A gradient sampling method with complexity guarantees for general Lipschitz functions

Damek Davis* Dmitriy Drusvyatskiy†

Abstract

Zhang et al. [9] proposed a novel modification of Goldstein’s classical subgradient method for minimizing Lipschitz continuous functions, which has finite-time efficiency guarantees. This work, however, makes use of a nonstandard subgradient oracle model and requires the function to be directionally differentiable. In this note, we show that both of these assumptions can be dropped by simply adding a small random perturbation in each step of their algorithm. The resulting method works on any Lipschitz function whose value and gradient can be evaluated at points of differentiability.

1 Introduction

The subgradient method is a classical algorithm for minimizing a nonsmooth Lipschitz continuous function $f$ on $\mathbb{R}^d$. Starting from an initial iterate $x_0$, the method simply iterates

$$x_{t+1} = x_t - \alpha_tv_t \quad \text{with } v_t \in \partial f(x_t).$$

(1.1)

Here, the positive sequence $\{\alpha_t\}_{t \geq 0}$ is user specified and the set $\partial f(x)$ is the Clarke subdifferential, which is defined as

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

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

Much is known about the subgradient method for problems that are smooth, convex, weakly convex, and more generally semialgebraic. For example, for each of these classes, limit points $\bar{x}$ of the subgradient method are known to be first-order critical, meaning $0 \in \partial f(\bar{x})$. Besides asymptotic guarantees, the subgradient method also enjoys rigorous complexity estimates, for smooth, convex, and weakly convex problems. Problems that lie beyond these

*School of ORIE, Cornell University, Ithaca, NY 14850, USA; people.orie.cornell.edu/dsd95/. Research of Davis supported by an Alfred P. Sloan research fellowship and NSF DMS award 2047637.

†Department of Mathematics, U. Washington, Seattle, WA 98195; www.math.washington.edu/~ddrusv. Research of Drusvyatskiy was supported by NSF DMS-1651851 and CCF-2023166 awards.
three classes have recently emerged in modern machine learning practice. Indeed, industry-backed solvers, such as Google’s TensorFlow and Facebook’s PyTorch, now routinely train nonsmooth deep networks via (stochastic) subgradient methods, powering widespread empirical success. However, no complexity of the method is known in the semialgebraic setting.

Recently, the work [9] established finite time guarantees for a modified subgradient method for the class of Lipschitz directionally differentiable functions under a nonstandard first-oracle model. Namely in each iteration, given a point \(x\) and direction \(u\) the algorithm assumes access to a vector \(v \in \partial f(x)\) satisfying \(\langle v, u \rangle = f'(x, u)\). The developed algorithm is a surprising leap forward, and motivates the question:

Is there a subgradient method with finite time complexity guarantees that succeeds on general Lipschitz functions assuming only that gradients and function values of the function can be computed at points of differentiability?

The existence of such a method would be surprising since the classical subgradient algorithm on general Lipschitz functions may even fail to asymptotically find any critical point, due to existence of highly pathological examples [5]. In this short note, we answer this question affirmatively. Indeed, we show that a small modification of the algorithm of [9], wherein one simply adds a small random perturbation in each iteration, works for any Lipschitz function assuming only that gradients and function values can be computed at almost every point of \(\mathbb{R}^d\) in the sense of Lebesgue measure. In particular, such an oracle arises from automatic differentiation schemes that are routinely used in deep learning [1, 2, Corollary 5].

2 Interpolated normalized gradient descent

In this section, we describe the recent work [9] and our proposed modification that establishes finite time guarantees for a modified subgradient method. The main construction that will be used is the Goldstein subdifferential introduced in [7]. Throughout the rest of the section, we fix an \(L\)-Lipschitz continuous function \(f\) on \(\mathbb{R}^d\).

**Definition 2.1 (Goldstein subdifferential).** Consider a locally Lipschitz function \(f : \mathbb{R}^d \to \mathbb{R}\), a point \(x \in \mathbb{R}^d\), and a parameter \(\delta > 0\). The Goldstein subdifferential of \(f\) at \(x\) is the set

\[
\partial_{\delta} f(x) = \text{conv}\left( \bigcup_{y \in B_\delta(x)} \partial f(y) \right).
\]

Thus, the Goldstein subdifferential of \(f\) at \(x\) is simply the convex hull of the union of all Clarke subgradients at points in a \(\delta\)-ball around \(x\). Famously, Goldstein [7] 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

\[
\hat{g} = \frac{g}{\|g\|} \quad \text{for any nonzero vector } g \in \mathbb{R}^d.
\]

**Theorem 2.2 (Uniform decrease).** Fix a point \(x\) and let \(g\) be a minimal norm element of \(\partial_{\delta} f(x)\). Then as long as \(g \neq 0\), the estimate holds:

\[
f(x - \delta \hat{g}) \leq f(x) - \delta \|g\|.
\]
Theorem 2.2 immediately motivates a conceptual descent algorithm, which repeats:

\[ x_{t+1} = x_t - \delta \hat{g}_t \quad \text{where} \quad g_t = \arg \min_{g \in \partial \delta f(x)} \|g\|. \tag{2.1} \]

In particular, Theorem 2.2 trivially guarantees that the approximate stationarity condition

\[ \min_{t=1,...,T} \|g_t\| \leq \epsilon \]

holds after \( T = \mathcal{O} \left( \frac{f(x_0) - \min f}{\delta \epsilon} \right) \) iterations.

Since evaluating the minimal norm element of \( \partial \delta f(x) \) is impossible in general, the descent method can not be applied directly. Nonetheless it does serve as a guiding principle for implementable algorithms. Notably, the gradient sampling algorithm [4] in each iteration forms polyhedral approximations \( K_t \) of \( \partial \delta f(x_t) \) by sampling gradients in the ball \( B_{\delta}(x) \) and computes search directions \( g_t \in \arg \min_{g \in K_t} \|g\| \). The number of gradient computations required by gradient sampling algorithms, however, scales linearly with the dimension of the ambient space; see the survey [3] for details.

The recent paper [9] shows remarkably 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 subgradients that is independent of the dimension. The idea of the procedure is as follows. Suppose that we have a trial vector \( g \in \partial \delta f(x) \) satisfying

\[ f(x - \delta \hat{g}) \geq f(x) - \frac{\delta}{2} \|g\|. \tag{2.2} \]

That is, the decrease in function value is not as large as guaranteed by Theorem 2.2 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 of all small \( \lambda > 0 \) as long as \( \langle u, g \rangle \leq \|g\|^2 \).

The task therefore reduces to:

find some \( u \in \partial \delta f(x) \) satisfying \( \langle u, g \rangle \leq \|g\|^2 \).

The ingenious idea of [9] 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 along with (2.2) yield

\[ \frac{1}{2} \|g\| \geq \frac{f(x) - f(x - \delta \hat{g})}{\delta} = \frac{1}{\delta} \int_0^\delta \langle \nabla f(x - \tau \hat{g}), \hat{g} \rangle \, d\tau. \tag{2.3} \]

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

\[ \mathbb{E}(\nabla f(y), g) \leq \frac{1}{2} \|g\|^2. \tag{2.4} \]

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 the proof of [9, Theorem 8].
Lemma 2.3 (Distance decrease). Fix a vector \( g \in \mathbb{R}^d \) and let \( u \in \mathbb{R}^d \) be a random vector satisfying \( \mathbb{E}(u, g) < \frac{1}{2}\|g\|^2 \). Suppose moreover that the inequality \( \|g\|, \|u\| \leq L \) holds for some \( L < \infty \). Then the minimal-norm vector \( z \) in the segment \([g, u]\) satisfies:

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

Proof. For any \( \lambda \in (0, 1) \), we compute

\[
\mathbb{E}\|z\|^2 \leq \mathbb{E}\|g + \lambda(u - g)\|^2 = \|g\|^2 + 2\lambda \mathbb{E}(g, u - g) + \lambda^2 \mathbb{E}\|u - g\|^2 \\
\leq \|g\|^2 - \lambda\|g\|^2 + \lambda^2 \mathbb{E}\|u - g\|^2 \\
\leq \|g\|^2 - \lambda\|g\|^2 + 4\lambda^2 L^2.
\]

Minimizing right hand side over \( \lambda \) yields \( \lambda = \frac{\|g\|^2}{8L^2} \in (0, 1) \). Plugging this value back in completes the proof.

The last technical difficulty that must be overcome is the requirement that \( f \) is differentiable along the line segment \([g, u]\). This assumption was crucially used to obtain 2.3 and 2.4. To cope with this problem, the authors of [9] introduce extra assumptions on the function \( f \) to be minimized and assume a nonstandard oracle access to subgradients. The following elementary lemma shows that no extra assumptions are needed if one slightly perturbs \( g \). Throughout, we let \( \text{dom}(\nabla f) \) denote the set of points where \( f \) is differentiable—a full Lebesgue measure set by Rademacher’s theorem.

Lemma 2.4. Let \( f : \mathbb{R}^d \to \mathbb{R} \) be a Lipschitz continuous 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 \( \text{dom}(\nabla f) \) in a full Lebesgue measure set in \( \mathbb{R} \). Consequently, for every \( y \in \mathcal{D} \) and all \( \tau \in \mathbb{R} \) it holds:

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

Proof. Without loss of generality we may assume \( x = 0 \) and \( f(x) = 0 \). Rademacher’s theorem guarantees that \( \text{dom}(\nabla f) \) 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 \( \mathbb{R}_+\{y\} \cap (\text{dom}(\nabla f))^c \) is Lebesgue null in \( \mathbb{R} \). It follows immediately that the set \( \mathcal{D} = \{\tau y : \tau > 0, y \in \mathcal{Q}\} \) 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

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

where \( f'(\cdot, \cdot) \) denotes the directional derivative. The proof is complete.
We now have all the ingredients to present the algorithm from [9] that either significantly decreases the objective function or finds an approximate minimal norm element of $\partial f$.

**Algorithm 1: MinNorm($x$)**

**Input:** center point $x$.
Set $k = 0$ and choose any $g_0 \in \partial f(x)$.

**while** $\|g_k\| > \varepsilon$ and $\frac{1}{4} \|g_k\| \geq f(x) - f(x - \delta g_k)$ **do:**

Choose any $r$ satisfying $0 < r < \|g_k\| \cdot \sqrt{1 - (1 - \frac{\|g_k\|^2}{128L^2})^2}$
Sample $\zeta_k$ uniformly from $\mathbb{B}_r(g_k)$
Sample $y_k$ uniformly from $[x, x - \delta \zeta_k]$
Set $g_{k+1} = \arg\min_{z \in [y_k, \nabla f(y_k)]} \|z\|$
Set $k = k + 1$

**Return:** $g_k$

The following theorem establishes the efficiency of Algorithm 1. The proof is a small modification of the proof of [9] Lemma 13.

**Theorem 2.5.** Let $\{g_k\}$ be generated by MinNorm($x$). Then for each index $k \geq 0$ the estimate holds:

$$\mathbb{E}\|g_k\|^2 1_{\tau > k} \leq \frac{16L^2}{16 + 9k^2}.$$  

where $\tau = \inf \{k : f(x - \delta \hat{g}_k) < f(x) - \delta\|g_k\|/4 \text{ or } \|g_k\| \leq \varepsilon\}$.

**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 Lipschitz continuity of $f$ and Lemma 2.3, we deduce that almost surely conditioned on $g_k$ the estimate holds:

$$\frac{1}{4} \|g_k\| \geq \frac{f(x) - f(x - \delta \hat{g}_k)}{\delta} \geq \frac{f(x) - f(x - \delta \cdot \hat{\zeta}_k)}{\delta} - L\|\hat{g}_k - \hat{\zeta}_k\|$$

$$\geq \frac{1}{\delta} \int_0^\delta \langle \nabla f(x - s\hat{\zeta}_k), \hat{\zeta}_k \rangle \, ds - L\|\hat{g}_k - \hat{\zeta}_k\|$$

$$\geq \frac{1}{\delta} \int_0^\delta \langle \nabla f(x - s\hat{\zeta}_k), \hat{g}_k \rangle \, ds - 2L\|\hat{g}_k - \hat{\zeta}_k\|$$

Rearranging yields $\mathbb{E}_k(\nabla f(y_k), \hat{g}_k) \leq \frac{1}{4} \|g_k\| + 2L\|\hat{g}_k - \hat{\zeta}_k\|$. Simple algebra shows $\|\hat{g}_k - \hat{\zeta}_k\|^2 \leq 2(1 - \sqrt{1 - r^2/\|g_k\|^2}) \leq \frac{\|g_k\|^2}{64L^2}$. Therefore we deduce $\mathbb{E}_k(\nabla f(y_k), \hat{g}_k) < \frac{1}{2} \|g_k\|$. Lemma 2.3 therefore guarantees that

$$\mathbb{E}_k[\|g_{k+1}\|^2 1_{\tau > k}] \leq \left(\|g_k\|^2 - \frac{\|g_k\|^4}{16L^2}\right) 1_{\tau > k}.$$
Define \( b_k := \|g_k\|^2 \) for all \( k \geq 0 \). Then the tower rule for expectations yields
\[
\mathbb{E}b_{k+1} \leq \mathbb{E}[\|g_{k+1}\|^2 \tau > k] \leq \mathbb{E} \left[ \left( 1 - \frac{b_k}{16L^2} \right) b_k \right] \leq \left( 1 - \frac{\mathbb{E}b_k}{16L^2} \right) \mathbb{E}b_k,
\]
where the last inequality follows from Jensen’s inequality and concavity of the function \( t \mapsto (1 - \tau/16L^2)t \). Setting \( a_k = \mathbb{E}b_k/L^2 \), this inequality becomes
\[
a_{k+1} \leq a_k - a_k^2/16 \text{ and therefore } \frac{1}{a_{k+1}} \geq \frac{1}{a_k(1-a_k/16)} \geq \frac{1}{a_k} + \frac{1}{16}.
\]
Iterating the recursion and taking into account \( a_0 \leq 1 \) completes the proof.

An immediate consequence of Theorem 2.5 is that MinNorm\((x)\) terminates with high-probability.

**Corollary 2.6 (High probability termination).** With probability at least \( 1 - \gamma \), MinNorm\((x)\) terminates in at most
\[
\left\lceil \frac{64L^2}{\epsilon^2} \right\rceil \cdot \left\lceil \frac{\log(1/\gamma)}{\log(4)} \right\rceil \text{ iterations.}
\]

**Proof.** Notice that when \( k \geq \frac{64L^2}{\epsilon^2} \), we have
\[
\Pr(\tau > k) \leq \Pr(\|g_k\|_{\tau > k} \geq \epsilon) \leq \frac{16L^2}{(16 + k)\epsilon^2} \leq \frac{1}{4}.
\]
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.

Combining Algorithm 1 with the original descent method of Goldstein yields the following procedure.

**Algorithm 2:** Interpolated Normalized Gradient Descent (INGD\((x_0, T, K)\))

| Input: | Initial \( x_0 \), counter \( T \). |
|--------|---------------------------------|
| for \( t = 1, \ldots, T \) do: |
| \( g = \text{MinNorm}(x_t) \) |
| Set \( x_{t+1} = x_t - \delta \hat{g} \) |
| **Return:** \( x_{T+1} \) |

The convergence guarantees of the algorithm are summarized in the following theorem, whose proof is identical to that of [9, Theorem 8].

**Theorem 2.7.** Fix an initial point \( x_0 \in \mathbb{R}^d \) and define \( \Delta = f(x_0) - \inf f \). Set the number of iterations \( T = \frac{4\Delta}{\delta \epsilon} \). Then the point \( x = \text{INGD}(x_0, T, K) \) satisfies \( \text{dist}(0, \partial f(x_T)) \leq \epsilon \). Moreover, with probability \( 1 - \gamma \) the total number of gradient evaluations used is at most
\[
\left\lceil \frac{64L^2}{\epsilon^2} \right\rceil \cdot \left\lceil \frac{\log(\frac{2\Delta}{\delta \epsilon})}{\log(4)} \right\rceil.
\]
In summary, the complexity of finding a point $x$ satisfying $\text{dist}(0, \partial f(x)) \leq \epsilon$ is at most

$$O\left( \frac{\Delta L^2}{\delta \epsilon^3} \log \left( \frac{4\Delta}{\gamma \delta \epsilon} \right) \right)$$

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 $y \in B_\delta(x)$ satisfying $\text{dist}(0, \partial f(y)) \leq \epsilon$. This is exactly the guarantee of subgradient methods on weakly convex functions in [6]. The paper [8] shows that for general Lipschitz continuous functions, the number of subgradient computations required to achieve this goal by any algorithm must scale with the dimension of the ambient space.

Finally, we mention that the perturbation technique similarly applies to the stochastic algorithm of [9, Algorithm 2], yielding an method that matches their complexity estimate.

References

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

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

[3] 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.

[4] 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.

[5] Aris Daniilidis and Dmitriy Drusvyatskiy. Pathological subgradient dynamics. SIAM Journal on Optimization, 30(2):1327–1338, 2020.

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

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

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

[9] 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. PMLR.