nabla} f_{\max}, \frac{\lambda}{2}\|\cdot - \bar{x}\|, \lambda, \mathcal{X} \cap \mathbb{B}_{r_\epsilon}(\bar{x}), \lceil 16G^2/(\lambda\varphi) \rceil)$ *(with $\hat{\nabla} f_{\max}$ implemented with Algorithm 9) outputs a valid point $\widetilde{\mathsf{P}}^{\varphi}_{f_{\max},\lambda}(\bar{x})$, and has complexity*

$$\mathbb{E}\mathcal{N}_{f_{(i)}} = O\left(N + \frac{G^2}{\lambda\varphi}\right) \text{ and } \mathcal{N}_{\partial f_{(i)}} = O\left(\frac{G^2}{\lambda\varphi}\right).$$

Similarly combining Lemma 1 with Corollary 2, one can also obtain the following expected oracle complexity guarantee for estimating the Moreau envelope gradient .

Lemma 10. *Let $f_{(i)}$ be convex and G-Lipschitz for all $i \in [N]$, let $\sigma, \epsilon, \delta > 0$ and $r_\epsilon = \epsilon/(2 \log N \cdot G)$. For any $\bar{x} \in \mathbb{R}^d$ and $\lambda > 0$, if $\mathsf{P}_{f_{\max},\lambda}(\bar{x}) \in \mathbb{B}_{r_\epsilon}(\bar{x})$ then $\hat{g} =$* MORGRADEST$(\hat{\nabla} f_{\max}, \lambda, \bar{x}, \delta, \sigma^2, \mathcal{X} \cap \mathbb{B}_{r_\epsilon}(\bar{x}))$ *(with $\hat{\nabla} f_{\max}$ implemented with Algorithm 9) is an estimator of the Moreau envelope gradient $\nabla f_{\max,\lambda}(\bar{x})$ with bias at most δ and expected square error at most σ^2. Its complexity is*

$$\mathbb{E}\mathcal{N}_{f_{(i)}} = O\left(N + \frac{G^2}{\sigma^2}\log^2\left(\frac{G}{\min\{\delta,\sigma\}}\right) + \log\left(\frac{G}{\min\{\delta,\sigma\}}\right)\right)$$
$$\mathbb{E}\mathcal{N}_{\partial f_{(i)}} = O\left(\frac{G^2}{\sigma^2}\log^2\left(\frac{G}{\min\{\delta,\sigma\}}\right) + \log\left(\frac{G}{\min\{\delta,\sigma\}}\right)\right).$$

D.2.3 Implementing NEXTLAMBDA via bisection

The third and final component in our algorithm is an implementation of the subroutine NEXTLAMBDA in line 2 of Algorithm 4 that guarantees the following things on λ_{k+1} and $\hat{x}_{k+1} = \mathsf{P}_{f_{\max},\lambda}(y_k)$: (i) that $\|\hat{x}_{k+1} - y_k\| \leq r$ and (ii) either $\|\hat{x}_{k+1} - y_k\| \geq 3r/4$ or $\lambda < 2\lambda_{\min}$; we later set $r = r_\epsilon$ and $\lambda_{\min} = \widetilde{O}(\epsilon/(r_\epsilon^{4/3} R^{2/3}))$ but for the development of the bisection procedure we keep them general.

Our implementation of NEXTLAMBDA is identical to the one in [13], and we reproduce it here for completeness.

We start by introducing the notion of a *Ball Regularization Optimization Oracle* (BROO).

Definition 4 ([13, Definition 1]). *We say that a mapping $\mathcal{O}_{\lambda,\rho}(\cdot) : \mathcal{X} \to \mathcal{X}$ is a Ball Regularized Optimization Oracle of radius r (r-BROO) for f, if for every query point $\bar{x}$, regularization parameter λ and desired accuracy ρ, it return $\tilde{x} = \mathcal{O}_{\lambda,\rho}(\bar{x})$ satisfying*

$$f(\tilde{x}) + \frac{\lambda}{2}\|\tilde{x} - \bar{x}\|^2 \leq \min_{x \in \mathbb{B}_r(\bar{x}) \cap \mathcal{X}} \left\{ f(x) + \frac{\lambda}{2}\|x - \bar{x}\|^2 \right\} + \frac{\lambda}{2}\rho^2. \tag{19}$$

While a BROO is quite similar to the approximate proximal mapping $\widetilde{\mathsf{P}}^{\varphi}_{\lambda}$, there are two important differences. First, in the BROO definition we constrain the minimization to $\mathbb{B}_r(\bar{x})$ where the approximate proximal mapping is defined for the all domain—this allows us to efficiently compute a BROO via stochastic methods even for values of λ where the true (unconstrained) proximal point is far from $\bar{x}$. Second, we require the sub-optimality guarantee to hold deterministically (a requirement that we will satisfy with high probability), as opposed the requirement (4) of an expected suboptimality bound. In addition, note that the accuracy parameter φ and ρ are related via $\varphi = \lambda\rho^2/2$ and that ρ has units of distance. Strong convexity of the BROO optimization objective then implies that $\|\mathcal{O}_{\lambda,\rho}(x) - \mathsf{P}_{f,\lambda}(x)\| \leq \rho$ whenever $\mathsf{P}_{f,\lambda}(x) \in \mathbb{B}_r(x)$.

We have the following high-probability complexity guarantee for implementing a BROO.

Lemma 11 ([13, Corollary 1]). *Let $f_{(i)}$ be convex and G-Lipschitz for all $i \in [N]$, let $p_{\mathrm{f}} \in (0,1)$, $\epsilon, \rho > 0$ and $r_\epsilon = \epsilon/(2\log N \cdot G)$. For any $\bar{x} \in \mathbb{R}^d$ and $\lambda \leq O(G/r_\epsilon)$, with probability at least $1 - p_{\mathrm{f}}$,* EPOCH-SGD-PROJ *[13, Algorithm 2] that outputs a valid r_ϵ-BROO response for f_{smax} to query $\bar{x}$ with regularization λ and accuracy ρ, and has complexity*

$$\mathcal{N}_{f_{(i)}} = O\left(N + \frac{G^2}{\lambda^2\rho^2}\log\left(\frac{\log(G/(\lambda\rho))}{p_{\mathrm{f}}}\right)\right) \text{ and } \mathcal{N}_{\partial f_{(i)}} = O\left(\frac{G^2}{\lambda^2\rho^2}\log\left(\frac{\log(G/(\lambda\rho))}{p_{\mathrm{f}}}\right)\right) \tag{20}$$

Given a BROO implementation Algorithm 10 outputs values of λ meeting the requirements of Proposition 2. The algorithm and the formal guarantee below are reproduced from [13] for completeness, and we refer the reader to Appendix B.3 of that paper for additional description and discussion.

Algorithm 10: λ-BISECTION(x, v, A)

Input: Points $x, v \in \mathcal{X}$, scalar $A \geq 0$.

Parameters: BROO $\mathcal{O}_{\lambda,\delta}(\cdot)$ (see Definition 4), bisection bounds $\lambda_{\min}, \lambda_{\max}$, Lipschitz bound G, distance bounds R and r.

For all λ', let $y_{\lambda'} \coloneqq \alpha_{2A\lambda'} \cdot x + (1 - \alpha_{2A\lambda'}) \cdot v$, where $\alpha_\tau \coloneqq \frac{\tau}{1+\tau+\sqrt{1+2\tau}}$

Define $\Delta(\lambda) \coloneqq \|\mathcal{O}_{\lambda,\frac{r}{17}}(y_\lambda) - y_\lambda\|$ ▷ approximation of ball optimizer to y_λ

Let $\lambda = \lambda_{\max}$

while $\lambda \geq \lambda_{\min}$ **and** $\Delta(\lambda) \leq \frac{13r}{16}$ **do** $\lambda \leftarrow \lambda/2$ ▷ terminates in $O(\log \frac{\lambda_{\max}}{\lambda_{\min}})$ steps

if $\lambda \leq \lambda_{\min}$ **then return** 2λ ▷ happens only if ball optimizer is $O(\epsilon)$-optimal

Let $\lambda_u = 2\lambda$, $\lambda_\ell = \lambda$ and $\lambda_m = \sqrt{\lambda_u\lambda_\ell}$

if $\Delta(\lambda_\ell) \leq \frac{15r}{16}$ **then return** λ_ℓ ▷ happens only if $\Delta(\lambda_\ell) \in [\frac{13r}{16}, \frac{15r}{16}]$

while $\Delta(\lambda_m) \notin [\frac{13r}{16}, \frac{15r}{16}]$ **and** $\log_2 \frac{\lambda_u}{\lambda_\ell} \geq \frac{r}{8(R+G/\lambda_\ell)}$ **do**

  if $\Delta(\lambda_m) < \frac{13r}{16}$ **then** $\lambda_u = \lambda_m$ **else** $\lambda_\ell = \lambda_m$

  $\lambda_m = \sqrt{\lambda_u\lambda_\ell}$

return λ_m ▷ the while loop terminates in $O\big(\log\big(\frac{R}{r} + \frac{G}{\lambda_{\min} r}\big)\big)$ steps

Proposition 4 ([13, Proposition 2]). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be G-Lipschitz and convex, and let $x, v \in \mathbb{R}^d$, $\epsilon, r, R \in \mathbb{R}_{>0}$ satisfy $\epsilon \leq GR$, $r \leq R$ and $\|x - v\| \leq 2R$. Given $\lambda_{\max} \geq \frac{2G}{r}$ and $\lambda_{\min} \in (0, \lambda_{\max})$, λ-BISECTION(x, v, A) outputs $\lambda \in [\lambda_{\min}, \lambda_{\max}]$ such that*

$$\|\mathsf{P}_{f,\lambda}(y_\lambda) - y_\lambda\| \leq r.$$

The subroutine uses $O(\log(\frac{\lambda_{\max}}{\lambda_{\min}}) + \log(\frac{R+G/\lambda_{\min}}{r}))$ calls to $\mathcal{O}_{\lambda', \frac{r}{17}}(\cdot)$ with $\lambda' \in [\frac{1}{2}\lambda, \lambda_{\max}]$. Moreover, for $\alpha_{2\lambda A} = \frac{2\lambda A}{1+2\lambda A+\sqrt{1+4\lambda A}}$ and $y_\lambda \coloneqq \alpha_{2\lambda A}x + (1-\alpha_{2\lambda A})v$ one of the following outcomes must occur:

1. *$\lambda \in [2\lambda_{\min}, \lambda_{\max}]$ and $\|\mathsf{P}_{f,\lambda}(y_\lambda) - y_\lambda\| > \frac{3r}{4}$, or*
2. *$\lambda < 2\lambda_{\min}$.*

When taking $\lambda_{\max} = \frac{2G}{\epsilon}$ and $\lambda_{\min} = \Omega(\frac{\epsilon}{rR})$, the number of calls to $\mathcal{O}_{\lambda', \frac{r}{17}}(\cdot)$ is $O(\log \frac{GR^2}{r\epsilon})$.

D.2.4 Proof of Theorem 4

Finally, we combine the guarantees collected above to prove our near-optimal rate for minimizing the maximum-loss.

Theorem 4. *Let $f_{(1)}, \ldots, f_{(N)} : \mathcal{X} \to \mathbb{R}$ be convex and G-Lipschitz and let $\mathcal{X} \subseteq \mathbb{B}_R(x_0)$. For any $\epsilon < \frac{1}{2}GR/\log N$, Algorithm 4 (with $\widetilde{\mathsf{P}}^{\varphi}_{f_{\max},\lambda}$ implemented in Algorithm 8, $\hat{\nabla} f_{\max}$ given by Algorithm 9 and* NEXTLAMBDA *given by Algorithm 10 with $\lambda_{\min} = \widetilde{O}(\epsilon/(r_\epsilon^{4/3}R^{2/3}))$ outputs $x \in \mathcal{X}$ that with probability at least $\frac{1}{2}$ is ϵ-suboptimal for $f_{\max}(x) = \max_{i\in[N]} f_{(i)}(x)$ and has complexity $\mathbb{E}\mathcal{N}_{f_{(i)}} = O\Big(\Big[N\Big(\frac{GR\log N}{\epsilon}\Big)^{2/3} + \big(\frac{GR}{\epsilon}\big)^2\Big]\log^2\frac{GR}{\epsilon}\Big)$ and $E\mathcal{N}_{\partial f_{(i)}} = O\Big(\big(\frac{GR}{\epsilon}\big)^2\log^2\frac{GR}{\epsilon}\Big)$.*

Proof. We first prove correctness. Since $\epsilon' = \frac{\epsilon}{2\log N}$, we have $0 \le f_{\max}(x) - f_{\max}(x) \le \epsilon/2$ [see, e.g., 12, Lemma 45]. Therefore, it suffices to find an $\epsilon/2$-approximate solution of $f_{\max}$ over the domain $\mathcal{X} \subseteq \mathbb{B}_R(x_0)$. Let p_{BROO} be the probability that all the BROO calls within Algorithm 10 (implemented as described in Lemma 11) result in a valid output. Then, noting that y_λ defined in Proposition 4 is precisely y_k defined in Algorithm 4, the guarantees of Proposition 4 imply that, for $\hat{x}_{k+1} = \mathsf{P}_{f_{\max},\lambda}(y_k)$, we have $\|\hat{x}_{k+1} - y_k\| \le r$ and either $\|\hat{x}_{k+1} - y_k\| \ge 3r/4$ or $\lambda < 2\lambda_{\min}$ with probability at least p_{BROO}. Consequently, Proposition 2 (with $\epsilon \to \epsilon/2$ and $K_{\max}$ as required by the proposition), the output x of Algorithm 4 satisfies $f_{\max}(x) - f_{\max}(x^\star) \le \epsilon/2$ with probability at least $1 - (1-\frac{2}{3}) - (1 - p_{\text{BROO}}) = p_{\text{BROO}} - \frac{1}{3}$.

To finish the proof of correctness, it remains to verify that $p_{\text{BROO}} \ge 5/6$. To that end, let $K_{\max}^{\text{bisect}} = O(\log\frac{GR^2}{r_\epsilon\epsilon})$ to be the total number of BROO calls in a single execution of λ-BISECTION, as per Proposition 4. Then, if the probability of failure of a single BROO implementation is p_{f} and we perform at most $K_{\max}$ calls to λ-BISECTION, we have $p_{\text{BROO}} \ge 1 - K_{\max}K_{\max}^{\text{bisect}}p_{\text{f}}$. Therefore, taking

$$p_{\text{f}} \le \frac{1}{6K_{\max}K_{\max}^{\text{bisect}}}$$

guarantees correctness.

We now proceed to bound the algorithm's complexity. To that end, we set

$$\lambda_{\min} = \frac{\epsilon}{r_\epsilon^{4/3}R^{2/3}}\log^2\left(\frac{GR}{\epsilon}\right).$$

Recalling that $r_\epsilon = \frac{\epsilon}{2G\log N}$, the total number of iterations in Algorithm 4 is at most

$$K_{\max} = O\left(\left(\frac{R}{r_\epsilon}\right)^{2/3}\log\frac{GR}{\epsilon} + \sqrt{\frac{\lambda_{\min}R^2}{\epsilon}}\right) = O\left(\left(\frac{GR\log N}{\epsilon}\right)^{2/3}\log\frac{GR}{\epsilon}\right). \tag{21}$$

Setting the approximation parameters to be $\varphi_k = O(\frac{\epsilon}{\lambda_k a_k})$, $\delta_k = O(\frac{\epsilon}{R})$ and $\sigma_k^2 = O(\frac{\epsilon}{a_k})$ as required in Proposition 2, the complexity of lines 4 and 5 in the kth iteration of Algorithm 4 is bounded by

Lemma 9 and Lemma 10 as

$$\begin{aligned}
\mathbb{E}\mathcal{N}_{f_{(i)}}^{(k),1} &= O\left(N + \frac{G^2}{\lambda_k \varphi_k} + \frac{G^2}{\sigma_k^2}\log^2\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right) + \log\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right)\right) \\
&= O\left(N + \frac{G^2 a_k}{\epsilon}\log^2\left(\frac{GR}{\epsilon}\right) + \log\left(\frac{GR}{\epsilon}\right)\right) \\
\mathbb{E}\mathcal{N}_{\partial f_{(i)}}^{(k),1} &= O\left(\frac{G^2}{\lambda_k \varphi_k} + \frac{G^2}{\sigma_k^2}\log^2\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right) + \log\left(\frac{G}{\min\{\delta_k, \sigma_k\}}\right)\right) \\
&= O\left(\frac{G^2 a_k}{\epsilon}\log^2\left(\frac{GR}{\epsilon}\right) + \log\left(\frac{GR}{\epsilon}\right)\right).
\end{aligned}$$

To bound the complexity of the bisection procedure at the kth iteration of Algorithm 4, note that it makes a total of $K_{\max}^{\text{bisect}} = O(\log \frac{GR^2}{r_\epsilon \epsilon}) = O(\log \frac{GR \log N}{\epsilon}) = O(\log \frac{GR}{\epsilon})$ BROO calls, $r_\epsilon = \frac{\epsilon}{2G \log N}$ and $\log N \leq \frac{GR}{2\epsilon}$. Applying Lemma 11 with p_{f} and $\lambda_{\min}$ as determined above, the complexity is bounded by

$$\begin{aligned}
\mathcal{N}_{f_{(i)}}^{(k),2} &= O\left(\left(N + \frac{G^2}{\lambda_{\min}^2 r_\epsilon^2}\log\left(\frac{\log(G/(\lambda_{\min} r_\epsilon))}{p_{\mathrm{f}}}\right)\right) K_{\max}^{\text{bisect}}\right) \\
&= O\left(\left(N + \frac{G^2 r_\epsilon^{2/3} R^{4/3}}{\epsilon^2 \log^4\left(\frac{GR}{\epsilon}\right)}\log\left(\frac{GR}{\epsilon}\right)\right)\log\left(\frac{GR}{\epsilon}\right)\right), \\
\mathcal{N}_{\partial f_{(i)}}^{(k),2} &= O\left(\frac{G^2}{\lambda_{\min}^2 r_\epsilon^2}\log\left(\frac{\log(G/(\lambda_{\min} r_\epsilon))}{p_{\mathrm{f}}}\right) K_{\max}^{\text{bisect}}\right) = O\left(\frac{G^2 r_\epsilon^{2/3} R^{4/3}}{\epsilon^2 \log^2\left(\frac{GR}{\epsilon}\right)}\right).
\end{aligned}$$

Summing the bounds above over iterations 1 to $K \leq K_{\max}$ and noting that $\sum_{k \leq K} a_k = A_K \leq 2A_{\max} = O(R^2/\epsilon)$ (see proof of Proposition 2) we obtain the total complexity bounds

$$\begin{aligned}
\mathbb{E}\mathcal{N}_{f_{(i)}} &= \sum_{k \leq K}\left(\mathbb{E}\mathcal{N}_{f_{(i)}}^{(k),1} + \mathbb{E}\mathcal{N}_{f_{(i)}}^{(k),2}\right) \\
&= O\left(K_{\max} N \log\frac{GR}{\epsilon} + \frac{G^2 A_K}{\epsilon}\log^2\frac{GR}{\epsilon} + K_{\max}\frac{G^2 r_\epsilon^{2/3} R^{4/3}}{\epsilon^2 \log^4\left(\frac{GR}{\epsilon}\right)} \cdot \log^2\left(\frac{GR}{\epsilon}\right)\right) \\
&= O\left(\left(\frac{GR \log N}{\epsilon}\right)^{2/3} N \cdot \log^2\frac{GR}{\epsilon} + \frac{G^2 R^2}{\epsilon^2}\log^2\frac{GR}{\epsilon}\right),
\end{aligned}$$

and

$$\mathbb{E}\mathcal{N}_{\partial f_{(i)}} = \sum_{k \leq K}\left(\mathbb{E}\mathcal{N}_{\partial f_{(i)}}^{(k),1} + \mathbb{E}\mathcal{N}_{\partial f_{(i)}}^{(k),2}\right) = O\left(\frac{G^2 R^2}{\epsilon^2}\log^2\frac{GR}{\epsilon}\right),$$

where we have used formula (21) for $K_{\max}$. This concludes the proof. □

E Proofs from Section 5

In the section we prove Theorem 5, the convergence guarantee for Algorithm 5, our gradient-efficient composite optimization method. We first provide a lemma (Lemma 12) that helps us analyze the behavior of the β_k and γ_k sequences in the algorithm. Then we combine it with the approximation guarantees of our estimator to show the convergence rate of Algorithm 5 in Proposition 5. Finally we apply this proposition and bound the expected number of gradient queries complete the proof of Theorem 5.

The following helper lemma is also used in Lan [33, 34]; we provide it here for completeness of analysis.

Lemma 12 (Convergence of geometric sequence, cf. Lemma 2 of Lan [34])**.** *Given $\gamma_k \in (0, 1)$, for all $k \in \mathbb{N}$, and $\Gamma_1 > 0$, define the sequence*

$$\Gamma_k \coloneqq (1 - \gamma_k)\Gamma_{k-1}, \quad \forall k \geq 2.$$

If a sequence E_k satisfies $E_k \leq (1-\gamma_k)E_{k-1} + B_k$, for all $k \geq 1$, then we have for any $k \geq 1$,

$$E_k \leq \Gamma_k \left[\frac{1-\gamma_1}{\Gamma_1} E_0 + \sum_{i \in [k]} \frac{B_i}{\Gamma_i} \right].$$

Using the helper lemma, we can show the following convergence rate for Algorithm 5.

Proposition 5 (Convergence rate)**.** *Given problem* (5) *with optimizer $x^\star$ and initial point $\|x_0 - x^\star\| \leq R$, let $\sigma_k^2 = \frac{R^2}{4N}$, $\delta_k = \frac{R}{16N}$, $\epsilon_k = \frac{LR^2}{2kN}$, and let parameters $\beta_k = \frac{2L}{k}$, $\gamma_k = \frac{2}{k+1}$. Then, the iterates of Algorithm 5 satisfy*

$$\Psi(x_N) - \Psi(x_\star) \leq O\left(\frac{LR^2}{N^2}\right).$$

Proof. We first observe that

$$\begin{aligned}
\Lambda(x_k) &\overset{(i)}{\leq} \Lambda(y_k) + \langle \nabla\Lambda(y_k), x_k - y_k \rangle + \frac{L}{2}\|x_k - y_k\|^2 \\
&\overset{(ii)}{=} (1-\gamma_k)\left[\Lambda(y_k) + \langle \nabla\Lambda(y_k), x_{k-1} - y_k\rangle\right] \\
&\quad + \gamma_k \left[\Lambda(y_k) + \langle \nabla\Lambda(y_k), \bar{v}_k - y_k\rangle\right] + \frac{L\gamma_k^2}{2}\|\bar{v}_k - \mathsf{Proj}_{\mathcal{X}}(v_{k-1})\|^2 \\
&\overset{(iii)}{\leq} (1-\gamma_k)\Lambda(x_{k-1}) + \gamma_k \left[\Lambda(y_k) + \langle \nabla\Lambda(y_k), \bar{v}_k - y_k\rangle + \frac{\beta_k}{2}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - \bar{v}_k\|^2\right] \\
&\quad - \frac{\gamma_k\beta_k - L\gamma_k^2}{2}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - \bar{v}_k\|^2 \\
&\overset{(iv)}{\leq} (1-\gamma_k)\Lambda(x_{k-1}) + \gamma_k \left[\Lambda(y_k) + \langle \nabla\Lambda(y_k), \bar{v}_k - y_k\rangle + \frac{\beta_k}{2}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - \bar{v}_k\|^2\right],
\end{aligned}$$

where we use (i) L smoothness of function Λ, (ii) expanding $x_k = (1-\gamma_k)x_{k-1} + \gamma_k\bar{v}_k$ and replacing $y_k - x_k = \gamma_k(\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - \bar{v}_k)$, (iii) convexity of Λ, and (iv) that $\beta_k \geq L\gamma_k$.

Similarly using convexity of the non-smooth component f and the definition of x_k and $\bar{v}_k$, we obtain

$$f(x_k) \leq (1-\gamma_k)f(x_{k-1}) + \gamma_k f(\bar{v}_k).$$

Thus, summing the two inequalities and recalling the definition $\bar{\Lambda}_k(v) = \Lambda(y_k) + \langle \nabla\Lambda(y_k), v - y_k\rangle$, this is equivalent to

$$\Lambda(x_k) + f(x_k) \leq (1-\gamma_k)\left(\Lambda(x_{k-1}) + f(x_{k-1})\right) + \gamma_k\left[\bar{\Lambda}_k(\bar{v}_k) + f(\bar{v}_k) + \frac{\beta_k}{2}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - \bar{v}_k\|^2\right].$$

Now we recall the definition of composite objectives $\Psi(x) = \Lambda(x) + f(x)$ and define

$$\Phi_k(x) = \bar{\Lambda}_k(x) + f(x) + \frac{\beta_k}{2}\|x - \mathsf{Proj}_{\mathcal{X}}(v_{k-1})\|^2.$$

By convexity of Ψ one has the recursion

$$\Psi(x_k) - \Psi(u) \leq (1-\gamma_k)\left(\Psi(x_{k-1}) - \Psi(u)\right) + \gamma_k\left(\Phi_k(\bar{v}_k) - \Phi_k(u) + \frac{\beta_k}{2}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2\right).$$

Let $v_k^\star$ be the exact minimizer of Φ_k restricted to $\bar{\mathcal{X}} \coloneqq \mathbb{B}_R(v_0) \cap \mathcal{X}$. We have, for any $u \in \bar{\mathcal{X}}$, that $\Phi_k(u) \geq \Phi_k(v_k^\star) + \frac{\beta_k}{2}\|v_k^\star - u\|^2$, and consequently

$$\begin{aligned}
\Psi(x_k) - \Psi(u) &\leq (1-\gamma_k)\left(\Psi(x_{k-1}) - \Psi(u)\right) \\
&\quad + \gamma_k\left(\Phi_k(\bar{v}_k) - \Phi_k(v_k^\star) + \frac{\beta_k}{2}\left(\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2 - \|v_k^\star - u\|^2\right)\right).
\end{aligned}$$

Conditioning on past events and taking expectation over randomness of v_k and $\bar{v}_k$, this gives for any $u \in \bar{\mathcal{X}}$,

$$\begin{aligned}
\mathbb{E}\Psi(x_k) - \Psi(u) &\leq (1-\gamma_k)\left(\Psi(x_{k-1}) - \Psi(u)\right) + \gamma_k\left(\mathbb{E}\Phi_k(\bar{v}_k) - \Phi_k(v_k^\star)\right) \\
&\quad + \frac{\gamma_k\beta_k}{2}\left(\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2 - \mathbb{E}\|v_k - u\|^2 + \mathbb{E}\|v_k - v_k^\star\|^2 + \mathbb{E}2\langle v_k^\star - u, v_k - v_k^\star\rangle\right) \\
&\overset{(i)}{\leq} (1-\gamma_k)\left(\Psi(x_{k-1}) - \Psi(u)\right) + \gamma_k\left(\mathbb{E}\Phi_k(\bar{v}_k) - \Psi(v_k^\star)\right) \\
&\quad + \frac{\gamma_k\beta_k}{2}\left(\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2 - \mathbb{E}\|v_k - u\|^2 + \mathbb{E}\|v_k - v_k^\star\|^2 + 4R\|\mathbb{E}v_k - v_k^\star\|\right) \\
&\overset{(ii)}{\leq} (1-\gamma_k)\left(\Psi(x_{k-1}) - \Psi(u)\right) + \gamma_k\left(\mathbb{E}\Phi_k(\bar{v}_k) - \Phi_k(v_k^\star)\right) \\
&\quad + \frac{\gamma_k\beta_k}{2}\left(\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2 - \mathbb{E}\|\mathsf{Proj}_{\mathcal{X}}(v_k) - u\|^2 + \mathbb{E}\|v_k - v_k^\star\|^2 + 4R\|\mathbb{E}v_k - v_k^\star\|\right)
\end{aligned}$$

where we use (i) the triangle inequality and $v_k^\star \in \bar{\mathcal{X}}$ to conclude $\|v_k^\star - u\| \leq \|v_k^\star - x_0\| + \|x_0 - u\| \leq 2R$, and (ii) the projection property that $\|\mathsf{Proj}_{\mathcal{X}}(v_k) - u\|^2 \leq \|v_k - u\|^2$ for any $u \in \bar{\mathcal{X}}$.

Note that $\mathbb{E}\Phi_k(\bar{v}_k) - \Phi_k(v_k^\star) \leq \epsilon_k$ by the definition of $\bar{v}_k = \widetilde{\mathsf{P}}^{\epsilon_k}_{\bar{\Lambda}_k + f, \beta_k}(v_{k-1})$. Moreover, Theorem 1 guarantees that $\mathbb{E}\|v_k - v_k^\star\| \leq \delta_k$ and that $\mathbb{E}\|v_k - v_k^\star\|^2 \leq \sigma_k^2$. Therefore, writing

$$E_k = \mathbb{E}\Psi(x_k) - \Psi(u)$$

and

$$B_k = \frac{\gamma_k\beta_k}{2}\left(\mathbb{E}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2 - \mathbb{E}\|\mathsf{Proj}_{\mathcal{X}}(v_k) - u\|^2\right) + \gamma_k\beta_k\left(\frac{\epsilon_k}{\beta_k} + \frac{\sigma_k^2}{2} + 2R\delta_k\right),$$

we conclude that $E_k \leq (1-\gamma_k)E_{k-1} + B_k$. Applying Lemma 12, we obtain

$$\begin{aligned}
\mathbb{E}\Psi(x_N) - \Psi(u) &\leq \Gamma_N \frac{1-\gamma_1}{\Gamma_1}\left[\Psi(x_0) - \Psi(u)\right] \\
&\quad + \Gamma_N \sum_{k=1}^{N} \frac{\beta_k\gamma_k}{2\Gamma_k}\left(\mathbb{E}\|\mathsf{Proj}_{\mathcal{X}}(v_{k-1}) - u\|^2 - \mathbb{E}\|\mathsf{Proj}_{\mathcal{X}}(v_k) - u\|^2\right) \\
&\quad + \Gamma_N \sum_{k\in[N]} \frac{\beta_k\gamma_k}{\Gamma_k}\left(\frac{\epsilon_k}{\beta_k} + \frac{\sigma_k^2}{2} + 2R\delta_k\right) \\
&\overset{(i)}{\leq} \Gamma_N L\|v_0 - u\|^2 + \Gamma_N \sum_{k\in[N]} \frac{\beta_k\gamma_k}{\Gamma_k}\left(\frac{\epsilon_k}{\beta_k} + \frac{\sigma_k^2}{2} + 2R\delta_k\right) \overset{(ii)}{\leq} \frac{4LR^2}{N(N+1)},
\end{aligned}$$

where (i) follows from telescoping and $\gamma_1 = 1$, and (ii) is due to $\Gamma_k = \prod_{k\geq 2}(1-\gamma_k) = \frac{2}{k(k+1)}$, so that $\frac{\beta_k\gamma_k}{\Gamma_k} = 2L$, and $\frac{\epsilon_k}{\beta_k} + \frac{\sigma_k^2}{2} + 2R\delta_k \leq \frac{R^2}{2N}$ by the choices $\sigma_k^2 = \frac{R^2}{4N}$, $\delta_k = \frac{R}{16N}$ and $\epsilon_k = \frac{LR^2}{2kN}$. $\square$

We are now ready to prove the main theorem of the section.

Theorem 5. *Given problem* (5) *with solution $x_\star$, a point x_0 such that $\|x_0 - x_\star\| \leq R$ and target accuracy $\epsilon > 0$, Algorithm 5 with $\epsilon_k = LR/2kN$, $\delta_k = R/16N$, $\sigma_k^2 = R^2/4N$, and $N = \Theta(\sqrt{LR^2/\epsilon})$ finds an approximate solution x satisfying $\mathbb{E}\Psi(x) \leq \Psi(x_\star) + \epsilon$ and has complexity $\mathcal{N}_{\nabla\Lambda} = O\left(\sqrt{\frac{LR^2}{\epsilon}}\right)$ and $\mathbb{E}\mathcal{N}_{\hat{\nabla}f} = O\left(\left(\frac{GR}{\epsilon}\right)^2 \log^2 \frac{GR}{\epsilon} + \sqrt{\frac{LR^2}{\epsilon}} \log\left(\frac{GR}{\epsilon}\right)\right)$.*

Proof. By Proposition 5, it suffices to run Algorithm 5 for $N = O(\sqrt{LR^2/\epsilon})$ iterations, which immediately implies the stated bound on $\mathcal{N}_{\nabla\Lambda}$.

Now we consider the cost of attaining the requiring accuracy ϵ_k when computing $\bar{v}_k$. Using the EPOCHSGD and Proposition 3 we can do so with

$$N_k^{(1)} = O\left(\frac{G^2}{\beta_k\epsilon_k}\right) = O\left(\frac{G^2k^2N}{L^2R^2}\right)$$

queries to $\hat{\nabla} f$.

Applying Theorem 1, the expected cost of attaining bias $\delta_k = \frac{R}{16N}$ and variance $\sigma_k^2 = \frac{R^2}{4N}$ is

$$\begin{aligned}\mathbb{E}N_k^{(2)} &= O\left(\log\left(\frac{GNk}{LR}\right) + \frac{NG^2}{\beta_k^2 R^2}\log^2\left(\frac{GNk}{LR}\right)\right) \\ &= O\left(\log\left(\frac{GNk}{LR}\right) + \frac{G^2k^2N}{L^2R^2}\log^2\left(\frac{GNk}{LR}\right)\right)\end{aligned}$$

queries to $\hat{\nabla} f$.

Summing these over all $k \le N = O(\sqrt{LR^2/\epsilon})$, we obtain the the required complexity bound

$$\begin{aligned}\mathbb{E}\mathcal{N}_{\hat{\nabla} f} &= \sum_k \left(N_k^{(1)} + \mathbb{E}N_k^{(2)} + 1\right) = O\left(N\log\left(\frac{GN^2}{LR}\right) + \frac{G^2N^4}{L^2R^2}\log^2\left(\frac{GN^2}{LR}\right)\right) \\ &= O\left(\sqrt{\frac{LR^2}{\epsilon}}\log\left(\frac{GR}{\epsilon}\right) + \frac{G^2R^2}{\epsilon^2}\log^2\left(\frac{GR}{\epsilon}\right)\right).\end{aligned}$$

□

F Proofs and additional remarks from Section 6

In this section we prove Theorem 6 which gives an optimal complexity and generalization bound for differentially private stochastic convex optimization, conditional on the existence of an improved optimum estimator (Definition 3). We begin by stating a standard privacy guarantee for the Gaussian mechanism applied on mappings with bounded ℓ_2 sensitivity, and a lemma that helps us bound the sensitivity of the conjunctured bounded estimator. With these results in hand, we prove Theorem 6. Finally, we discuss some challenges and prospects for constructing bounded estimators that satisfy Definition 3.

F.1 Helper lemmas

Privacy of the Gaussian mechanism. In this section, we present the privacy guarantees of the Gaussian mechanism which will be useful for the proof of Theorem 6. First, for an estimator (or a function) $h : \mathbb{S}^n \to \mathbb{R}^d$, the ℓ_2-sensitivity of the estimator is upper bounded by Δ if $\sup_{\mathcal{S},\mathcal{S}' \in \mathbb{S}^n : d_{\mathsf{ham}}(\mathcal{S},\mathcal{S}') \le 1} \|h(\mathcal{S}) - h(\mathcal{S}')\| \le \Delta$, where d_{ham} is the hamming distance between the two samples (i.e., $\mathcal{S}, \mathcal{S}'$ with hamming distance $d_{\mathsf{ham}}(\mathcal{S}, \mathcal{S}') \le 1$ have at most a single different element). We can now state the privacy guarantees of the Gaussian mechanism.

Lemma 13 (Gaussian mechanism [21, Theorem A.1])**.** *Let $h : \mathbb{S}^n \to \mathbb{R}^d$ have ℓ_2-sensitivity Δ. Then the Gaussian mechanism $\mathcal{A}(\mathcal{S}) = h(\mathcal{S}) + \mathsf{N}(0, \sigma^2 I_d)$ with $\sigma = 2\Delta\log(2/\beta)/\alpha$ is (α, β)-DP.*

Bounding the number of estimator copies that use a particular sample. To prove Theorem 6, we begin with a lemma which bounds the number of optimum estimator copies that each sample can participate in. To this end, let $S_{i,t}$ denote the set of samples used in iteration i of Algorithm 6 during the computation of the t'th optimum estimator copy. For a sample s_ℓ, we let $K_{i,\ell}$ denote the number of sets $S_{i,t}$ such that $z_\ell \in S_{i,t}$. Recalling that the number of iterations $k = \lceil \log n \rceil$ and that $\bar{n} = n/k$, we have the following lemma.

Lemma 14. *Let $\mu_i = \frac{1}{\eta_i \bar{n}}$. Assume we use an optimum oracle $\mathcal{O}$ satisfying Definition 3 with constant C_2 and $\delta_i^2 = \frac{G^2}{\mu_i^2 \bar{n}}$. Then, for any $\beta \le 1/n$,*

$$\mathbb{P}\left(\max_{1\le i\le k, 1\le \ell\le n} K_{i,\ell} \ge 20\log(1/\beta) + 6C_2\log^2 n\right) \le \beta/2.$$

Proof. We first prove the claim for a fixed i and ℓ and then we apply a union bound. Fix $1 \le i \le k$ and $1 \le \ell \le n$ and define $Y_t = \mathbb{1}_{\{z_\ell \in S_{i,t}\}}$. Now we upper bound $p = \mathbb{P}(Y_t = 1)$. Let the random variable N_t denote the number of subgradients the t'th query to $\mathcal{O}_\delta$ at iteration i uses. First, note that whenever $N_t = j$, we have

$$\mathbb{P}(Y_t = 1 \mid N_t = j) \le j/\bar{n},$$

by the union bound. Thus, Definition 3 now implies

$$\begin{aligned}\mathbb{P}(Y_t = 1) &= \sum_{k=1}^{\infty} \mathbb{P}(Y_t = 1 \mid N_t = j)\mathbb{P}(N_t = j) \\ &\le \frac{1}{\bar{n}} \sum_{k=1}^{\infty} \mathbb{P}(N_t = j)j = \frac{\mathbb{E}[N]}{\bar{n}} = \frac{C_2}{\bar{n}} \log \frac{G}{\mu_i \delta_i}.\end{aligned}$$

We can now use a Chernoff bound to prove the claim. Indeed, as $K_{i,\ell} = \sum_{t=1}^{n} Y_t$ and $Y_t \sim \mathsf{Bernoulli}(p)$ are i.i.d., Lemma 15 below implies that for $c \ge 6$,

$$\mathbb{P}(K_{i,\ell} \ge c\mathbb{E}[K_{i,\ell}]) = \mathbb{P}\left(\sum_{t=1}^{n} Y_t \ge cnp\right) \le 2^{-cnp}.$$

As $p \le C_2 \log(n) \log(G/\mu_i\delta_i)/n$, we take $c \ge 6$ such that $cnp \ge 20\log(1/\beta)$, hence we have

$$\mathbb{P}(K_{i,\ell} \ge 20\log(1/\beta) + 6C_2 \log(n)\log(G/\mu_i\delta_i))) \le \beta^4.$$

Applying a union bound over all n samples and all $k = \lceil \log n \rceil$ iterations, we have that

$$\mathbb{P}\left(\max_{1\le i\le k, 1\le \ell\le n} K_{i,\ell} \ge 20\log(1/\beta) + 6C_2\log(n)\log(G/\mu_i\delta_i))\right) \le \beta/2.$$

The claim now follows by noting that $\frac{G}{\mu_i\delta_i} \le \sqrt{n}$ using our choice of δ_i in Algorithm 6.

□

Lemma 15 ([37], Ch. 4.2.1)**.** *Let $X = \sum_{i=1}^{n} X_i$ for $X_i \overset{\text{iid}}{\sim} \mathsf{Bernoulli}(p)$. Then for $c \ge 6$,*

$$P(X \ge cnp) \le 2^{-cnp}.$$

F.2 Proof of Theorem 6

Theorem 6 (conditional)**.** *Given an efficient bounded low-bias estimator $\mathcal{O}_\delta$ satisfying Definition 3 for any $\delta > 0$, then for $\alpha \le \log(1/\beta)$, $\mathcal{X} \in \mathbb{B}_R(x_0)$, convex and G-Lipschitz $\hat{f}(x; s)$, Algorithm 6 is (α, β)-DP, queries $\widetilde{O}(n)$ subgradients and has (hiding logarithmic factors in n) $\mathbb{E}[f(x_k) - \min_{x\in\mathcal{X}} f(x)] \le GR \cdot \widetilde{O}\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log^3(1/\beta)}}{n\alpha}\right)$.*

Proof. We begin by proving the privacy claim. We show that each iterate is (α, β)-DP which completes the proof by post-processing as each sample is used in exactly one iterate. To this end, first we show that, with high probability, each sample z_ℓ is used in at most $B = 20(\log(\frac{1}{\beta}) + C_2 \log^2 n)$ different optimum-estimator queries; we let $\mathfrak{E}$ denote this event. More precisely, let $S_{i,t}$ denote the set of samples used in iteration i during the application of the t'th oracle. Then for every i and sample z_ℓ, letting $K_{i,\ell}$ be the number of sets $S_{i,t}$ such that $z_\ell \in S_{i,t}$. Using this notation, the event $\mathfrak{E}$ is equivalent to $\max_{1\le i\le k, 1\le \ell\le n} K_{i,\ell} \le B$. Lemma 14 implies that $P[\mathfrak{E}] \ge 1 - \beta/2$, therefore we only have to prove $(\alpha, \beta^2/2)$-differential privacy assuming event $\mathfrak{E}$ happens as we have using $e^\alpha \le 1/\beta$ that

$$\begin{aligned}P[\mathcal{A}(\mathcal{S}) \in \mathcal{O}] &\le P[\mathcal{A}(\mathcal{S}') \in \mathcal{O} \mid \mathfrak{E}]P[\mathfrak{E}] + (1 - P[\mathfrak{E}]) \\ &\le e^\alpha P[\mathcal{A}(\mathcal{S}') \in \mathcal{O} \mid \mathfrak{E}]P[\mathfrak{E}] + \beta/2 \\ &\le e^\alpha P[\mathcal{A}(\mathcal{S}') \in \mathcal{O}] + \beta.\end{aligned}$$

We therefore assume $\mathfrak{E}$ holds and proceed to bound the ℓ_2-sensitivity of $\tilde{x}_i$. To this end, let $\mu_i = 1/(\eta_i \bar{n})$ and $\hat{x}_i = \operatorname{argmin}_{x\in\mathcal{X}} F_i(x)$. First, note that each optimum estimation oracle output satisfies

$$\begin{aligned}\|\mathcal{O}_{\delta_i}(F_i) - x_{i-1}\| &\le \|\mathcal{O}_{\delta_i}(F_i) - \hat{x}_i\| + \|\hat{x}_i - x_{i-1}\| \\ &\overset{(\star)}{\le} \sqrt{C_1} G\sqrt{\log n}/\mu_i + G/\mu_i \\ &= (\sqrt{C_1 \log n} + 1)G/\mu_i,\end{aligned}$$

where the first term in inequality $(\star)$ above holds since the estimator $\mathcal{O}_{\delta_i}$ satisfies Definition 3 and $F_i = f_i + \psi_i$ where f_i is G-Lipschitz and ψ_i is μ_i-strongly convex with $G/(\mu_i\delta_i) \le \sqrt{n}$. The second term of the inequality holds since $\psi_i(x) = \mu_i\|x - x_{i-1}\|^2$, thus as f_i is G-Lipschitz we have

$$\mu_i\|\hat{x}_i - x_{i-1}\|^2 \le f_i(x_{i-1}) - f_i(\hat{x}_i) \le G\|\hat{x}_i - x_{i-1}\|.$$

As event $\mathfrak{E}$ holds, each sample participates in at most B of the optimum estimator computations queries, hence we have that the ℓ_2-sensitivity of $\tilde{x}_i$ is at most $2\frac{B}{\bar{n}}(\sqrt{C_1\log n} + 2)G/\mu_i$. Privacy properties of the Gaussian mechanism (Lemma 13) and our choice of σ_i now imply that each iterate is $(\alpha, \beta^2/2)$-DP whenever event $\mathfrak{E}$ holds, which proves the claim about privacy.

Let us now prove utility following steps similar to the proof of Theorem 4.4 in [23]. We define the non-private minimizers, $\hat{x}_i = \operatorname{argmin}_{x\in\mathcal{X}} F_i(x)$ and $\hat{x}_0 = x_\star$. We have

$$f(x_k) - f(x_\star) = \sum_{i=1}^{k}[f(\hat{x}_i) - f(\hat{x}_{i-1})] + f(x_k) - f(\hat{x}_k). \tag{22}$$

Using the definitions of σ_i and η_i in Algorithm 6, we also have that for every $i \ge 1$

$$\begin{aligned}
\mathbb{E}[\|\hat{x}_i - x_i\|^2] &\le 2\mathbb{E}[\|\hat{x}_i - \tilde{x}_i\|^2] + 2\mathbb{E}[\|\tilde{x}_i - x_i\|^2] \\
&\le 2\mathbb{E}[\|\hat{x}_i - \tilde{x}_i\|^2] + O\left(\frac{G^2B^2\eta_i^2 d\log(n)\log(1/\beta)}{\alpha^2}\right) \\
&\le 2\mathbb{E}[\|\hat{x}_i - \tilde{x}_i\|^2] + O\left(\frac{G^2B^2\eta^2 d\log(n)\log(1/\beta)}{\alpha^2 2^{8i}}\right).
\end{aligned}$$

Moreover, using properties of the bounded-optimum estimator from Definition 3, that is, $\|\mathcal{O}_{\delta_i}(F_i, x_{i-1}) - \hat{x}_i\|^2 \le C_1G^2\log(n)/\mu_i^2$ and $\|\mathbb{E}[\mathcal{O}_{\delta_i}(F_i, x_{i-1}) - \hat{x}_i]\|^2 \le \delta_i^2$, we have by choosing $\delta_i^2 = G^2/\mu_i^2\bar{n} = G^2\eta_i^2\bar{n}$,

$$\begin{aligned}
\mathbb{E}\|\tilde{x}_i - \hat{x}_i\|^2 &= \mathbb{E}\left\|\frac{1}{\bar{n}}\sum_{j=1}^{\bar{n}}\mathcal{O}_{\delta_i}(F_i, x_{i-1}) - \hat{x}_i\right\|^2 \\
&\le \frac{C_1G^2\log(n)}{\mu_i^2\bar{n}} + \rho^2 \le (C_1+1)G^2\eta_i^2\bar{n}\log(n).
\end{aligned}$$

We can now bound the terms in (22). For the second term, the choice of η gives

$$\begin{aligned}
\mathbb{E}[f(x_k) - f(\hat{x}_k)] &\le G\mathbb{E}[\|x_k - \hat{x}_k\|] \\
&\le G\cdot O\left(G\eta_k\sqrt{\bar{n}\log(n)} + \frac{RB}{2^{6k}}\right) \\
&\le G\cdot O\left(\frac{2G\eta\sqrt{\bar{n}\log(n)}}{2^{4k}} + \frac{RB}{2^{6k}}\right) \\
&\le O\left(\frac{RG}{n}\right).
\end{aligned}$$

For the first term in (22), as F_i is G-Lipschitz over $\mathcal{X}_i = \{x \in \mathcal{X} : \|x - x_{i-1}\| \le 2G\eta_i\bar{n}\}$, Theorems 6 and 7 in [46] imply that for all $y \in \mathcal{X}_i$

$$\mathbb{E}[f(\hat{x}_i) - f(y)] \le \frac{\mathbb{E}[\|y - x_{i-1}\|^2]}{\eta_i\bar{n}} + 2G^2\eta_i,$$

hence we now have

$$\begin{aligned}
\sum_{i=1}^{k} \mathbb{E}[f(\hat{x}_i) - f(\hat{x}_{i-1})] &\leq \sum_{i=1}^{k} \mu_{i-1}\mathbb{E}[\|\hat{x}_{i-1} - x_{i-1}\|^2] + 2G^2\eta_i \\
&\leq O\left(\frac{R^2}{\eta\bar{n}} + \sum_{i=2}^{k} \mu_i \left(\frac{G^2\log(n)}{\mu_i^2\bar{n}} + \frac{G^2B^2\eta_i^2 d\log(n)\log(1/\beta)}{\alpha_i^2}\right) + G^2\eta_i\right) \\
&\leq O\left(\frac{R^2}{\eta\bar{n}} + \sum_{i=2}^{k} G^2\eta_i\log(n) + \frac{G^2B^2\eta_i d\log(n)\log(1/\beta)}{\alpha_i^2\bar{n}}\right) \\
&\leq O\left(\frac{R^2}{\eta\bar{n}} + G^2\eta\log(n) + \sum_{i=2}^{k} 2^{-i}\frac{G^2B^2\eta d\log(n)\log(1/\beta)}{\alpha^2\bar{n}}\right) \\
&\leq GR\cdot O\left(\frac{\log n}{\sqrt{n}} + \frac{B\log(n)\sqrt{d\log(1/\beta)}}{n\alpha}\right),
\end{aligned}$$

where the last inequality follows since $\bar{n} = n/\lceil\log(n)\rceil$, and $\eta = \frac{R}{G}\min(1/\sqrt{n}, \alpha/B\log(n)\sqrt{d\log(1/\beta)})$. □

F.3 The challenges of obtaining a bounded optimum estimator

To highlight the challenge of finding bounded estimators that satisfy Definition 3, let us explain why our MLMC optimum estimator (1) fails to do so. For this estimator, we have (when $2^J \leq T_{\max}$)

$$\|\hat{x}_\star - x_\star\| \leq \|x_\star - x_0\| + 2^J\|x_J - x_{J-1}\|,$$

where x_j is the output of an ODC algorithm with query budget 2^j. The ODC property and the triangle inequality then roughly imply that $\|x_j - x_{j-1}\| = O(2^{-j/2}G/\mu)$ and consequently (since $\|x_\star - x_0\| = O(G/\mu)$) we have $\|\hat{x}_\star - x_\star\| = O(2^{J/2}G/\mu) = O(\sqrt{T_{\max}}G/\mu)$ which clearly is not enough to guarantee an $\widetilde{O}(G/\mu)$ bound on $\|\hat{x}_\star - x_\star\|$. Indeed, to guarantee such bound with a similar analysis we would have needed $\|x_j - x_{j-1}\| = O(2^{-j}G/\mu)$. However, this would imply that, by the triangle inequality,

$$\|x_j - x_\star\| = \|x_j - x_\infty\| \leq \sum_{k=j+1}^{\infty} \|x_k - x_{k-1}\| = \sum_{k=j+1} O(2^{-k}G/\mu) = O(2^{-j}G/\mu),$$

which contradicts the lower bound on the optimal distance convergence rate in Appendix A.4.

Having explained why the analysis strategy underlying our estimator (1) cannot directly yield a bounded optimum estimator, we discuss two approaches with a potential to solve the problem. The first approach is to apply ODC algorithms on a smooth surrogate of the true objective F, for which the faster convergence to the optimum is possible, e.g., using randomized smoothing [20, 32].

The second approach is try to directly bound the ℓ_2 sensitivity of our MLMC-based approach. In particular, it might be possible to leverage the structure of our estimator (or an improved version thereof) in order to control the ℓ_2 sensitivity without relying on the boundedness of the estimator as we currently do in the proof of Theorem 6.