A gradient sampling method with complexity guarantees for Lipschitz functions in high and low dimensions

Anonymous Author(s)
Affiliation
Address
email

Abstract

Zhang et al. [ZLJ+20] introduced a novel modification of Goldstein's classical subgradient method, with an efficiency guarantee of $O(\varepsilon^{-4})$ for minimizing Lipschitz functions. Their work, however, makes use of an oracle that is not efficiently implementable. In this paper, we obtain the same efficiency guarantee with a standard subgradient oracle, thus making our algorithm efficiently implementable. Our resulting method works on any Lipschitz function whose value and gradient can be evaluated at points of differentiability. We additionally present a new cutting plane algorithm that achieves a complexity of $O(d\varepsilon^{-2}\log S)$ for the class of S-smooth (and possibly non-convex) functions in low dimensions. Strikingly, this ϵ-dependence matches the lower bounds for the convex setting.

1 Introduction

The subgradient method [SKR85] is a classical procedure for minimizing a nonsmooth Lipschitz function f on $\mathbb{R}^d$. Starting from an initial iterate x_0, the method computes

$$x_{t+1} = x_t - \alpha_t v_t \text{ where } v_t \in \partial f(x_t). \tag{1}$$

Here, the positive sequence $\{\alpha_t\}_{t\geq 0}$ is user-specified, and the set ∂f is the *Clarke subdifferential*,

$$\partial f(x) = \operatorname{conv}\left\{\lim_{i\to\infty} \nabla f(x_i) : x_i \to x,\ x_i \in \operatorname{dom}(\nabla f)\right\}.$$

In classical circumstances, the subdifferential reduces to familiar objects: for example, when f is C^1-smooth at x, the subdifferential $\partial f(x)$ consists of only the gradient $\nabla f(x)$, while for convex functions, it reduces to the subdifferential in the sense of convex analysis.

For problems that are weakly convex, the limit points $\bar{x}$ of the subgradient method are known to be first-order critical, meaning $0 \in \partial f(\bar{x})$. Recall that a function f is called ρ-weakly convex if the quadratically perturbed function $x \mapsto f(x) + \frac{\rho}{2}\|x\|^2$ is convex. The class of weakly convex functions includes *all* smooth (possibly non-convex) functions. In particular, any function that is S-smooth is also S-weakly convex. Going beyond asymptotic guarantees, finite-time complexity estimates are known for smooth, convex, or weakly convex problems [GL13, RHS+16, JGN+17, AZ18, CDHS18, DDMP18, FLLZ18, ZXG18].

Modern machine learning, however, has witnessed the emergence of problems far beyond the weakly convex problem class. Indeed, tremendous empirical success has been recently powered by industry-backed solvers, such as Google's TensorFlow and Facebook's PyTorch, which routinely train nonsmooth nonconvex deep networks via (stochastic) subgradient methods. Despite a vast body of work on the asymptotic convergence of subgradient methods for nonsmooth nonconvex problems

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

[BHS05, Kiw07, MMM18, DDKL20, BP21], no finite-time nonasymptotic convergence rates were known outside the weakly convex setting until recently, with [ZLJ+20] making a big leap forward towards this goal.

In particular, restricting themselves to the class of Lipschitz and directionally differentiable functions, [ZLJ+20] developed an efficient algorithm motivated by Goldstein's conceptual subgradient method [Gol77]. Moreover, this was recently complemented by [KS21] with lower bounds for finding *near*-approximate-stationary points for nonconvex nonsmooth functions.

A major limitation of [ZLJ+20] is that their complexity guarantees and algorithm use a nonstandard first-order oracle whose validity is unclear in examples. To elaborate, their algorithm requires the following oracle access: given $x, u \in \mathbb{R}^d$ solve the *auxiliary convex feasibility problem*:

$$\text{find } g \in \partial f(x) \text{ subject to } \langle g, u \rangle = f'(x, u), \tag{2}$$

where $f'(x, u)$ is the directional derivative of f at x in the direction of u. The first issue with this oracle is that no general recipe exists for representing the full subdifferential $\partial f(x)$ analytically. Moreover, $\partial f(x)$ could be a very complicated set, e.g., for a deep ReLU neural network, the subdifferential is a polyhedron with a potentially huge number of facets, making the complexity of (2) unclear.

Further, [ZLJ+20] claim that for a composition of directionally differentiable functions with a closed-form directional derivative for each function, we can find the desired g by the chain rule. While the chain rule does compute the directional derivative $f'(x, u)$, to the best of our knowledge, this does not translate to solving (2). We were therefore unable to verify this claim.

Finally, we are unaware of other optimization algorithms imposing this oracle model. Therefore, at face value, the convergence guarantees of [ZLJ+20] are not comparable to those of others.

1.1 Our results

Weakly convex optimization via a standard oracle. Our first contribution is to *replace the non-standard assumption in* (2) *with a standard first-order oracle model*. We show (Section 2) that a variant of the algorithm of [ZLJ+20]works for any Lipschitz function assuming only an oracle that can compute gradients and function values at almost every point of $\mathbb{R}^d$ in the sense of Lebesgue measure. In particular, such oracles arise from automatic differentiation schemes routinely used in deep learning [BP20, BP21].

Our end result is a randomized algorithm for minimizing any L-Lipschitz function that outputs a (δ, ϵ)-stationary point (Definition 1) after using at most $\widetilde{\mathcal{O}}\left(\frac{\Delta L^2}{\epsilon^3 \delta} \log(1/\gamma)\right)$ [1] gradient and function evaluations. Here Δ is the initial function gap and γ is the failure probability.

In light of the above modifications, our algorithm is implementable in the many important settings like deep neural networks that [ZLJ+20] is not. Along the way, we also simplify their proof techniques by providing a geometric viewpoint of the algorithm.

Improved complexity in low dimensions. Having obtained the result of [ZLJ+20] within the standard first-order oracle model, we then proceed to investigate the following question.

Can we improve the efficiency of the algorithm in ***low dimensions****?*

In addition to being natural from the viewpoint of complexity theory, this question is well-grounded in applications. For instance, numerous problems in control theory involve minimization of highly irregular functions of a small number of variables. We refer the reader to the survey [BCL+20, Section 6] for an extensive list of examples, including Chebyshev approximation by exponential sums, spectral and pseudospectral abscissa minimization, maximization of the "distance to instability", and fixed-order controller design by static output feedback. We note that for many of these problems, the gradient sampling method of [BCL+20] is often used. Despite its ubiquity in applications, the gradient sampling method does not have finite-time efficiency guarantees. The algorithms we present here offer an alternative approach with a complete complexity theory.

[1] Throughout the paper, we use $\widetilde{\mathcal{O}}(\,\cdot\,)$ to hide poly-logarithmic factors in L, δ, Δ, and ϵ.

The second contribution of our paper is *an affirmative answer to the highlighted question*. We present a novel algorithm that uses $\widetilde{\mathcal{O}}\left(\frac{\Delta L d}{\epsilon^2 \delta}\log(1/\gamma)\right)$ calls to our (weaker) oracle. Thus we are able to trade off the factor $L\epsilon^{-1}$ with d. Further, if the function is ρ-weakly convex, the complexity improves to $\widetilde{\mathcal{O}}\left(\frac{\Delta d}{\epsilon\delta}\log(\rho)\right)$, which matches the complexity in $\delta = \epsilon$ of gradient descent for smooth minimization. As a direct corollary, for any S-smooth function, we obtain an efficiency of $\widetilde{\mathcal{O}}\left(\frac{\Delta d}{\epsilon\delta}\log(S)\right)$. Strikingly, the dependence on the weak convexity constant ρ (or the smoothness constant S) is only logarithmic.

To put this contribution in perspective, assume for now $\delta = \epsilon$: then, our algorithm's dependence on ϵ in the case of Lipschitz, weakly convex functions is likely optimal in low dimensions, following a conjecture by Bubeck and Mikulincer [BM20] on the optimality of gradient descent for *smooth optimization* in dimension $d = \log(\frac{1}{\epsilon})$ (thus matching the lower bound by Carmon, Duchi, Hinder, and Sidford [CDHS20]). Aside from possible optimality, the logarithmic dependence on smoothness/weak convexity exhibited by our iteration complexity is a significant improvement over the prior result of either $O(1/\epsilon^4)$ by [ZLJ+20] or Nemirovski and Yudin's rate of $O(1/\epsilon^2)$ with a polynomial dependence on smoothness. In the process, we also show that the minimal-norm element of the Goldstein-subdifferential in low dimensions can be found in time $O(\log(1/\epsilon))$, thus settling a question open since the 70s [Gol77].

Techniques. The main idea underlying our improved dependence on ϵ in low dimensions is outlined next. The algorithm of [ZLJ+20] comprises of an outer loop with $\mathcal{O}\left(\frac{\Delta}{\epsilon\delta}\right)$ iterations, each performing either a decrease in the function value or an ingenious random sampling step to update the descent direction. Our observation, central to improving the ε dependence, is that the violation of the descent condition can be transformed into a gradient oracle for the problem of finding a minimal norm element of the Goldstein subdifferential. This gradient oracle may then be used within a cutting plane method, which achieves better ε dependence at the price of a dimension factor (Section 3).

Limitations. One limitation of our work is that our second contribution does not immediately extend to the stochastic setting. We consider this to be an interesting open problem to resolve.

Notation. Throughout, we let $\mathbb{R}^d$ denote a d-dimensional Euclidean space equipped with a dot product $\langle\cdot,\cdot\rangle$ and the Euclidean norm $\|x\|_2 = \sqrt{\langle x, x\rangle}$. The symbol $\mathbb{B}_r(x)$ denotes an open Euclidean ball of radius $r > 0$ around a point x. Throughout, we fix a function $f\colon \mathbb{R}^d \to \mathbb{R}$ that is L-Lipschitz, and let $\mathrm{dom}(\nabla f)$ denote the set of points where f is differentiable—a full Lebesgue measure set by Rademacher's theorem. The symbol $f'(x,u) \stackrel{\text{def}}{=} \lim_{\tau\downarrow 0}\tau^{-1}(f(x+\tau u) - f(x))$ denotes the directional derivative of f at x in direction u, whenever the limit exists.

2 Interpolated normalized gradient descent

In this section, we describe the results in [ZLJ+20] and our modified subgradient method that achieves finite-time guarantees in obtaining (δ,ϵ)-stationarity for an L-Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$. The main construction we use is the Goldstein subdifferential [Gol77].

Definition 1 (Goldstein subdifferential)**.** *Consider a locally Lipschitz function $f\colon \mathbb{R}^d \to \mathbb{R}$, a point $x \in \mathbb{R}^d$, and a parameter $\delta > 0$. The* Goldstein subdifferential *of f at x is the set*

$$\partial_\delta f(x) \stackrel{\text{def}}{=} \mathrm{conv}\Big(\bigcup_{y\in\mathbb{B}_\delta(x)} \partial f(y)\Big).$$

A point x is called (δ,ϵ)-stationary if dist$(0, \partial_\delta f(x)) \le \epsilon$.

Thus, the Goldstein subdifferential of f at x is the convex hull of all Clarke subgradients at points in a δ-ball around x. Famously, [ZLJ+20] showed that one can significantly decrease the value of f by taking a step in the direction of the minimal norm element of $\partial_\delta f(x)$. Throughout the rest of the section, we fix $\delta \in (0,1)$ and use the notation

$$\hat{g} \stackrel{\text{def}}{=} g/\|g\|_2 \text{ for any nonzero vector } g \in \mathbb{R}^d.$$

Theorem 1 ([Gol77])**.** *Fix a point x, and let g be a minimal norm element of $\partial_\delta f(x)$. Then as long as $g \neq 0$, we have $f(x - \delta\hat{g}) \le f(x) - \delta\|g\|_2$.*

Theorem 1 immediately motivates the following conceptual descent algorithm:

$$x_{t+1} = x_t - \delta \hat{g}_t, \text{ where } g_t \in \operatorname*{argmin}_{g \in \partial_\delta f(x)} \|g\|_2. \tag{3}$$

In particular, Theorem 1 guarantees that, defining $\Delta \stackrel{\text{def}}{=} f(x_0) - \min f$, the *approximate stationarity condition*

$$\min_{t=1,\ldots,T} \|g_t\|_2 \leq \epsilon \text{ holds after } T = \mathcal{O}\left(\frac{\Delta}{\delta\epsilon}\right) \text{ iterations of (3).}$$

Evaluating the minimal norm element of $\partial_\delta f(x)$ is impossible in general, and therefore the descent method described in (3) cannot be applied directly. Nonetheless it serves as a guiding principle for implementable algorithms. Notably, the gradient sampling algorithm [BLO05] in each iteration forms polyhedral approximations K_t of $\partial_\delta f(x_t)$ by sampling gradients in the ball $\mathbb{B}_\delta(x)$ and computes search directions $g_t \in \operatorname{argmin}_{g \in K_t} \|g\|_2$. These gradient sampling algorithms, however, have only asymptotic convergence guarantees [BCL+20].

The recent paper [ZLJ+20] remarkably shows that for any $x \in \mathbb{R}^d$ one *can* find an *approximate* minimal norm element of $\partial_\delta f(x)$ using a number of subgradient computations that is independent of the dimension. The idea of their procedure is as follows. Suppose that we have a trial vector $g \in \partial_\delta f(x)$ (not necessarily a minimal norm element) satisfying

$$f(x - \delta\hat{g}) \geq f(x) - \frac{\delta}{2}\|g\|_2. \tag{4}$$

That is, the decrease in function value is not as large as guaranteed by Theorem 1 for the true minimal norm subgradient. One would like to now find a vector $u \in \partial_\delta f(x)$ so that the norm of some convex combination $(1-\lambda)g + \lambda u$ is smaller than that of g. A short computation shows that this is sure to be the case for all small $\lambda > 0$ as long as $\langle u, g\rangle \leq \|g\|_2^2$. The task therefore reduces to:

$$\text{find some } u \in \partial_\delta f(x) \quad \text{satisfying} \quad \langle u, g\rangle \leq \|g\|_2^2.$$

The ingenious idea of [ZLJ+20] is a randomized procedure for establishing exactly that in expectation. Namely, suppose for the moment that f happens to be differentiable along the segment $[x, x - \delta\hat{g}]$; we will revisit this assumption shortly. Then the fundamental theorem of calculus, in conjunction with (4), yields

$$\frac{1}{2}\|g\|_2 \geq \frac{f(x) - f(x - \delta\hat{g})}{\delta} = \frac{1}{\delta}\int_0^\delta \langle \nabla f(x - \tau\hat{g}), \hat{g}\rangle \, d\tau. \tag{5}$$

Consequently, a point y chosen uniformly at random in the segment $[x, x - \delta\hat{g}]$ satisfies

$$\mathbb{E}\langle \nabla f(y), g\rangle \leq \frac{1}{2}\|g\|_2^2. \tag{6}$$

Therefore the vector $u = \nabla f(y)$ can act as the subgradient we seek. Indeed, the following lemma shows that, in expectation, the minimal norm element of $[g, u]$ is significantly shorter than g. The proof is extracted from that of [ZLJ+20, Theorem 8].

Lemma 2 ([ZLJ+20])**.** *Fix a vector $g \in \mathbb{R}^d$, and let $u \in \mathbb{R}^d$ be a random vector satisfying $\mathbb{E}\langle u, g\rangle < \frac{1}{2}\|g\|_2^2$. Suppose moreover that the inequality $\|g\|_2, \|u\|_2 \leq L$ holds for some $L < \infty$. Then the minimal-norm vector z in the segment $[g, u]$ satisfies:*

$$\mathbb{E}\|z\|_2^2 \leq \|g\|_2^2 - \frac{\|g\|_2^4}{16L^2}.$$

Proof. Applying $\mathbb{E}\langle u, g\rangle \leq \frac{1}{2}\|g\|_2^2$ and $\|g\|_2, \|u\|_2 \leq L$, we have, for any $\lambda \in (0,1)$,

$$\begin{aligned}\mathbb{E}\|z\|_2^2 \leq \mathbb{E}\|g + \lambda(u-g)\|_2^2 &= \|g\|_2^2 + 2\lambda\mathbb{E}\langle g, u-g\rangle + \lambda^2\mathbb{E}\|u-g\|_2^2 \\ &\leq \|g\|_2^2 - \lambda\|g\|_2^2 + 4\lambda^2 L^2.\end{aligned}$$

Plugging in the value $\lambda = \frac{\|g\|_2^2}{8L^2} \in (0,1)$ minimizes the right hand side and completes the proof. □

The last technical difficulty to overcome is the requirement that f be differentiable along the line segment $[g, u]$. This assumption is crucially used to obtain (5) and (6). To cope with this problem, [ZLJ+20] introduce extra assumptions on the function f to be minimized and assume a nonstandard oracle access to subgradients.

We show, using Lemma 3, that no extra assumptions are needed if one slightly perturbs g.

Lemma 3. *Let $f\colon \mathbb{R}^d \to \mathbb{R}$ be a Lipschitz function, and fix a point $x \in \mathbb{R}^d$. Then there exists a set $\mathcal{D} \subset \mathbb{R}^d$ of full Lebesgue measure such that for every $y \in \mathcal{D}$, the line spanned by x and y intersects $\mathrm{dom}(\nabla f)$ in a full Lebesgue measure set in $\mathbb{R}$. Then, for every $y \in \mathcal{D}$ and all $\tau \in \mathbb{R}$, we have*

$$f(x + \tau(y - x)) - f(x) = \int_0^\tau \langle \nabla f(x + s(y - x)), y - x \rangle \, ds.$$

Proof. Without loss of generality, we may assume $x = 0$ and $f(x) = 0$. Rademacher's theorem guarantees that $\mathrm{dom}(\nabla f)$ has full Lebesgue measure in $\mathbb{R}^d$. Fubini's theorem then directly implies that there exists a set $\mathcal{Q} \subset \mathbb{S}^{d-1}$ of full Lebesgue measure within the sphere $\mathbb{S}^{d-1}$ such that for every $y \in \mathcal{Q}$, the intersection $\mathbb{R}_+\{y\} \cap (\mathrm{dom}(\nabla f))^c$ is Lebesgue null in $\mathbb{R}$. It follows immediately that the set $\mathcal{D} = \{\tau y : \tau > 0, y \in Q\}$ has full Lebesgue measure in $\mathbb{R}^d$. Fix now a point $y \in \mathcal{D}$ and any $\tau \in \mathbb{R}_+$. Since f is Lipschitz, it is absolutely continuous on any line segment and therefore

$$f(x + \tau(y - x)) - f(x) = \int_0^\tau f'(x + s(y - x), y - x) \, ds = \int_0^\tau \langle \nabla f(x + s(y - x)), y - x \rangle \, ds.$$

The proof is complete. □

We now have all the ingredients to present a modification of the algorithm from [ZLJ+20], which, under a standard first-order oracle model, either significantly decreases the objective value or finds an approximate minimal norm element of $\partial_\delta f$.

Algorithm 1 `MinNorm`(x)

Input. x, $\delta > 0$, and $\epsilon > 0$.
Let $k = 0$, $g_0 = \nabla f(\zeta_0)$ where $\zeta_0 \sim \mathbb{B}_\delta(x)$.
while $\|g_k\|_2 > \epsilon$ and $\frac{\delta}{4}\|g_k\|_2 \geq f(x) - f(x - \delta \hat{g}_k)$ **do**
    Choose any r satisfying $0 < r < \|g_k\|_2 \cdot \sqrt{1 - (1 - \frac{\|g_k\|_2^2}{128L^2})^2}$.
    Sample ζ_k uniformly from $\mathbb{B}_r(g_k)$.
    Choose y_k uniformly at random from the segment $[x, x - \delta\widehat{\zeta_k}]$.
    $g_{k+1} = \mathrm{argmin}_{z \in [g_k, \nabla f(y_k)]} \|z\|_2$.
    $k = k + 1$.
end while
Return g_k.

The following theorem establishes the efficiency of Algorithm 1, and its proof is similar to that of [ZLJ+20, Lemma 13]. For completeness, we include the full proof in Appendix A.

Theorem 4. *Let $\{g_k\}$ be generated by* `MinNorm`(x). *Fix an index $k \geq 0$, and define the stopping time $\tau \stackrel{\text{def}}{=} \inf\{k\colon f(x - \delta\hat{g}_k) < f(x) - \delta\|g_k\|_2/4$ or $\|g_k\|_2 \leq \epsilon\}$. Then, we have*

$$\mathbb{E}\left[\|g_k\|_2^2 1_{\tau > k}\right] \leq \frac{16L^2}{16 + k}.$$

An immediate consequence of Theorem 4 is that `MinNorm`(x) terminates with high-probability.

Corollary 5. `MinNorm`(x) *terminates in at most $\left\lceil \frac{64L^2}{\epsilon^2} \right\rceil \cdot \lceil 2\log(1/\gamma) \rceil$ iterations with probability at least $1 - \gamma$.*

Combining Algorithm 1 with (3) yields Algorithm 2, with convergence guarantees summarized in Theorem 6, whose proof is identical to that of [ZLJ+20, Theorem 8].

Algorithm 2 Interpolated Normalized Gradient Descent ($\texttt{INGD}(x_0, T)$)

Initial x_0, counter T
for $t = 0, \ldots, T-1$ **do**
  $g = \texttt{MinNorm}(x_t)$ ▷ Computational complexity $\widetilde{\mathcal{O}}(L^2/\epsilon^2)$
    Set $x_{t+1} = x_t - \delta \hat{g}$
end for
Return x_T

Theorem 6. *Fix an initial point $x_0 \in \mathbb{R}^d$, and define $\Delta = f(x_0) - \inf_x f(x)$. Set the number of iterations $T = \frac{4\Delta}{\delta\epsilon}$. Then, with probability $1-\gamma$, the point $x_T = \texttt{INGD}(x_0, T)$ satisfies dist$(0, \partial_\delta f(x_T)) \le \epsilon$ in a total of at most*

$$\left\lceil \frac{4\Delta}{\delta\epsilon} \right\rceil \cdot \left\lceil \frac{64L^2}{\epsilon^2} \right\rceil \cdot \left\lceil 2\log\left(\frac{4\Delta}{\gamma\delta\epsilon}\right) \right\rceil \qquad \textit{function-value and gradient evaluations.}$$

In summary, the complexity of finding a point x satisfying $\text{dist}(0, \partial_\delta f(x)) \le \epsilon$ is at most $\mathcal{O}\left(\frac{\Delta L^2}{\delta\epsilon^3} \log\left(\frac{4\Delta}{\gamma\delta\epsilon}\right)\right)$ with probability $1-\gamma$. Using the identity $\partial f(x) = \limsup_{\delta\to 0} \partial_\delta f(x)$, this result also provides a strategy for finding a Clarke stationary point, albeit with no complexity guarantee. It is thus natural to ask whether one may efficiently find some point x for which there exists $y \in \mathbb{B}_\delta(x)$ satisfying $\text{dist}(0, \partial f(y)) \le \epsilon$. This is exactly the guarantee of subgradient methods on weakly convex functions in [DD19]. [Sha20] shows that for general Lipschitz functions, the number of subgradient computations required to achieve this goal by any algorithm scales with the dimension of the ambient space. Finally, we mention that the perturbation technique similarly applies to the stochastic algorithm of [ZLJ+20, Algorithm 2], yielding a method that matches their complexity estimate.

3 Faster INGD in low dimensions

In this section, we describe our modification of Algorithm 1 ("INGD") for obtaining improved runtimes in the low-dimensional setting. Our modified algorithm hinges on computations similar to (4), (5), and (6) except for the constants involved, and hence we explicitly state this setup. Given a vector $g \in \partial_\delta f(x)$, we say it satisfies the *descent condition* at x if

$$f(x - \delta\hat{g}) \le f(x) - \frac{\delta\epsilon}{3}. \tag{7}$$

Recall that Lemma 3 shows that for almost all g, we have

$$f(x) - f(x - \delta\hat{g}) = \int_0^1 \langle \nabla f(x - t\delta\hat{g}), \hat{g}\rangle \, dt = \delta \cdot \mathbb{E}_{z\sim \text{Unif}[x-\delta\hat{g}, x]} \langle \nabla f(z), \hat{g}\rangle.$$

Hence, when g does *not* satisfy the descent condition (7), we can output a random vector $u \in \partial_\delta f(x)$ such that

$$\mathbb{E}\langle u, g\rangle \le \frac{\epsilon}{3}\|g\|_2. \tag{8}$$

Then, an arbitrary vector g either satisfies (7) or can be used to output a random vector u satisfying (8). As described in Corollary 5, Algorithm 1 achieves this goal in $\widetilde{\mathcal{O}}(L^2/\epsilon^2)$ iterations.

In this section, we improve upon this oracle complexity by applying cutting plane methods to design Algorithm 3, which finds a better descent direction in $\widetilde{\mathcal{O}}(Ld/\epsilon)$ oracle calls for L-Lipschitz functions and $\mathcal{O}(d\log(L/\epsilon)\log(\delta\rho/\epsilon))$ oracle calls for ρ-weakly convex functions. In Appendix B, we demonstrate how to remove the expectation in (8) and turn the inequality into a high probability statement. For now, we assume the existence of an oracle $\mathscr{O}$ as in Definition 2.

Definition 2 (Inner Product Oracle)**.** *Given a vector $g \in \partial_\delta f(x)$ that does not satisfy the descent condition (7), the inner product oracle $\mathscr{O}(g)$ outputs a vector $u \in \partial_\delta f(x)$ such that*

$$\langle u, g\rangle \le \frac{\epsilon}{2}\|g\|_2.$$

We defer the proof of the lemma below to Appendix B.

Lemma 7. *Fix $x \in \mathbb{R}^d$ and a unit vector $\hat{g} \in \mathbb{R}^d$ such that f is differentiable almost everywhere on the line segment $[x, y]$, where $y \stackrel{\text{def}}{=} x - \delta\hat{g}$. Suppose that $z \in \mathbb{R}^d$ sampled uniformly from $[x, y]$ satisfies $\mathbb{E}_z\langle \nabla f(z), \hat{g}\rangle \le \frac{\epsilon}{3}$. Then we can find $\bar{z} \in \mathbb{R}^d$ using at most $O(\frac{L}{\epsilon}\log(1/\gamma))$ gradient evaluations of f, such that with probability at least $1-\gamma$ the estimate $\langle \nabla f(\bar{z}), \hat{g}\rangle \le \frac{\epsilon}{2}$ holds. Moreover, if f is ρ-weakly convex, we can find $\bar{z} \in \mathbb{R}^d$ such that $\langle \nabla f(\bar{z}), \hat{g}\rangle \le \frac{\epsilon}{2}$ using only $O(\log(\delta\rho/\epsilon))$ function evaluations of f.*

Our key insight is that this oracle is almost identical to the gradient oracle of the minimal norm element problem

$$\min_{g \in \partial_\delta f(x)} \|g\|_2.$$

Therefore, we can use it in the cutting plane method to find an approximate minimal norm element of $\partial_\delta f$. When there is no element of $\partial_\delta f$ with norm less than ϵ, our algorithm will instead find a vector that satisfies the descent condition. The main result of this section is the following theorem.

Theorem 8. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be an L-Lipschitz function. Fix an initial point $x_0 \in \mathbb{R}^d$, and let $\Delta \stackrel{\text{def}}{=} f(x_0) - \inf_x f(x)$. Then, there exists an algorithm that outputs a point $x \in \mathbb{R}^d$ satisfying $\operatorname{dist}(0, \partial_\delta f(x)) \le \epsilon$ and, with probability at least $1-\gamma$, uses at most*

$$\mathcal{O}\left(\frac{\Delta L d}{\delta\epsilon^2} \cdot \log(L/\epsilon) \cdot \log(1/\gamma)\right) \qquad \textit{function value/gradient evaluations.}$$

If f is ρ-weakly convex, the analogous statement holds with probability one and with the improved efficiency estimate $\mathcal{O}\big(\frac{\Delta d}{\delta\epsilon}\log(L/\epsilon) \cdot \log(\delta\rho/\epsilon)\big)$ of function value/gradient evaluations.

3.1 Finding a minimal norm element

In this section, we show, via Algorithm 3, how to find an approximate minimal norm element of $\partial_\delta f(x)$. Instead of directly working with the minimal norm problem, we note that, by Cauchy-Schwarz inequality and the Minimax Theorem, for any closed convex set Q, we have

$$\min_{g \in Q} \|g\|_2 = \min_{g \in Q}\left[\max_{\|v\|_2 \le 1} \langle g, v\rangle\right] = \max_{\|v\|_2 \le 1}\left[\min_{g \in Q}\langle g, v\rangle\right] = \max_{\|v\|_2 \le 1} \phi_Q(v), \tag{9}$$

where $\phi_Q(v) \stackrel{\text{def}}{=} \min_{g \in Q}\langle g, v\rangle$, and Lemma 9 formally connects the problem of finding the minimal norm element with that of maximizing ϕ_Q. The key observation in this section (Lemma 10) is that the inner product oracle $\mathscr{O}$ is a separation oracle for the (dual) problem $\max_{\|v\|_2 \le 1} \phi_Q(v)$ with $Q = \partial_\delta f(x)$ and hence can be used in cutting plane methods.

Lemma 9. *Let $Q \subset \mathbb{R}^d$ be a closed convex set that does not contain the origin. Let g_Q^* be a minimizer of $\min_{g \in Q}\|g\|_2$. Then, the vector $v_Q^* = g_Q^*/\|g_Q^*\|_2$ satisfies*

$$\langle v_Q^*, g\rangle \ge \|g_Q^*\|_2 \qquad \textit{for all } g \in Q.$$

and $v_Q^* = \arg\max_{\|v\|_2 \le 1} \phi_Q(v)$.

Proof. We omit the subscript Q to simplify notation. Since, by definition, g^* minimizes $\|g\|_2$ over all $g \in Q$, we have

$$\langle g^*, g\rangle \ge \|g^*\|_2^2 \text{ for all } g \in Q,$$

and the inequality is tight for $g = g^*$. Using this fact and $\phi(v^*) = \min_{g \in Q}\langle g, \frac{g^*}{\|g^*\|_2}\rangle$ gives

$$\phi(v^*) = \|g^*\|_2 = \min_{g \in Q}\|g\|_2 = \min_{g \in Q}\max_{v:\|v\|_2 \le 1}\langle g, v\rangle = \max_{\|v\|_2 \le 1}\min_{g \in Q}\langle g, v\rangle = \max_{v:\|v\|_2 \le 1}\phi(v),$$

where we used Sion's minimax theorem in the second to last step. This completes the proof. □

Using this lemma, we can show that $\mathscr{O}$ is a separation oracle.

Algorithm 3 `MinNormCG`(x)

Initialize center point x.
Set $k = 0$, the search region $\Omega_0 = \mathbb{B}_2(0)$, the set of gradients $Q_0 = \{\nabla f(x)\}$, and r satisfying $0 < r < \epsilon/(32dL)$
while $\min_{g \in Q_k} \|g\|_2 > \epsilon$ **do**
    Let v_k be the center of gravity of Ω_k.
    if v_k satisfies the descent condition (7) at x **then**
        Return v_k
    end if
    Sample ζ_k uniformly from $\mathbb{B}_r(v_k)$
    $u_k \leftarrow \mathscr{O}(\zeta_k)$
    $\Omega_{k+1} = \Omega_k \cap \{w : \langle u_k, \zeta_k - w\rangle \le 0\}$.
    $Q_{k+1} = \mathrm{conv}(Q_k \cup \{u_k\})$
    $k = k + 1$
end while
Return $\arg\min_{g \in Q_k} \|g\|_2$.

Lemma 10. *Consider a vector $g \in \partial f_\delta(x)$ that does not satisfy the descent condition (7), and let the output of querying the oracle at g be $u \in \mathscr{O}(g)$. Suppose that $\mathrm{dist}(0, \partial_\delta f(x)) \ge \frac{\epsilon}{2}$. Let g^* be the minimal-norm element of $\partial_\delta f(x)$. Then the normalized vector $v^* \stackrel{\text{def}}{=} g^*/\|g^*\|_2$ satisfies the inclusion:*

$$v^* \in \left\{w \in \mathbb{R}^d : \langle u, \hat{g} - w\rangle \le 0\right\}.$$

Proof. Set $Q = \partial_\delta f(x)$. By using $\langle u, \hat{g}\rangle \le \frac{\epsilon}{2}$ (the guarantee of $\mathscr{O}$ per Definition 2) and $\langle u, v^*\rangle \ge \|g^*\|_2$ (from Lemma 9), we have $\langle u, \hat{g} - v^*\rangle = \langle u, \hat{g}\rangle - \langle u, v^*\rangle \le \frac{\epsilon}{2} - \|g^*\|_2 \le 0$. □

Thus Lemma 10 states that if x is not a $(\delta, \frac{\epsilon}{2})$-stationary point of f, then the oracle $\mathscr{O}$ produces a halfspace $\mathcal{H}_v$ that separates $\hat{g}$ from v^*. Since $\mathscr{O}$ is a separation oracle, we can combine it with any cutting plane method to find v^*. For concreteness, we use the center of gravity method and display our algorithm in Algorithm 3. Note that in our algorithm, we use a point ζ_k close to the true center of gravity of Ω_k, and therefore, we invoke a result about the *perturbed* center of gravity method.

Theorem 11 (Theorem 3 of [BV04]; see also [Grü60])**.** *Let K be a convex set with center of gravity μ and covariance matrix A. For any halfspace H that contains some point x with $\|x - \mu\|_{A^{-1}} \le t$, we have*

$$\mathrm{vol}(K \cap H) \le (1 - 1/e + t)\mathrm{vol}(K).$$

Theorem 12 (Theorem 4.1 of [KLS95])**.** *Let K be a convex set in $\mathbb{R}^d$ with center of gravity μ and covariance matrix A. Then,*

$$K \subset \left\{x : \|x - \mu\|_{A^{-1}} \le \sqrt{d(d+2)}\right\}.$$

We now have all the tools to show correctness and iteration complexity of Algorithm 3.

Theorem 13. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be an L-Lipschitz function. Then Algorithm 3 returns a vector $v \in \partial_\delta f(x)$ that either satisfies the descent condition (7) at x or satisfies $\|v\|_2 \le \epsilon$ in*

$$\lceil 8d \log(8L/\epsilon))\rceil \text{ calls to } \mathscr{O}.$$

Proof. By the description of Algorithm 3, either it returns a vector v satisfying the descent condition or returns $g \in \partial_\delta f(x)$ with $\|g\|_2 \le \epsilon$. We now obtain the algorithm's claimed iteration complexity.

Consider an iteration k such that Ω_k *does* contain a ball of radius $\frac{\epsilon}{4L}$. Let A_k be the covariance matrix of convex set Ω_k. By Theorem 12, we have

$$A_k \succeq \left(\frac{\epsilon}{8dL}\right)^2 I.$$

Applying this result to the observation that in Algorithm 3 ζ_k is sampled uniformly from $\mathbb{B}_r(v_k)$ gives

$$\|v_k - \zeta_k\|_{A_k^{-1}} \le r \cdot \frac{8dL}{\epsilon} \le \frac{1}{4}. \tag{10}$$

Recall from Algorithm 3 and the preceding notation that Ω_k has center of gravity v_k and covariance matrix A_k. Further, the halfspace $\{w : \langle u_k, \zeta_k - w\rangle \le 0\}$ in Algorithm 3 contains the point ζ_k satisfying (10). Given these statements, since Algorithm 3 sets $\Omega_{k+1} = \Omega_k \cap \{w : \langle u_k, \zeta_k - w\rangle\}$, we may invoke Theorem 11 to obtain

$$\mathrm{vol}(\Omega_k) \le (1 - 1/e + 1/4)^k \mathrm{vol}(\mathbb{B}_2(0)) \le (1 - 1/10)^k \mathrm{vol}(\mathbb{B}_2(0)). \tag{11}$$

We claim that Algorithm 3 takes at most $T + 1$ steps where $T = d \log_{(1-\frac{1}{10})}(\epsilon/(8L))$. For the sake of contradiction, suppose that this statement is false. Then, applying (11) with $k = T + 1$ gives

$$\mathrm{vol}(\Omega_{T+1}) \le \left(\frac{\epsilon}{4L}\right)^d \mathrm{vol}(\mathbb{B}_1(0)). \tag{12}$$

On the other hand, Algorithm 3 generates points $u_i = \mathscr{O}(\zeta_i)$ in the i-th call to $\mathscr{O}$ and the set $Q_i = \mathrm{conv}\{u_1, u_2, \cdots, u_i\}$. Since we assume that the algorithm takes more than $T + 1$ steps, we have $\min_{g \in Q_{T+1}} \|g\|_2 \ge \epsilon$. Using this and $u_i \in Q_{T+1}$, Lemma 10 lets us conclude that $v^*_{Q_{T+1}} \in \left\{w \in \mathbb{R}^d : \langle u_i, \zeta_i - w\rangle \le 0\right\}$ for all $i \in [T+1]$. Since Ω_{T+1} is the intersection of the unit ball and these halfspaces, we have

$$v^*_{Q_{T+1}} \in \Omega_{T+1}.$$

Per (12), Ω_{T+1} does not contain a ball of radius $\frac{\epsilon}{4L}$, and therefore we may conclude that

$$\text{there exists a point } \widetilde{v} \in \mathbb{B}_{\frac{\epsilon}{2L}}(v^*_{Q_{T+1}}) \text{ such that } \widetilde{v} \notin \Omega_{T+1}.$$

Since $\widetilde{v} \in \mathbb{B}_2(0)$, the fact $\widetilde{v} \notin \Omega_{T+1}$ must be true due to one of the halfspaces generated in Algorithm 3. In other words, there must exist some $i \in [T + 1]$ with

$$\langle u_i, \zeta_i - \widetilde{v}\rangle > 0.$$

By the guarantee of $\mathscr{O}$, we have $\langle u_i, \zeta_i\rangle \le \frac{\epsilon}{2}$, and hence

$$\langle u_i, \widetilde{v}\rangle = \langle u_i, v_i\rangle - \langle u, v_i - \widetilde{v}\rangle < \frac{\epsilon}{2}. \tag{13}$$

By applying $\widetilde{v} \in \mathbb{B}_{\frac{\epsilon}{2L}}(v^*_{Q_{T+1}})$, $u_i \in \partial_\delta f(x)$, L-Lipschitzness of f, and Lemma 9, we have

$$\langle u_i, \widetilde{v}\rangle \ge \langle u_i, v^*_{Q_{T+1}}\rangle - \frac{\epsilon}{2L}\|u_i\|_2 \ge \langle u_i, v^*_{Q_{T+1}}\rangle - \frac{\epsilon}{2} \ge \|g^*_{Q_{T+1}}\|_2 - \frac{\epsilon}{2}. \tag{14}$$

Combining (13) and (14) yields that $\min_{g \in Q_{T+1}} \|g\|_2 = \|g^*_{Q_{T+1}}\|_2 < \epsilon$. This contradicts the assumption that the algorithm takes more than $T + 1$ steps and concludes the proof. □

Now, we are ready to prove the main theorem.

Proof of Theorem 8. We note that the outer loop in Algorithm 2 runs at most $\mathcal{O}(\frac{\Delta}{\delta\epsilon})$ times because we decrease the objective by $\Omega(\delta\epsilon)$ every step. Combining this with Theorem 13 and Lemma 7, we have that with probability $1 - \gamma$, the oracle complexity for L-Lipschitz function is

$$\left\lceil \frac{4\Delta}{\delta\epsilon} \right\rceil \cdot \lceil 8d \log(8L/\epsilon)) \rceil \cdot \left\lceil \frac{36L}{\epsilon} \right\rceil \cdot \left\lceil 2 \log\left(\frac{4\Delta}{\gamma\delta\epsilon}\right) \right\rceil = \mathcal{O}\left(\frac{\Delta L d}{\delta\epsilon^2} \cdot \log(L/\epsilon) \cdot \log(1/\gamma)\right)$$

and for L-Lipschitz and ρ-weakly convex function is $\mathcal{O}\big(\frac{\Delta d}{\delta\epsilon} \log(L/\epsilon) \cdot \log(\delta\rho/\epsilon)\big)$.

□

References

[AZ18] Zeyuan Allen-Zhu. How to make the gradients small stochastically: Even faster convex and nonconvex sgd. *Advances in Neural Information Processing Systems*, 31, 2018.

[BCL+20] James V Burke, Frank E Curtis, Adrian S Lewis, Michael L Overton, and Lucas EA Simões. Gradient sampling methods for nonsmooth optimization. In *Numerical Nonsmooth Optimization*, pages 201–225. Springer, 2020.

[BHS05] Michel Benaïm, Josef Hofbauer, and Sylvain Sorin. Stochastic approximations and differential inclusions. *SIAM Journal on Control and Optimization*, 44(1):328–348, 2005.

[BLO05] James V Burke, Adrian S Lewis, and Michael L Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM Journal on Optimization*, 15(3):751–779, 2005.

[BM20] Sébastien Bubeck and Dan Mikulincer. How to trap a gradient flow. In *Conference on Learning Theory*, pages 940–960. PMLR, 2020.

[BP20] Jerome Bolte and Edouard Pauwels. A mathematical model for automatic differentiation in machine learning. *arXiv preprint arXiv:2006.02080*, 2020.

[BP21] Jérôme Bolte and Edouard Pauwels. Conservative set valued fields, automatic differentiation, stochastic gradient methods and deep learning. *Mathematical Programming*, 188(1):19–51, 2021.

[BV04] Dimitris Bertsimas and Santosh S. Vempala. Solving convex programs by random walks. *J. ACM*, 51(4):540–556, 2004.

[CDHS18] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018.

[CDHS20] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points i. *Mathematical Programming*, 184(1):71–120, 2020.

[DD19] Damek Davis and Dmitriy Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

[DDKL20] Damek Davis, Dmitriy Drusvyatskiy, Sham Kakade, and Jason D Lee. Stochastic subgradient method converges on tame functions. *Foundations of computational mathematics*, 20(1):119–154, 2020.

[DDMP18] Damek Davis, Dmitriy Drusvyatskiy, Kellie J MacPhee, and Courtney Paquette. Subgradient methods for sharp weakly convex functions. *Journal of Optimization Theory and Applications*, 179(3):962–982, 2018.

[FLLZ18] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. *Advances in Neural Information Processing Systems*, 31, 2018.

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[Gol77] AA Goldstein. Optimization of lipschitz continuous functions. *Mathematical Programming*, 13(1):14–22, 1977.

[Grü60] Branko Grünbaum. Partitions of mass-distributions and of convex bodies by hyperplanes. *Pacific Journal of Mathematics*, 10(4):1257–1261, 1960.

[JGN+17] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *International Conference on Machine Learning*, pages 1724–1732. PMLR, 2017.

[Kiw07] Krzysztof C Kiwiel. Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization. *SIAM Journal on Optimization*, 18(2):379–388, 2007.

[KLS95] R. Kannan, L. Lovász, and M. Simonovits. Isoperimetric problems for convex bodies and a localization lemma. *Discrete Comput. Geom.*, 13(3–4):541–559, Dec 1995.

[KS21] Guy Kornowski and Ohad Shamir. Oracle complexity in nonsmooth nonconvex optimization. *Advances in Neural Information Processing Systems*, 34, 2021.

[MMM18] Szymon Majewski, Błażej Miasojedow, and Eric Moulines. Analysis of nonsmooth stochastic approximation: the differential inclusion approach. *arXiv preprint arXiv:1805.01916*, 2018.

[RHS+16] Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323. PMLR, 2016.

[Sha20] Ohad Shamir. Can we find near-approximately-stationary points of nonsmooth nonconvex functions? *arXiv preprint arXiv:2002.11962*, 2020.

[SKR85] Naum Z. Shor, Krzysztof C Kiwiel, and Andrzej Ruszcayński. Minimization methods for non-differentiable functions, 1985.

[ZLJ+20] Jingzhao Zhang, Hongzhou Lin, Stefanie Jegelka, Suvrit Sra, and Ali Jadbabaie. Complexity of finding stationary points of nonconvex nonsmooth functions. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11173–11182, Virtual, 13–18 Jul 2020.

[ZXG18] Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 1
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] Due to the theoretical nature of our work, we do not foresee any immediate potential negative societal impacts of it.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] All our complete proofs are in the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A]
(b) Did you mention the license of the assets? [N/A]
(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Missing proofs from Section 2

Theorem 4. *Let $\{g_k\}$ be generated by* $\texttt{MinNorm}(x)$. *Fix an index $k \geq 0$, and define the stopping time $\tau \stackrel{\text{def}}{=} \inf\{k\colon f(x-\delta\hat{g}_k) < f(x) - \delta\|g_k\|_2/4$ or $\|g_k\|_2 \leq \epsilon\}$. Then, we have*

$$\mathbb{E}\left[\|g_k\|_2^2 1_{\tau>k}\right] \leq \frac{16L^2}{16+k}.$$

Proof. Fix an index k, and let $\mathbb{E}_k[\cdot]$ denote the conditional expectation on g_k. Suppose we are in the event $\{\tau > k\}$. Taking into account the Lipschitz continuity of f and Lemma 3, we deduce that almost surely, conditioned on g_k, the following estimate holds:

$$\begin{aligned}
\frac{1}{4}\|g_k\|_2 \geq \frac{f(x) - f(x-\delta\hat{g}_k)}{\delta} &\geq \frac{f(x) - f(x-\delta\cdot\hat{\zeta}_k)}{\delta} - L\|\hat{g}_k - \hat{\zeta}_k\|_2 \\
&= \frac{1}{\delta}\int_0^\delta \langle \nabla f(x - s\hat{\zeta}_k), \hat{\zeta}_k\rangle\, ds - L\|\hat{g}_k - \hat{\zeta}_k\|_2 \\
&\geq \frac{1}{\delta}\int_0^\delta \langle \nabla f(x - s\hat{\zeta}_k), \hat{g}_k\rangle\, ds - 2L\|\hat{g}_k - \hat{\zeta}_k\|_2 \\
&= \mathbb{E}_k\langle \nabla f(y_k), \hat{g}_k\rangle - 2L\|\hat{g}_k - \hat{\zeta}_k\|_2.
\end{aligned}$$

Rearranging yields $\mathbb{E}_k\langle \nabla f(y_k), \hat{g}_k\rangle \leq \frac{1}{4}\|g_k\|_2 + 2L\|\hat{g}_k - \hat{\zeta}_k\|$. Simple algebra shows $\|\hat{g}_k - \hat{\zeta}_k\|_2^2 \leq 2(1 - \sqrt{1 - r^2/\|g_k\|_2^2}) \leq \frac{\|g_k\|_2^2}{64L^2}$. Therefore, we infer that $\mathbb{E}_k\langle \nabla f(y_k), \hat{g}_k\rangle < \frac{1}{2}\|g_k\|_2$. Lemma 2 then guarantees that

$$\mathbb{E}_k[\|g_{k+1}\|_2^2 1_{\tau>k}] \leq \left(\|g_k\|_2^2 - \frac{\|g_k\|_2^4}{16L^2}\right) 1_{\tau>k}.$$

Define $b_k := \|g_k\|_2^2 1_{\tau>k}$ for all $k \geq 0$. Then the tower rule for expectations yields

$$\mathbb{E}b_{k+1} \leq \mathbb{E}[\|g_{k+1}\|_2^2 1_{\tau>k}] \leq \mathbb{E}\left[\left(1 - \frac{b_k}{16L^2}\right) b_k\right] \leq \left(1 - \frac{\mathbb{E}b_k}{16L^2}\right)\mathbb{E}b_k,$$

by Jensen's inequality applied to the concave function $t \mapsto (1 - t/16L^2)t$. Setting $a_k = \mathbb{E}b_k/L^2$, this inequality becomes $a_{k+1} \leq a_k - a_k^2/16$, which, upon rearranging, yields $\frac{1}{a_{k+1}} \geq \frac{1}{a_k(1-a_k/16)} \geq \frac{1}{a_k} + \frac{1}{16}$. Iterating the recursion and taking into account $a_0 \leq 1$ completes the proof. □

Corollary 14. $\texttt{MinNorm}(x)$ *terminates in at most $\left\lceil \frac{64L^2}{\epsilon^2}\right\rceil \cdot \lceil 2\log(1/\gamma)\rceil$ iterations with probability at least $1-\gamma$.*

Proof. Notice that when $k \geq \frac{64L^2}{\varepsilon^2}$, we have, by Theorem 4, that

$$\Pr(\tau > k) \leq \Pr(\|g_k\|_2 1_{\tau>k} \geq \epsilon) \leq \frac{16L^2}{(16+k)\varepsilon^2} \leq \frac{1}{4}.$$

Similarly, for all $i \in \mathbb{N}$, we have $\Pr(\tau > ik \mid \tau > (i-1)k) \leq 1/4$. Therefore,

$$\Pr(\tau > ik) = \Pr(\tau > ik \mid \tau > (i-1)k)\Pr(\tau > (i-1)k) \leq \frac{1}{4}\Pr(\tau > (i-1)k) \leq \frac{1}{4^i}.$$

Consequently, we have $\Pr(\tau > ik) \leq \frac{1}{4^i} \leq \gamma$ whenever $i \geq \log(1/\gamma)/\log(4)$, as desired. □

B Implementation of the oracles: proof of Lemma 7

In this section, we show how to convert (8) into a deterministic guarantee.

Lemma 15. *Fix a unit vector $\hat{g} \in \mathbb{R}^d$ and let $z \in \mathbb{R}^d$ be a random vector satisfying $\mathbb{E}\langle \nabla f(z), \hat{g}\rangle \leq \frac{\epsilon}{3}$. Let $z_1, \ldots, z_k$ be i.i.d realizations of z with $k = \left\lceil \frac{36L}{\epsilon}\right\rceil \cdot \left\lceil \frac{\log(1/\gamma)}{\log(4)}\right\rceil$. Then with probability at least $1-\gamma$, one of the samples z_i satisfies $\langle \nabla f(z_i), \hat{g}\rangle \leq \frac{\epsilon}{2}$.*

Proof. Define the random variable $Y \stackrel{\text{def}}{=} \langle \nabla f(z), \hat{g} \rangle$, and use $p \stackrel{\text{def}}{=} \Pr[Y \leq \frac{\epsilon}{2}]$. We note that

$$\mathbb{E}[Y] = p \cdot \mathbb{E}[Y \mid Y \leq \frac{\epsilon}{2}] + (1-p) \cdot \mathbb{E}[Y \mid Y > \frac{\epsilon}{2}].$$

Rearranging the terms and using $\mathbb{E}[Y] \leq \epsilon/3$ gives

$$p \cdot \left(\mathbb{E}[Y \mid Y > \frac{\epsilon}{2}] - \mathbb{E}[Y \mid Y \leq \frac{\epsilon}{2}] \right) \geq \frac{\epsilon}{6}.$$

Finally, taking into account that f is L-Lipschitz, we deduce $|Y| \leq L$, which further implies $p \geq \frac{\epsilon}{12L}$. The results follows immediately. □

Lemma 16. *Let $f \colon \mathbb{R}^d \to \mathbb{R}$ be an L-Lipschitz continuous and ρ-weakly convex function. Fix a point x and a unit vector $\hat{g} \in \mathbb{R}^d$ such that f is differentiable almost everywhere on the line segment $[x, y]$, where $y \stackrel{\text{def}}{=} x - \delta \hat{g}$. Suppose that a random vector z sampled uniformly from $[x, y]$ satisfies $\mathbb{E}_z \langle \nabla f(z), \hat{g} \rangle \leq \frac{\epsilon}{3}$. Then, Algorithm 4 finds $\bar{z} \in \mathbb{R}^d$ such that $\langle \nabla f(\bar{z}), \hat{g} \rangle \leq \frac{\epsilon}{2}$ using $3 \log(12\delta\rho/\epsilon)$ function evaluations of f.*

Algorithm 4 Binary Search for $\bar{z}$

Input. Line Segment $[x, y = x - \delta \hat{g}]$
Let $[a, b] = [0, 1]$
while $b - a > \frac{\epsilon}{6\delta\rho}$ **do**
  if $f(x - a\delta\hat{g}) - f(x - \frac{a+b}{2}\delta\hat{g}) \leq f(x - \frac{a+b}{2}\delta\hat{g}) - f(x - b\delta\hat{g})$ **then**
    Let $[a, b] \leftarrow [a, \frac{a+b}{2}]$
  else
    Let $[a, b] \leftarrow [\frac{a+b}{2}, b]$
  end if
end while
Return $x - a\delta\hat{g}$

Proof. Define the new function $h : [0, 1] \to \mathbb{R}$ by $h(t) = \langle \nabla f(x + t(y - x)), \hat{g} \rangle$. Clearly, we have

$$\frac{\epsilon}{3} \geq \mathbb{E}[h(t)] = \frac{1}{2} \underbrace{\mathbb{E}[h(t) \mid t \leq 0.5]}_{P_{\leq}} + \frac{1}{2} \underbrace{\mathbb{E}[h(t) \mid t > 0.5]}_{P_{>}}.$$

Therefore $P_{\leq}$ or $P_{>}$ is at most $\epsilon/3$. The fundamental theorem of calculus directly implies $P_{\leq} = \frac{f(x) - f(x - \frac{\delta}{2}\hat{g})}{2\delta}$ and $P_{>} = \frac{f(x - \frac{\delta}{2}\hat{g}) - f(y)}{2\delta}$. Therefore with three function evaluations we may determine one of the two alternatives. Repeating this procedure $\log(12\delta\rho/\epsilon)$ times, each times shrinking the interval by half, we can find an interval $[a, b] \subset [0, 1]$ such that $b - a \leq \frac{\epsilon}{6\delta\rho}$ and $\mathbb{E}_{t \in [a,b]} h(t) \leq \frac{\epsilon}{3}$. Note that for any $\bar{t} \in [a, b]$, we have $h(\bar{t}) = \mathbb{E}h(t) + (h(\bar{t}) - \mathbb{E}h(t))$, while weak convexity implies

$$\begin{aligned} h(\bar{t}) - \mathbb{E}h(t) &= \frac{1}{\delta} \mathbb{E}_{t \in [a,b]} \langle \nabla f(x + \bar{t}(y - x)) - \nabla f(x + t(y - x)), x - y \rangle \\ &\leq \mathbb{E}_{t \in [a,b]} \frac{\bar{t} - t}{\delta} \rho \|y - x\|^2 \leq \frac{\epsilon}{6}. \end{aligned}$$

We thus conclude $h(\bar{t}) \leq \frac{\epsilon}{3} + \frac{\epsilon}{6} = \frac{\epsilon}{2}$ as claimed. □