Noname manuscript No.
(will be inserted by the editor)

No Dimension-Free Deterministic Algorithm Computes Approximate Stationarities of Lipschitzians

Lai Tian · Anthony Man-Cho So

Received: date / Accepted: date

Abstract We consider the oracle complexity of computing an approximate stationary point of a Lipschitz function. When the function is smooth, it is well known that the simple deterministic gradient method has finite dimension-free oracle complexity. However, when the function can be nonsmooth, it is only recently that a randomized algorithm with finite dimension-free oracle complexity has been developed. In this paper, we show that no deterministic algorithm can do the same. Moreover, even without the dimension-free requirement, we show that any finite-time deterministic method cannot be general zero-respecting. In particular, this implies that a natural derandomization of the aforementioned randomized algorithm cannot have finite-time complexity. Our results reveal a fundamental hurdle in modern large-scale nonconvex nonsmooth optimization.

Keywords Stationary points · Black-box optimization · Information-based complexity · Dimension-free rates · Lower bounds

Mathematics Subject Classification (2020) 68Q25 · 90C60 · 90C56

This work is supported in part by the Hong Kong Research Grants Council (RGC) General Research Fund (GRF) project CUHK 14216122.

Lai Tian
Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong,
Shatin, N. T., Hong Kong
E-mail: tianlai@se.cuhk.edu.hk

Anthony Man-Cho So
Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong,
Shatin, N. T., Hong Kong
E-mail: manchoso@se.cuhk.edu.hk

1 Introduction

Convexity and differentiability have long been considered desirable properties for an optimization model to possess, as they can be exploited in the design of iterative methods with strong convergence guarantees. Nevertheless, many contemporary applications in machine learning, operations research, and statistics — such as ReLU neural networks, generative adversarial network, piecewise affine regression [14] — give rise to nonconvex and nonsmooth models. Such models are challenging from both theoretical and computational perspectives, as automatic differentiation with PyTorch/TensorFlow may not be correct [28], subgradient flow is not necessarily convergent [15], and even stationarity concepts for them are not trivial at all [34].

Consider the optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}),$$

where $f:\mathbb{R}^d\to\mathbb{R}$ is L-Lipschitz for some $L>0$ and could be both nonsmooth and nonconvex. In such a general setting, one of the arguably most fundamental questions is whether a stationary point of f is computable, and if so, how. When f is smooth, it is folkloric that an ε-approximate stationary point $\boldsymbol{x}$ of f (i.e., $\|\nabla f(\boldsymbol{x})\|\le\varepsilon$) can be computed by gradient descent using only $O(\varepsilon^{-2})$ calls to the gradient oracle [39], independent of the dimension d. Extensive efforts have been devoted to the fast computation of approximate stationary points of smooth functions in various settings [22, 8, 26]. Moreover, lower bounds on the complexity of computing approximate stationary points of smooth functions with different methods/oracles are rather well-understood [9, 10, 11].

When f is nonsmooth, there is a variety of stationarity concepts (see, e.g., [34]), and the complexity of computing/approximating these concepts is relatively less explored. As shown in [49, Theorem 5] and [33, Proposition 1], computing an ε-approximate stationary point $\boldsymbol{x}$ of f in the sense that $\operatorname{dist}(\mathbf{0},\partial f(\boldsymbol{x}))\le\varepsilon$ [1] is impossible for any finite-time randomized/deterministic algorithm interacting with a local oracle. For well-behaved problems with ρ-weakly convex[2] objective functions, Davis and Drusvyatskiy [16], Davis and Grimmer [19] introduced a concept called near-approximate stationarity (NAS), which is closely related to the gradient norm of the Moreau envelope of the objective. Informally, a point is (ε,δ)-NAS for f if it is within a distance of δ from an ε-approximate stationary point of f; see Definition 4. They showed that a subgradient-type method computes an (ε,δ)-NAS point with $O(\mathrm{poly}(\rho,\varepsilon^{-1},\delta^{-1}))$ calls to the subgradient oracle, independent of the dimension. For general Lipschitz objective functions, Kornowski and Shamir [33] proved that the oracle complexity of any randomized/deterministic algorithm for computing an (ε,δ)-NAS point cannot avoid an exponential dependence on the dimension. This implies that the computation of NAS points is in general

[1] Here $\partial f(\boldsymbol{x})$ is the Clarke subdifferential of f at $\boldsymbol{x}$; see Definition 1 for details.

[2] Recall that f is ρ-weakly convex if $\boldsymbol{x}\mapsto f(\boldsymbol{x})+\frac{\rho}{2}\|\boldsymbol{x}\|^2$ is convex.

intractable. Nevertheless, another concept that dates back to the seminal work of Goldstein [23], termed Goldstein approximate stationarity (GAS), exhibits favorable algorithmic consequences. Roughly speaking, a point $\boldsymbol{x}$ is (ε, δ)-GAS for f if there exists a vector of norm at most ε in a δ-approximation of the Clarke subdifferential at $\boldsymbol{x}$ (denoted by $\partial_\delta f(\boldsymbol{x})$); see Definitions 2 and 3. The conceptual scheme in [23] computes the iterates via the update

$$\boldsymbol{x}^{(t+1)} \leftarrow \boldsymbol{x}^{(t)} - \delta \cdot \boldsymbol{g}^{(t)} / \|\boldsymbol{g}^{(t)}\|,$$

where $\boldsymbol{g}^{(t)} := \arg\min_{\boldsymbol{g} \in \partial_\delta f(\boldsymbol{x}^{(t)})} \|\boldsymbol{g}\|$ is the minimal norm element in $\partial_\delta f(\boldsymbol{x}^{(t)})$. It is shown in [23] that an (ε, δ)-GAS point of f can be computed by such a scheme in $O(\varepsilon^{-1}\delta^{-1})$ steps. However, obtaining $\boldsymbol{g}^{(t)}$ for a general Lipschitz function equipped with an implementable oracle can be intractable, as there is no known approach to evaluate $\partial_\delta f(\boldsymbol{x})$. Therefore, a series of works, e.g., [6, 7, 29], proposes to build a polyhedral approximation of $\partial_\delta f(\boldsymbol{x}^{(t)})$ via random sampling and compute an approximate $\boldsymbol{g}^{(t)}$ by solving a quadratic program in every iteration. However, the number of samples needed for a meaningful approximation of $\partial_\delta f(\boldsymbol{x}^{(t)}) \subseteq \mathbb{R}^d$ is lower bounded by the dimension d. Thus, a dimension-free finite-time complexity cannot be achieved with the existing gradient sampling schemes.

Recently, Zhang et al. [49] have introduced a novel randomized algorithm that, when equipped with a sufficiently powerful oracle, computes (ε, δ)-GAS points of L-Lipschitz directionally differentiable functions with probability at least $1 - \gamma$ and has a dimension-free oracle complexity of

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

where $f(\mathbf{0}) - \inf_{\boldsymbol{x}} f(\boldsymbol{x}) \leq \Delta$.[3] A natural question, which is also posed in [49], is whether the algorithm in [49] can be derandomized. An answer to this question could have both theoretical and practical impact on the black-box optimization of Lipschitz functions and potentially deepen our understanding of the computability of various approximate stationarity concepts.

1.1 Our Results and Techniques

Our first main result shows that the answer to the above question is negative. Specifically, we show in Theorem 1 that for any sufficiently small $\varepsilon \geq 0, \delta \geq 0$:

> *No deterministic algorithm for computing (ε, δ)-GAS points of Lipschitz functions has dimension-free finite-time complexity.*

[3] In this paper, we assume that an algorithm will always start from $\boldsymbol{x}^{(1)} = \mathbf{0}$. This is without loss of generality due to the lack of information about f before querying $\boldsymbol{x}^{(1)}$ and the translational invariance of all considered function classes.

This puts the dimension-free complexity of computing a GAS point of a Lipschitz function in a situation similar to that of computing the volume of a convex body [20, 21], for which randomization yields strict improvement. It also reveals a fundamental hurdle in modern large-scale nonconvex nonsmooth optimization.

Now, suppose that we drop the dimension-free requirement and allow any deterministic algorithm with finite complexity (potentially with exponential dependence on the dimension). As our second main result, we show in Theorem 2 that for any sufficiently small $\varepsilon \geq 0, \delta \geq 0$:

> *Any deterministic finite-time algorithm for computing (ε, δ)-GAS points of Lipschitz functions cannot be general zero-respecting.*

The notion of a *general zero-respecting* algorithm (see Section 2.3) generalizes that of a *zero-respecting* algorithm in the smooth setting [9, Section 2.2] to the nonsmooth setting. Informally, a (general) zero-respecting algorithm never explores the coordinates along which the function is in some sense locally "flat." Moreover, it captures the classic notion of *linear span* algorithm [42, Assumption 2.1.4] as a special case. The above result rules out any natural derandomization of the algorithm by Zhang et al. [49].

The major obstacle in lower bounding the oracle complexity of GAS is the lack of hardness source. In the smooth setting, almost all the hard constructions [9, 10, 22] are built upon what Nesterov called "the worst-function in the world" [42, Chapter 2.1.2]. However, these constructions fail to rule out (ε, δ)-GAS points when the number of oracle calls is $\omega\big(\log(1/\delta)\big)$. In the nonsmooth case, simple resisting oracle-type constructions [47, 49] would not rule out (ε, δ)-GAS points if there exist $i \in [T]$, $j \in [T]$ such that $0 < \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\| < \delta$. Another source of hardness called "string guessing" [4, 5] is popular in the online and nonsmooth convex settings. With a modified "string guessing" function, Kornowski and Shamir [33] proved that the oracle complexity of any randomized/deterministic algorithm for computing (ε, δ)-NAS points cannot avoid an exponential dependence on the dimension. However, these constructions would also be inapplicable to the computation of (ε, δ)-GAS points when the number of oracle calls is $\omega\big(\log(1/\delta)\big)$.

Our main technical contribution is a new resisting oracle-type, wedge-shaped hard construction that is tailored for deterministic computation of GAS points. On a high level, given only the local information about a function at the queried points, none of the algorithms we consider can distinguish between a single-coordinate resisting construction similar to [47, 49] or our "wedge" construction. While there may be many GAS points of the former among the queried points, with careful design and analysis, we can eliminate all GAS points below certain precision of the latter near the queried points. As a result, no algorithm can identify a GAS point of all Lipschitz functions. We remark that our hardness results hold even under the very strong assumption that the local oracle returns (generalized) derivatives of all orders (if exist).

1.2 Related Work.

Asymptotic Analysis. Clarke stationary points $\boldsymbol{x}$ of quite general functions f (i.e., $\boldsymbol{x}$ satisfies $\mathbf{0} \in \partial f(\boldsymbol{x})$) are computable in the asymptotic regime. Benaïm et al. [2], Majewski et al. [36], Davis et al. [17] studied the asymptotic convergence of subgradient-type methods from a differential inclusion perspective. In particular, Davis et al. [17] established the asymptotic convergence of the subgradient method to a Clarke stationary point when applied to a Whitney stratifiable objective function. Daniilidis and Drusvyatskiy [15] constructed a pathological Lipschitz function for which the vanilla subgradient method may not converge even in continuous time. A discussion of the relationship between our hardness results and these asymptotic convergence results can be found in Remark 4.

Nonasymptotic Analysis. The nonasymptotic analysis of iterative methods for general nonconvex nonsmooth optimization problems is still in its infancy stage. For the concept of NAS, on the positive side, it is shown in [19, 16] that in the case of ρ-weakly convex functions, an (ε, δ)-NAS point is computable with $O(\mathrm{poly}(\rho, \varepsilon^{-1}, \delta^{-1}))$ oracle calls. On the negative side, Kornowski and Shamir [33] showed that neither deterministic nor randomized algorithms can compute NAS points of Lipschitz functions without having an oracle complexity that is exponential in the dimension, thus establishing the intractability of NAS. Tian and So [45] extended the hardness results in [33] to cover ρ-weakly convex functions by proving an $\Omega(\log(\rho))$ lower bound on the oracle complexity. For the concept of GAS, the conventional gradient sampling scheme [7, 29, 30, 6] cannot promise dimension-free finite-time computation. Metel and Takeda [37] introduced a perturbed SGD method and established its nonasymptotic convergence under mild assumptions. However, the complexity is not dimension-free. Recently, Zhang et al. [49] have presented a randomized algorithm with dimension-free oracle complexity for computing a GAS point. However, the algorithm in [49] requires a non-standard, impractical subgradient oracle. Such an oracle can be replaced by a standard one by introducing extra randomized procedures; see Tian et al. [46] and Davis et al. [18].

Derandomization of [49]. Several recent concurrent works, including Kornowski and Shamir [32], Jordan et al. [27], and Kong and Lewis [31], investigate the deterministic computation of GAS points in various settings.[4] On the negative side, both [32, 27] present hardness results for the deterministic computation of GAS points. While their constructions are different from that we will present in Section 4, their hardness results cannot achieve the same level of generality as our Theorems 1 and 2. On the positive side, both [32, 27] show that derandomization is possible when f is a β-smooth function. In particular, they introduced algorithms with $\widetilde{O}(\log(\beta)\delta^{-1}\varepsilon^{-3})$ complexity for computing

[4] We became aware of these concurrent, independent developments when a preliminary version of our manuscript was being reviewed for possible publication in the proceedings of a conference.

an (ε, δ)-GAS point of such an f. The work [27] then further develops a "white-box" deterministic smoothing technique. By contrast, the work [31] presents a deterministic algorithm that is applicable to more general difference-of-convex, piecewise linear, and weakly convex functions. Furthermore, it establishes a dimension-free finite-time complexity of the algorithm up to a natural *nonconvexity modulus*. A detailed discussion of these concurrent, independent results appears in Section 3.2.

Notation. Throughout this paper, scalars, vectors, and matrices are denoted by lowercase letters, boldface lower case letters, and boldface uppercase letters, respectively. The notation used in this paper is mostly standard: x_i denotes the i-th coordinate of $\boldsymbol{x}$; $A \otimes B := \{(a, b) : a \in A, b \in B\}$ denotes the Cartesian product of two sets A and B with $A^{\otimes k} := A \otimes \cdots \otimes A$ (k times); $\mathbb{B}_\varepsilon(\boldsymbol{x}) := \{\boldsymbol{v} : \|\boldsymbol{v} - \boldsymbol{x}\| \leq \varepsilon\}$ with $\mathbb{B} := \mathbb{B}_1(\mathbf{0})$ (we may write $\mathbb{B}_\varepsilon^d(\boldsymbol{x})$ to emphasize the dimension); $[\boldsymbol{x}, \boldsymbol{y}] := \{\gamma \boldsymbol{x} + (1-\gamma)\boldsymbol{y} : \gamma \in [0, 1]\}$; $\operatorname{dist}(\boldsymbol{x}, S) := \inf_{\boldsymbol{v} \in S} \|\boldsymbol{v} - \boldsymbol{x}\|$ for a non-empty closed set S; $\operatorname{conv}(S)$, $\operatorname{int}(S)$, $\operatorname{bd}(S)$, and S^c denote the convex hull, interior, boundary, and complement of the set S, respectively; $\operatorname{supp}(\boldsymbol{x}) := \{i : x_i \neq 0\}$; $\left\{\boldsymbol{x}^{A[f],(t)}\right\}_t$ denotes the sequence of iterates generated by the algorithm A when applied to the function f (we may write $\{\boldsymbol{x}^{(t)}\}_t$ when A and f are clear from the context); $\boldsymbol{e}_i$ denotes the i-th column of the identity matrix; $a \vee b := \max\{a, b\}$; $a \wedge b := \min\{a, b\}$; $\mathbb{N}_+ := \mathbb{N} \backslash \{0\}$; $[a] := \{1, \ldots, a\}$ for any integer $a \geq 1$; $\|f\|_{\mathrm{Lip}}$ denotes the Lipschitz constant of the function f.

Organization. We introduce the necessary background on variational analysis, definitions of approximate stationarity concepts, and the formal setting of our results in Section 2. Then, in Section 3, we present our main hardness results. The hard constructions and proofs are collected in Section 4. We conclude the paper in Section 5.

2 Preliminaries

2.1 Generalized Differentiation Theory

We begin with the following classic construction of generalized subdifferential for a Lipschitz function [44, Theorem 9.61]:

Definition 1 (Clarke subdifferential) The Clarke subdifferential of a Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$ at the point $\boldsymbol{x} \in \mathbb{R}^d$ is defined by

$$\partial f(\boldsymbol{x}) := \operatorname{conv}\left(\left\{\boldsymbol{s} : \exists \boldsymbol{x}_n \to \boldsymbol{x}, \nabla f(\boldsymbol{x}_n) \text{ exists}, \nabla f(\boldsymbol{x}_n) \to \boldsymbol{s}\right\}\right).$$

Perturbation and approximation are powerful principles underlying the theory of and many algorithms for optimization. The following δ-approximation of the Clarke subdifferential introduced by Goldstein [23, Definition 2.2] has nice limiting behavior (see Fact 1) and is convenient for algorithmic developments.

Definition 2 (Goldstein δ-subdifferential) Given a constant $\delta \geq 0$, the Goldstein δ-subdifferential of a Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$ at the point $\boldsymbol{x} \in \mathbb{R}^d$ is defined by

$$\partial_\delta f(\boldsymbol{x}) := \operatorname{conv}\big(\bigcup_{\boldsymbol{y}\in\mathbb{B}_\delta(\boldsymbol{x})} \partial f(\boldsymbol{y})\big).$$

Some useful properties of the Clarke subdifferential and its Goldstein approximation are collected below:

Fact 1 (cf. Clarke [12], Goldstein [23]) *For a Lipschitz function f,*

- *$\partial f(\boldsymbol{x}), \partial_\delta f(\boldsymbol{x})$ are nonempty, convex, compact for any $\delta > 0$;*
- *$\partial f(\boldsymbol{x}) = \cap_{\delta>0} \partial_\delta f(\boldsymbol{x})$;*
- *if f is continuously differentiable at $\boldsymbol{x}$, then $\partial f(\boldsymbol{x}) = \{\nabla f(\boldsymbol{x})\}$;*
- *if f is convex, then ∂f is equal to the convex subdifferential.*

2.2 Approximate Stationarity Concepts

We are now ready to provide the definitions of two important approximate stationarity concepts; i.e., GAS [23, 49, 46] and NAS [16, 19].

Definition 3 (Goldstein approximate stationarity, GAS) Given a Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$, we say that $\boldsymbol{x} \in \mathbb{R}^d$ is an (ε, δ)-GAS point of f if

$$\operatorname{dist}\Big(\mathbf{0}, \partial_\delta f(\boldsymbol{x})\Big) \leq \varepsilon.$$

Definition 4 (Near-approximate stationarity, NAS) Given a Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$, we say that $\boldsymbol{x} \in \mathbb{R}^d$ is an (ε, δ)-NAS point of f if

$$\operatorname{dist}\Big(\mathbf{0}, \bigcup_{\boldsymbol{y}\in\mathbb{B}_\delta(\boldsymbol{x})} \partial f(\boldsymbol{y})\Big) \leq \varepsilon.$$

It is easy to see that if $\boldsymbol{x}$ is an NAS point, then it is also a GAS point since $\partial_\delta f(\boldsymbol{x}) \supseteq \cup_{\boldsymbol{y}\in\mathbb{B}_\delta(\boldsymbol{x})} \partial f(\boldsymbol{y})$. However, the converse does not hold in general, even for convex [46, Proposition 2.7] and continuously differentiable functions [33, Proposition 2]. Besides, Kornowski and Shamir [33] proved that the oracle complexity of any randomized/deterministic algorithm for computing NAS points of Lipschitz functions cannot avoid an exponential dependence on the dimension, which implies the intractability of NAS.

Now, let us record the following simple result, which concerns the existence of an NAS point (thus, also a GAS point) for a Lipschitz function.

Proposition 1 (Existence) *Let $f : \mathbb{R}^d \to \mathbb{R}$ be Lipschitz with $f(\mathbf{0}) - \inf_{\boldsymbol{z}} f(\boldsymbol{z}) \leq \Delta < +\infty$. Then, for any $\varepsilon > 0$, there exists an $\boldsymbol{x} \in \mathbb{R}^d$ such that*

$$\operatorname{dist}\Big(\mathbf{0}, \partial f(\boldsymbol{x})\Big) \leq \varepsilon \quad \textit{and} \quad \|\boldsymbol{x}\| \leq \frac{\Delta}{\varepsilon}.$$

Proof This follows from the variational principle for subgradient [44, Proposition 10.44] and [44, Theorem 8.49]. □

2.3 Settings

We formally define the oracle, algorithm class, and function class used in our technical development.

Local Oracle. Given a function $f : \mathbb{R}^d \to \mathbb{R}$, a local oracle $\mathcal{O}_f$ is a map that, when queried at a point $\boldsymbol{x} \in \mathbb{R}^d$, returns a function $g : \mathbb{R}^d \to \mathbb{R}$ satisfying

$$f(\boldsymbol{y}) = g(\boldsymbol{y}), \qquad \forall \boldsymbol{y} \in \mathbb{B}_\nu^d(\boldsymbol{x})$$

for some $\nu > 0$.

Remark 1 *A subtle but crucial point here is that the local oracle $\mathcal{O}_f$ only returns the function g but not the radius ν. Otherwise, the resisting oracle argument in Section 4.1 would fail when an algorithm queries $\boldsymbol{x}^{(t+1)} \in \mathbb{B}_{\nu(\boldsymbol{x}^{(t)})}\left(\boldsymbol{x}^{(t)}\right)$. Nevertheless, the local oracle is still a very powerful notion. If f is smooth at $\boldsymbol{x}$, then $\mathcal{O}_f(\boldsymbol{x})$ is capable of providing information about (if exist) $f(\boldsymbol{x})$ and $\nabla^p f(\boldsymbol{x})$, for all $p \in \mathbb{N}_+$. If f is nonsmooth at $\boldsymbol{x}$, then $\mathcal{O}_f(\boldsymbol{x})$ is capable of providing information about (if exist) the Clarke subdifferential [44, Theorem 9.61], Fréchet subdifferential [44, Exercise 8.4], limiting subdifferential [44, Theorem 8.3(b)], and even the impractical subgradient selection oracle in [49, Assumption 1(a)] and [31, Oracle 2.5]. We note here that assuming such a (unreasonably) strong oracle would only strengthen our hardness results, as the algorithms are allowed to use more information.*

Algorithm Class. We consider algorithms from the classes $\mathcal{A}_{\text{det}}$ and $\mathcal{A}_{\text{det-gzr}}$, which are defined as follows:

- $\mathcal{A}_{\text{det}}$: Algorithms that use information from the local oracle $\mathcal{O}_f$ at the queried points deterministically. Formally, for any algorithm $A \in \mathcal{A}_{\text{det}}$, integer $T \in \mathbb{N}_+$, and Lipschitz functions $f, h : \mathbb{R}^d \to \mathbb{R}$, if $\boldsymbol{x}^{A[f],(1)} = \boldsymbol{x}^{A[h],(1)}$ and there exists a constant $\nu > 0$ such that $\mathcal{O}_f\left(\boldsymbol{x}^{A[f],(i)}\right)(\boldsymbol{y}) = \mathcal{O}_h\left(\boldsymbol{x}^{A[f],(i)}\right)(\boldsymbol{y})$, $\boldsymbol{y} \in \mathbb{B}_\nu\left(\boldsymbol{x}^{A[f],(i)}\right)$ for all $i \in [T]$, then $\boldsymbol{x}^{A[f],(j)} = \boldsymbol{x}^{A[h],(j)}$ for all $j \in [T+1]$.
- $\mathcal{A}_{\text{det-gzr}}$: Algorithms that are deterministic and *general zero-respecting*; i.e., coordinates along which the function is locally constant are never explored. Formally, every algorithm $A \in \mathcal{A}_{\text{det-gzr}}$, when applied to any Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$ and initialized at the origin, satisfies

$$\operatorname{supp}\left(\boldsymbol{x}^{(t)}\right) \subseteq \bigcup_{i<t} \left\{ j \in [d] : \forall \nu > 0, \exists \theta \in \mathbb{R}, \boldsymbol{y} \in \mathbb{B}_\nu\left(\boldsymbol{x}^{(i)}\right) \text{ s.t. } \boldsymbol{y} + \theta \boldsymbol{e}_j \in \mathbb{B}_\nu\left(\boldsymbol{x}^{(i)}\right), f(\boldsymbol{y}) \neq f(\boldsymbol{y} + \theta \boldsymbol{e}_j) \right\}$$

  for all $t \geq 2$.

Remark 2 *The oracle complexity of any deterministic algorithm interacting with a p^{th}-order oracle $\left(f(\boldsymbol{x}), \nabla f(\boldsymbol{x}), \ldots, \nabla^p f(\boldsymbol{x})\right)$ in the smooth setting [9] is lower bounded by that of any algorithm in $\mathcal{A}_{\text{det}}$ interacting with a local oracle. The class of general zero-respecting algorithms is a nonsmooth generalization of that of zero-respecting algorithms [9, Section 2.2], which we recall informally as follows. An algorithm A is called p^{th}-*order zero-respecting*, *denoted by $A \in \mathcal{A}_{\text{zr}}^{(p)}$, if it, when applied to any function f with well-defined $\nabla^p f$ and initialized at the origin, satisfies*

$$\operatorname{supp}\left(\boldsymbol{x}^{A[f],(t)}\right) \subseteq \bigcup_{q\in[p]} \bigcup_{i<t} \operatorname{supp}\left\{\nabla^q f(\boldsymbol{x}^{A[f],(i)})\right\}$$

for all $t \geq 2$ (we refer the reader to [9, Section 2.2] for the definition of supp *when applied to a high-order tensor and for further details). With a slight abuse of notation, let us denote the class of deterministic p^{th}-order zero-respecting algorithms by $\mathcal{A}_{\text{det}} \cap \mathcal{A}_{\text{zr}}^{(p)}$. We note that even for infinitely differentiable functions, the algorithm class $\mathcal{A}_{\text{det-gzr}}$ is strictly more general than $\mathcal{A}_{\text{det}} \cap \mathcal{A}_{\text{zr}}^{(\infty)}$. Indeed, consider the function*

$$f(x) = \begin{cases} e^{-1/x^2} & \text{if } x \neq 0, \\ 0 & \text{if } x = 0. \end{cases}$$

It is easy to check that $f(0) = 0$ and $\nabla^p f(0) = 0$ for all $p \in \mathbb{N}_+$. Then, for any algorithm $A \in \mathcal{A}_{\text{det}} \cap \mathcal{A}_{\text{zr}}^{(\infty)}$, we must have $x^{A[f],(t)} = 0$ for all $t \in \mathbb{N}_+$. On the contrary, as $f(x) \neq 0 = f(0)$ when $x \neq 0$, for any $A' \in \mathcal{A}_{\text{det-gzr}}$, the query point $x^{A'[f],(t)}$ can be arbitrary for $t \geq 2$. Formally, for any $T \geq 2$, we have

$$\left\{\left\{x^{A[f],(t)}\right\}_{t\in[T]} : A \in \mathcal{A}_{\text{det}} \cap \mathcal{A}_{\text{zr}}^{(\infty)}\right\} \subsetneq \left\{\left\{x^{A'[f],(t)}\right\}_{t\in[T]} : A' \in \mathcal{A}_{\text{det-gzr}}\right\}.$$

The class of zero-respecting algorithms contains most of the oracle-based methods in smooth optimization [9], such as gradient descent, Nesterov's accelerated gradient descent [40], conjugate gradient [24], BFGS and L-BFGS [35], Newton, cubic-regularized Newton [43], and trust-region [13] methods. It also contains the widely used class of linear span algorithms [42, Assumption 2.1.4] as a special case. Besides, the algorithms developed in the recent works [32, 27, 31] are all general zero-respecting.

Function Class. For any given constant $C > 0$ and dimension $d \in \mathbb{N}_+$, we consider the function class

$$\mathcal{F}_{C,d}^{\text{Lip}} := \left\{f : \mathbb{R}^{d'} \to \mathbb{R} : d' \in [d], \|f\|_{\text{Lip}} \leq C, f(\mathbf{0}) - \inf_{\boldsymbol{x}} f(\boldsymbol{x}) \leq C\right\}.$$

As an immediate illustration of the above definitions, let us record the following result, which shows that there is no deterministic finite-time algorithm for testing whether a point is GAS for a Lipschitz function.

Proposition 2 (Testing) *Suppose that $0 \leq \varepsilon, \delta < 1$. For any $A \in \mathcal{A}_{\text{det}}$ and $T \in \mathbb{N}_+$, there exists a 4-Lipschitz function $f : \mathbb{R} \to \mathbb{R}$ and a point $y \in \mathbb{R}$ such that the algorithm A cannot decide whether y is an (ε, δ)-GAS point of f within T oracle calls.*

Proof Consider the function $\mathbb{R} \ni x \mapsto f_1(x) = x \in \mathbb{R}$ and the point $y = 0$. Suppose that A makes the queries $\{x^{(t)}\}_{t=1}^T$ to the local oracle and returns the correct answer; i.e., $\operatorname{dist}(0, \partial_\delta f_1(0)) > \varepsilon$. Then, we only need to establish the existence of a function f_2 that is equal to f_1 in a neighborhood of $x^{(t)}$ for any $t \in [T]$ but satisfies $\operatorname{dist}(0, \partial_\delta f_2(0)) \leq \varepsilon$. The construction of f_2 is easy and similar to [41, Section 5] but not identical. It is clear that there exists a line segment $[a, b] \subseteq \mathbb{B}_\delta(0)$ satisfying $x^{(t)} \notin [a, b], \forall t \in [T]$. Let $f_2 : \mathbb{R} \to \mathbb{R}$ be given by

$$f_2(x) = \min\left\{x, a + 4\left|x - \frac{a+b}{2}\right|\right\}.$$

Then, it can be verified that $\|f_2\|_{\text{Lip}} \leq 4$ and $0 \in \partial_\delta f_2(0)$, as required. □

3 Deterministic Inapproximability of Stationarity Concepts

We report the main results of this paper in Section 3.1. Then, we discuss some recent works on deterministic computation of GAS points for various function classes in Section 3.2.

3.1 Main Results

For the general deterministic setting, we have the following hardness result:

Theorem 1 (Deterministic) *Suppose that $0 \leq \varepsilon < \frac{1}{2\sqrt{17}}$, $0 \leq \delta < \frac{1}{2}$, and $C \geq 6$. For any $T \in \mathbb{N}_+$ and $d \geq T + 1$, we have*

$$\inf_{A \in \mathcal{A}_{\text{det}}} \sup_{f \in \mathcal{F}_{C,d}^{\text{Lip}}} \min_{t \in [T]} \operatorname{dist}\left(\mathbf{0}, \partial_\delta f\left(\boldsymbol{x}^{A[f],(t)}\right)\right) > \varepsilon.$$

Theorem 1 shows that any deterministic algorithm for computing GAS points of Lipschitz functions must have an oracle complexity that is at least linear in the dimension. In particular, no such algorithm has dimension-free finite-time complexity. Contrasting this with the results in [46, 18], we see that randomization provably helps in the dimension-free computation of GAS points.

Without the dimension-free requirement, we have the following finite-time hardness result:

Theorem 2 (Deterministic general zero-respecting) *Suppose that $0 \leq \varepsilon < \frac{1}{2\sqrt{17}}$, $0 \leq \delta < \frac{1}{2}$, and $C \geq 6$. For any $T \in \mathbb{N}_+$ and $d \geq 2$, we have*

$$\inf_{A \in \mathcal{A}_{\text{det-gzr}}} \sup_{f \in \mathcal{F}_{C,d}^{\text{Lip}}} \min_{t \in [T]} \operatorname{dist}\left(\mathbf{0}, \partial_\delta f\left(\boldsymbol{x}^{A[f],(t)}\right)\right) > \varepsilon.$$

As far as computing GAS points of Lipschitz functions is concerned, while Theorem 1 does not rule out the existence of a deterministic algorithm with finite-time complexity (in fact, we shall see one such algorithm shortly), Theorem 2 shows that no deterministic general zero-respecting algorithm has finite-time complexity. Thus, Theorem 2 suggests that any deterministic finite-time algorithm for computing GAS points of Lipschitz functions must be quite different from most of the commonly used algorithmic schemes in smooth optimization.

Remark 3 *Our hard constructions for establishing both Theorems 1 and 2 are Lipschitz continuous piecewise linear functions. Therefore, the results in Section 3 still hold even if we restrict the functions in $\mathcal{F}_{C,d}^{\mathrm{Lip}}$ to be also piecewise linear. We note that such functions form a subclass of difference-of-convex functions [14, Proposition 4.4.3], semi-algebraic functions, and functions exactly representable by ReLU neural networks [1, Theorem 2.1].*

Remark 4 *As noted in Remark 3, the hard construction in the proof of Theorem 2 is semi-algebraic. In the recent work [17], Davis et al. showed that when applied to a semi-algebraic function f, every limit point of the vanilla subgradient method is a Clarke stationary point of f; i.e., $\limsup_{t\to+\infty}\{\boldsymbol{x}^{(t)}\} \subseteq \{\boldsymbol{x} : \mathbf{0} \in \partial f(\boldsymbol{x})\}$, where the limit supremum is defined in the set-theoretic sense (see [44, Definition 4.1]). By passing to a convergent subsequence of $\{\boldsymbol{x}^{(t)}\}_t$ if necessary, it is evident that for any $\delta > 0$, there exists a $T \in \mathbb{N}_+$ such that $\boldsymbol{x}^{(T)} \in \mathbb{B}_\delta\left(\boldsymbol{x}^{(\infty)}\right)$. Then, by definition, the point $\boldsymbol{x}^{(T)}$ is $(0,\delta)$-GAS for f. This seems to contradict Theorem 2, as the vanilla subgradient method is clearly deterministic and general zero-respecting. The subtlety here is that Theorem 2 rules out any $A \in \mathcal{A}_{\text{det-gzr}}$ with* a priori *finite-time complexity; i.e., a finite T that works uniformly for all $f \in \mathcal{F}_{C,d}^{\mathrm{Lip}}$. While the result in [17] implies that for any semi-algebraic function f, there exists a $T \in \mathbb{N}_+$ such that the point $\boldsymbol{x}^{(T)}$ generated by the vanilla subgradient method is $(0,\delta)$-GAS for f, Theorem 2 shows that any* a priori *bound on T is impossible.*

As mentioned above, there exists a finite-time algorithm in the class $\mathcal{A}_{\text{det}}$ for computing GAS points of Lipschitz functions. Of course, in view of Theorem 1, the oracle complexity of any such algorithm will necessarily depend on the dimension. To demonstrate the existence, recall from Proposition 1 that given a function $f : \mathbb{R}^d \to \mathbb{R}$ with $f \in \mathcal{F}_{C,d}^{\mathrm{Lip}}$, there exists an $(\varepsilon, 0)$-GAS point of f in the ball $\mathbb{B}_{C/\varepsilon}(\mathbf{0})$. Thus, by querying the oracle at the points on a δ-net $\mathcal{N}$ of the ball $\mathbb{B}_{C/\varepsilon}(\mathbf{0})$ and noting that $|\mathcal{N}| \le \left\lceil \left(1 + \frac{2C}{\varepsilon\delta}\right)^d \right\rceil$, we immediately obtain the following result:

Proposition 3 (Finite-time algorithm) *Suppose that $\varepsilon > 0$, $\delta > 0$, $C > 0$, and $d \in \mathbb{N}_+$. There exists an $A \in \mathcal{A}_{\text{det}}$ such that for $T = \left\lceil \left(1 + \frac{2C}{\varepsilon\delta}\right)^d \right\rceil$, we have*

$$\sup_{f\in\mathcal{F}_{C,d}^{\mathrm{Lip}}} \min_{t\in[T]} \operatorname{dist}\Big(\mathbf{0}, \bigcup_{\boldsymbol{y}\in\mathbb{B}_\delta\left(\boldsymbol{x}^{A[f],(t)}\right)} \partial f(\boldsymbol{y})\Big) \le \varepsilon.$$

Proposition 3 indicates that for any fixed dimension d, we can compute an (ε,δ)-GAS point of a function in $\mathcal{F}_{C,d}^{\mathrm{Lip}}$ using only $\mathrm{poly}(\varepsilon^{-1}, \delta^{-1}, C)$ oracle calls.

3.2 Discussion

On Lower Bounds. By considering the more restrictive first-order oracle (i.e., $\mathcal{O}_f$ maps the query point $\boldsymbol{x}$ to the pair $(f(\boldsymbol{x}), \partial f(\boldsymbol{x}))$), the works [32, 27] establish a similar $\Omega(d)$ lower bound on the oracle complexity of algorithms in the class $\mathcal{A}_{\text{det}}$ as our Theorem 1. However, their constructions do not apply to algorithms that use higher-order information. Specifically, if we consider a p^{th}-order oracle with $p \geq 3$, then these constructions cannot be used to establish dimension-free hardness for deterministic algorithms and finite-time hardness for algorithms in $\mathcal{A}_{\text{det}} \cap \mathcal{A}_{\text{zr}}^{(p)}$. Indeed, when localized within the ball $\mathbb{B}_r\left(\boldsymbol{x}^{(t)}\right)$, both constructions in [32, 27] reduce to

$$g_{\boldsymbol{x}^{(t)}}(\boldsymbol{x}) = \frac{\|\boldsymbol{x} - \boldsymbol{x}^{(t)}\|^2}{r^2} \cdot \boldsymbol{v}^\top \boldsymbol{x} + \left(1 - \frac{\|\boldsymbol{x} - \boldsymbol{x}^{(t)}\|^2}{r^2}\right) \cdot \boldsymbol{e}_1^\top \left(\boldsymbol{x} - \boldsymbol{x}^{(t)}\right),$$

where $r > 0$ is a sufficiently small constant and $\boldsymbol{v} \in \mathbb{R}^d$ is a "hidden direction" that should not be recognized by the algorithm. However, it is easy to see that with the information of $\nabla^3 g_{\boldsymbol{x}^{(t)}}\left(\boldsymbol{x}^{(t)}\right)$, the hidden $\boldsymbol{v}$ can be computed deterministically using one single oracle query. Thus, a simple third-order algorithm would invalidate these constructions. By contrast, our hardness results in Theorems 1 and 2 hold for the more general local oracle. It seems non-trivial to attain the same level of generality without using a construction similar in spirit to ours in Section 4.1.

On Upper Bounds. The works [32, 27, 31] introduce deterministic algorithms that compute GAS points for different function classes. It is worth pointing out that all these algorithms are general zero-respecting. Therefore, by Theorem 2, they cannot compute GAS points of Lipschitz functions using only a finite number of oracle queries. Nevertheless, these developments deepen our understanding of the computability and complexity of GAS for various function classes. While the works [32, 27] mainly study β-smooth functions, the work [31] focuses on C-Lipschitz functions with a finite *nonconvexity modulus* $\Lambda(\delta)$; see [31, Section 6] for the definition of $\Lambda(\delta)$. It is shown in [31, Theorem 6.6] that an (ε, δ)-GAS point of such a function can be computed deterministically with a dimension-free oracle complexity of

$$O\left(\frac{C^3 \Lambda(\delta)}{\varepsilon^4 \delta}\right).$$

This is an interesting result, as any piecewise linear function has a finite nonconvexity modulus $\Lambda(\delta)$. An easy corollary of our construction implies that the $O(\Lambda(\delta))$ dependence in the complexity is actually optimal. Indeed, for any fixed $T \in \mathbb{N}_+$, the hard function in the proof of Theorem 1 (and also of Theorem 2) is piecewise linear and has $5T + 2$ affine pieces. By [31, Corollary 5.8], we know that $\Lambda(\delta) = O(T)$ for our hard function. Combining Theorem 1 or Theorem 2 with [31, Theorem 6.6], we have the following conclusion:

Proposition 4 *The dependence of the complexity of any dimension-free finite-time algorithm in $\mathcal{A}_{\text{det}}$ or that of any finite-time algorithm in $\mathcal{A}_{\text{det-gzr}}$ on the nonconvexity modulus $\Lambda(\delta)$ is $\Theta(\Lambda(\delta))$; i.e., the dependence is optimal.*

4 Proofs

We now collect the proofs of the main results in Section 3. In what follows, we assume that $0 \leq \varepsilon < \frac{1}{2\sqrt{17}}$, $0 \leq \delta < \frac{1}{2}$, $C \geq 6$, and $d \geq 2$. Besides, we assume that every algorithm starts from $\mathbf{0}$; i.e., for any $A \in \mathcal{A}_{\text{det}}$ and $f \in \mathcal{F}_{C,d}^{\text{Lip}}$, we have $\boldsymbol{x}^{A[f],(1)} = \mathbf{0}$. Such an assumption is common in the literature [9, 33] and can be made without loss of generality as $f((\cdot) - \boldsymbol{x}^{(1)}) \in \mathcal{F}_{C,d}^{\text{Lip}}$ whenever $f : \mathbb{R}^d \to \mathbb{R}$ satisfies $\|f\|_{\text{Lip}} \leq C$ and $f(\boldsymbol{x}^{(1)}) - \inf_{\boldsymbol{x}} f(\boldsymbol{x}) \leq C$.

4.1 The Construction

4.1.1 Single-Coordinate Resisting Function

We first construct a resisting function using the classic resisting oracle argument [42, Chapter 1.1.3]. The construction is similar to those in [49, 47] and we repeat the argument here for completeness' sake. Let $T \in \mathbb{N}_+$ and $d \geq 2$ be fixed. Consider the resisting oracle $\mathcal{O}$, which is a local oracle in the sense defined in Section 2.3 and, when queried at the point $\bar{\boldsymbol{x}} \in \mathbb{R}^d$, returns the function $\mathcal{O}(\bar{\boldsymbol{x}}) : \mathbb{R}^d \to \mathbb{R}$ given by

$$\mathcal{O}\left(\bar{\boldsymbol{x}}\right)\left(\boldsymbol{x}\right) = x_1 - \bar{x}_1.$$

Given an algorithm $A \in \mathcal{A}_{\text{det}}$, let $\left\{\boldsymbol{x}^{A,(t)}\right\}_{t=1}^{T}$ be the sequence of iterates generated by A after T calls to $\mathcal{O}$. Now, we establish the existence of a resisting function $f_s \in \mathcal{F}_{1,d}^{\text{Lip}}$ that is compatible with the resisting oracle $\mathcal{O}$. Upon eliminating the duplicated values in the collection $\left\{x_1^{A,(t)}\right\}_{t=1}^{T}$ and arranging the rest in increasing order, we obtain a sequence $\left\{x_1^{(i)}\right\}_{i=1}^{T'}$ with $T' \leq T$ and $x_1^{(1)} < x_1^{(2)} < \cdots < x_1^{(T')}$. Let

$$\sigma := \min\left\{\min_{1 \leq i < j \leq T'} \left|x_1^{(i)} - x_1^{(j)}\right|, 1\right\} > 0$$

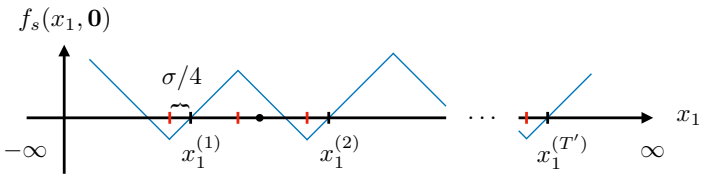

Fig. 1 The single-coordinate resisting function in (SC).

and define $f_s : \mathbb{R}^d \to \mathbb{R}$ by

$$f_s(\boldsymbol{x}) := \tag{SC}$$
$$\begin{cases} -x_1 + x_1^{(1)} - \frac{\sigma}{2} \text{ for } x_1 \in \left(-\infty, x_1^{(1)} - \frac{\sigma}{4}\right), \\ x_1 - x_1^{(t)} \text{ for } x_1 \in \bigcup_{t\in[T'-1]} \left[x_1^{(t)} - \frac{\sigma}{4}, \frac{1}{2}\left(x_1^{(t)} + x_1^{(t+1)}\right) - \frac{\sigma}{4}\right), \\ -x_1 + x_1^{(t+1)} - \frac{\sigma}{2} \text{ for } x_1 \in \bigcup_{t\in[T'-1]} \left[\frac{1}{2}\left(x_1^{(t)} + x_1^{(t+1)}\right) - \frac{\sigma}{4}, x_1^{(t+1)} - \frac{\sigma}{4}\right), \\ x_1 - x_1^{(T')} \text{ for } x_1 \in \left[x_1^{(T')} - \frac{\sigma}{4}, +\infty\right); \end{cases}$$

see Figure 1. It is easy to see that f_s is 1-Lipschitz and $f_s(\mathbf{0}) - \inf_{\boldsymbol{x}} f_s(\boldsymbol{x}) \le 1 \le C$. Moreover, we have the following result, which, together with the definition of $\left\{x_1^{(i)}\right\}_{i=1}^{T'}$, shows that f_s is compatible with the resisting oracle.

Lemma 1 *For any $t \in [T']$ and $\boldsymbol{x} \in \mathbb{B}_{\frac{\sigma}{8}}^1\left(x_1^{(t)}\right) \otimes \mathbb{R}^{d-1}$, we have $f_s(\boldsymbol{x}) = x_1 - x_1^{(t)}$.*

Proof For any such $\boldsymbol{x}$, we have $x_1^{(t)} - \frac{\sigma}{4} < x_1^{(t)} - \frac{\sigma}{8} \le x_1 \le x_1^{(t)} + \frac{\sigma}{8} < x_1^{(t)} + \frac{\sigma}{4} \le \frac{1}{2}\left(x_1^{(t)} + x_1^{(t+1)}\right) - \frac{\sigma}{4}$. □

The single-coordinate resisting function f_s defined in (SC) is not sufficient for establishing a lower bound on the oracle complexity of GAS. Indeed, suppose that for any $\delta > 0$ and $T \ge 2$, we query $\boldsymbol{x}^{(t)} = (t-1)\delta \cdot \boldsymbol{e}_1$ for any $t \in [T]$. Then, both $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ are $(0, \delta)$-GAS points of the generated f_s. The main difficulty here is that f_s zigzags too much, which results in many GAS points. We need a construction that is compatible with the resisting oracle without any GAS point near the queried points.

4.1.2 A "Wedge" Replacement

In this section, we build a resisting function with a wedge-like shape. Our main building block is the "wedge" function $h : \mathbb{R}^2 \to \mathbb{R}$ defined by

$$h(x, y) := \max\left\{\underbrace{y - \frac{\eta}{2}}_{①}, \tilde{h}(x, y)\right\},$$

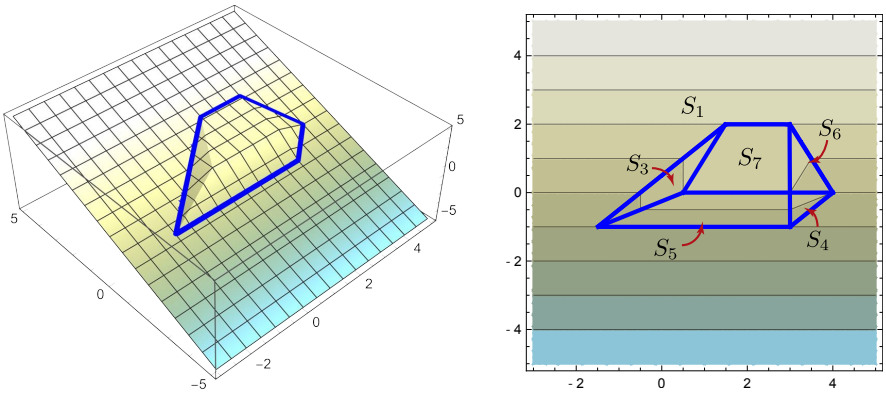

Fig. 2 The wedge-shaped resisting function in (PL).

where η satisfies $0 < \eta \le \frac{\sigma}{32}$ (recall that σ is defined above (SC) and pertains to the minimum gap between the first coordinates of different queried points) and the function $\tilde{h} : \mathbb{R}^2 \to \mathbb{R}$ is defined by

$$\tilde{h}(x,y) := \min\left\{\underbrace{x+\frac{\eta}{2}}_{②}, \underbrace{2y+\eta}_{③}, \underbrace{\frac{y}{2}+\eta}_{④}\right\} + \min\left\{\underbrace{-x+\frac{5\eta}{2}}_{⑤}, \underbrace{-\frac{\eta}{2}}_{⑥}\right\}. \tag{W}$$

The following piecewise linear representation of h is more convenient for analysis:

$$h(x,y) = \begin{cases} y-\frac{\eta}{2} \text{ for } (x,y) \in S_1 := \{(x,y) : h(x,y) = ①\}, \\ 3\eta \text{ for } (x,y) \in S_2 := \{(x,y) : h(x,y) = ② + ⑤\}, \\ x \text{ for } (x,y) \in S_3 := \{(x,y) : h(x,y) = ② + ⑥\}, \\ -x+2y+\frac{7}{2}\eta \text{ for } (x,y) \in S_4 := \{(x,y) : h(x,y) = ③ + ⑤\}, \\ 2y+\frac{1}{2}\eta \text{ for } (x,y) \in S_5 := \{(x,y) : h(x,y) = ③ + ⑥\}, \\ -x+\frac{y}{2}+\frac{7\eta}{2} \text{ for } (x,y) \in S_6 := \{(x,y) : h(x,y) = ④ + ⑤\}, \\ \frac{y}{2}+\frac{\eta}{2} \text{ for } (x,y) \in S_7 := \{(x,y) : h(x,y) = ④ + ⑥\}. \end{cases} \tag{PL}$$

Here are some elementary properties of the sets $\{S_i\}_{i=1}^7$.

Lemma 2 $S_2 = \emptyset$.

Proof Let us first examine

$$S_2 = \{(x,y) : ① \le (② \wedge ③ \wedge ④) + (⑤ \wedge ⑥), ② \le ③ \wedge ④, ⑤ \le ⑥\}.$$

Suppose that $(x,y) \in S_2$. By $⑤ \le ⑥$, we know that $x \ge 3\eta$. Due to $② \le ③ \wedge ④$, we get $x \le \min\{\frac{y}{2}, 2y\} + \frac{\eta}{2}$. Thus, we have $0 < 3\eta \le x \le \frac{y}{2} + \frac{\eta}{2}$, or equivalently, $\frac{y}{2} \ge \frac{5\eta}{2}$. Now, we compute

$$① - (② \wedge ③ \wedge ④) - (⑤ \wedge ⑥) \ge ① - ④ - ⑥ = \frac{y}{2} - \eta \ge \frac{3\eta}{2} > 0,$$

which gives a contradiction. □

Lemma 3 $S_3 = \left\{(x,y) : y - \frac{\eta}{2} \le x \le \min\left\{\frac{y}{2}, 2y\right\} + \frac{\eta}{2}\right\} \subseteq [-\frac{3}{2}\eta, \frac{3}{2}\eta] \times [-\eta, 2\eta]$.

Proof Note that

$$S_3 = \{(x,y) : ① \le ② + ⑥, ② \le ③ \wedge ④, ⑤ \ge ⑥\}.$$

By ② $\le$ ③$\wedge$④, we have $x \le \min\{\frac{y}{2}, 2y\} + \frac{\eta}{2}$. From ① $\le$ ②+⑥, we get $y - \frac{\eta}{2} \le x$. Then, we have $-\eta \le y \le 2\eta$ and $-\frac{3}{2}\eta \le x \le \frac{3}{2}\eta$. Thus, the constraint $x \le 3\eta$ in ⑤ $\ge$ ⑥ is always satisfied. □

Lemma 4 *If* $(x,y) \in S_3^c$, *then* $\partial h(x,y) \subseteq \begin{bmatrix} [-1,0] \\ [\frac{1}{2}, 2] \end{bmatrix}$.

Proof By Lemma 2 and the piecewise linear representation of h in (PL), we know that

$$\nabla h(x,y) \subseteq \begin{bmatrix} [-1,0] \\ [\frac{1}{2}, 2] \end{bmatrix}, \qquad \forall (x,y) \in \bigcup_{i \ne 3} \operatorname{int}(S_i).$$

It is easy but tedious to verify that $S_i \cap S_j$ has zero Lebesgue measure for any $i \ne j$ (see Figure 2), as they are the solution sets of systems of linear equations. Taking the convex hull of $\{\nabla h(x,y) : (x,y) \in \cup_{i \ne 3} \operatorname{int}(S_i)\}$, invoking [44, Theorem 9.61], and using $\operatorname{conv}(A \times B) = \operatorname{conv}(A) \times \operatorname{conv}(B)$, we obtain the desired result. □

Now, we proceed to the final construction. Let $\widetilde{h_w} : \mathbb{R}^2 \to \mathbb{R}$ be defined by

$$\widetilde{h_w}(x_1, x_2) := \max\left\{x_2 - \frac{\eta}{2}, \max_{t \in [T']} \tilde{h}\left(x_1 - x_1^{(t)}, x_2\right)\right\},$$

where $\tilde{h}$ is the function defined in (W) and $\left\{x_1^{(i)}\right\}_{i=1}^{T'}$ is the sorted sequence of first coordinates of the queried points after removing duplicates (see the paragraph above (SC)). Then, the final "wedge" hard construction $h_w : \mathbb{R}^d \to \mathbb{R}$ for $d \ge 2$ is defined by

$$h_w(\boldsymbol{x}) := \max\left\{-5, \widetilde{h_w}(x_1, x_2)\right\}.$$

Lemma 5 *For any* $t \in [T']$ *and* $(x_1, x_2) - (x_1^{(t)}, 0) \in \operatorname{int}(S_3)$, *we have*

$$x_1 - x_1^{(t)} = \tilde{h}\left(x_1 - x_1^{(t)}, x_2\right) > x_2 - \frac{\eta}{2} > \max_{t' \in [T'] \setminus \{t\}} \tilde{h}\left(x_1 - x_1^{(t')}, x_2\right).$$

Besides, we have $\mathbb{B}^2_{\frac{\eta}{8}} \subseteq \operatorname{int}(S_3)$.

Proof The first two relations follow directly from the piecewise linear representation of h in (PL) and Lemma 3. For the last strict inequality, suppose there exists a $t' \in [T']$, $x_1^{(t')} \ne x_1^{(t)}$ such that the opposite holds. Then, we have ① $\le$ (②$\wedge$③$\wedge$④)+(⑤$\wedge$⑥). From ① $\le$ (③$\wedge$④)+⑥, we see that $-\eta \le x_2 \le 2\eta$.

Since ① $\le$ (③ $\wedge$ ④) + ⑤, we get $x_1 - x_1^{(t')} \le \min\{x_2, -\frac{x_2}{2}\} + 4\eta \le 4\eta$. Moreover, since ① $\le$ ② + ⑥, we get $x_1 - x_1^{(t')} \ge x_2 - \frac{\eta}{2} \ge -\frac{3}{2}\eta$. It follows that $\left|x_1 - x_1^{(t')}\right| \le 4\eta$. However, by Lemma 3 and noting that $\sigma \ge 32\eta$, we have

$$\left|x_1 - x_1^{(t')}\right| \ge \left|x_1^{(t)} - x_1^{(t')}\right| - \left|x_1 - x_1^{(t)}\right| \ge \sigma - 3\eta > 30\eta,$$

which gives a contradiction. To establish $\mathbb{B}^2_{\frac{\eta}{8}} \subseteq \mathrm{int}(S_3)$, we note that for any $\boldsymbol{x} \in \mathbb{B}^2_{\frac{\eta}{8}}$,

$$x_2 - \frac{\eta}{2} \le -\frac{3\eta}{8} < -\frac{\eta}{8} \le x_1 \le \frac{\eta}{8} < \frac{\eta}{2} - \frac{\eta}{4} \le \min\left\{\frac{x_2}{2}, 2x_2\right\} + \frac{\eta}{2}.$$

This, together with Lemma 3, gives the desired result. □

We are now ready to prove the main lemma of this subsection.

Lemma 6 *The following hold:*

- *The function h_w is 3-Lipschitz and satisfies $h_w(\mathbf{0}) - \inf_{\boldsymbol{x}} h_w(\boldsymbol{x}) \le 6 \le C$.*
- *There exists a $\nu \in (0, \sigma/256]$ such that*

$$f_s(\boldsymbol{y}) = h_w(\boldsymbol{y}), \qquad \forall \boldsymbol{y} \in V := \bigcup_{t \in [T']} \mathbb{B}^2_\nu \left(\begin{bmatrix} x_1^{(t)} \\ 0 \end{bmatrix} \right) \otimes \mathbb{R}^{d-2}.$$

Proof From the piecewise linear representation of h in (PL), it is easy to see that h_w is 3-Lipschitz. Note that $h(x, 0) \le \max\{-\frac{\eta}{2}, \eta - \frac{\eta}{2}\} = \frac{\eta}{2} \le 1$ for any $x \in \mathbb{R}$. It follows that $h_w(\mathbf{0}) - \inf_{\boldsymbol{x}} h_w(\boldsymbol{x}) = \max\{-5, \widetilde{h_w}(0,0)\} + 5 \le 1 + 5 = 6$. Let $\nu = \frac{\eta}{8} \in (0, \frac{\sigma}{256}]$. Then, Lemma 5 implies that $\widetilde{h_w}(y_1, y_2) > y_2 - \frac{\eta}{2} \ge -\frac{5\eta}{8} > -5$ for all $\boldsymbol{y} \in V$. This, together with Lemma 1 and the fact that $\nu < \frac{\sigma}{8}$, shows that $f_s(\boldsymbol{y}) = \widetilde{h_w}(y_1, y_2) = h_w(\boldsymbol{y})$ for all $\boldsymbol{y} \in V$. □

4.1.3 Resolution of Approximate Stationarity

In this subsection, we prove that the function h_w has no GAS point below certain precision at which the function value is lower bounded by -1.

Lemma 7 *Let $\boldsymbol{x} \in \mathbb{R}^d$ be such that $h_w(\boldsymbol{x}) \ge -1$ and $0 \le \delta < 1/2$. Then, for any $\boldsymbol{y} \in \mathbb{B}^d_{2\delta}(\boldsymbol{x})$, we have $h_w(\boldsymbol{y}) = \widetilde{h_w}(y_1, y_2)$.*

Proof By Lemma 6 and the assumption that $0 \le \delta < \frac{1}{2}$, we know that for any $\boldsymbol{y} \in \mathbb{B}^d_{2\delta}(\boldsymbol{x})$,

$$h_w(\boldsymbol{y}) \ge h_w(\boldsymbol{x}) - |h_w(\boldsymbol{y}) - h_w(\boldsymbol{x})| \ge -1 - 6\delta \ge -4 > -5,$$

as required. □

The main result of this subsection is the following:

Lemma 8 *Let $\boldsymbol{x} \in \mathbb{R}^d$ be such that $h_w(\boldsymbol{x}) \geq -1$ and $0 \leq \delta < 1/2$. Then, we have* $\operatorname{dist}\Big(\mathbf{0}, \partial_\delta h_w(\boldsymbol{x})\Big) \geq \frac{1}{\sqrt{17}}$.

Proof Observe that

$$\begin{aligned}
\partial_\delta h_w(\boldsymbol{x}) &= \operatorname{conv}\left(\bigcup\nolimits_{\boldsymbol{y}\in\mathbb{B}_\delta^d(\boldsymbol{x})} \partial h_w(\boldsymbol{y})\right) && \text{(Definition 2)}\\
&= \operatorname{conv}\left(\bigcup\nolimits_{(y_1,y_2)\in\mathbb{B}_\delta^2\left((x_1,x_2)\right)} \partial\widetilde{h_w}(y_1,y_2)\otimes\{0\}^{\otimes d-2}\right) && \text{(Lemma 7)}\\
&= \operatorname{conv}\left(\left(\bigcup\nolimits_{(y_1,y_2)\in\mathbb{B}_\delta^2\left((x_1,x_2)\right)} \partial\widetilde{h_w}(y_1,y_2)\right)\otimes\{0\}^{\otimes d-2}\right)\\
& && \text{([3, §3, Exercise 3(4)])}\\
&= \operatorname{conv}\left(\bigcup\nolimits_{(y_1,y_2)\in\mathbb{B}_\delta^2\left((x_1,x_2)\right)} \partial\widetilde{h_w}(y_1,y_2)\right)\otimes\{0\}^{\otimes d-2}\\
&= \partial_\delta\widetilde{h_w}(x_1,x_2)\otimes\{0\}^{\otimes d-2}.
\end{aligned}$$

Therefore, it suffices to show that $\operatorname{dist}\Big(\mathbf{0}, \partial_\delta\widetilde{h_w}(x_1,x_2)\Big) \geq \frac{1}{\sqrt{17}}$. Let $\boldsymbol{g} := \arg\min_{\boldsymbol{z}\in\partial_\delta\widetilde{h_w}(x_1,x_2)} \|\boldsymbol{z}\|$. By Carathéodory's theorem [44, Theorem 2.29], we can write $\boldsymbol{g}$ as a finite convex combination

$$\boldsymbol{g} = \sum_{i=1}^{3} \lambda_i \boldsymbol{g}^i,$$

where $\boldsymbol{g}^i \in \partial\widetilde{h_w}(\boldsymbol{y}^i)$, $\boldsymbol{y}^i \in \mathbb{B}_\delta^2\big((x_1,x_2)\big)$, $\lambda_i \geq 0$ for $i \in \{1,2,3\}$ and $\sum_{j=1}^3 \lambda_j = 1$. Let

$$\begin{aligned}
P_1 &:= \left\{i \in \{1,2,3\} : \exists t \in [T'], \boldsymbol{y}^i - (x_1^{(t)}, 0) \in \operatorname{int}(S_3)\right\},\\
P_2 &:= \left\{i \in \{1,2,3\} : \forall t \in [T'], \boldsymbol{y}^i - (x_1^{(t)}, 0) \in S_3^c\right\},\\
P_3 &:= \left\{i \in \{1,2,3\} : \exists t \in [T'], \boldsymbol{y}^i - (x_1^{(t)}, 0) \in \operatorname{bd}(S_3)\right\}.
\end{aligned}$$

We claim that P_1, P_2, P_3 are mutually disjoint. By definition, P_2 and $P_1 \cup P_3$ are disjoint. Thus, it suffices to prove that P_1 and P_3 are disjoint. Suppose to the contrary that there exists an $i \in P_1 \cap P_3$. Then, there exist $t, t' \in [T']$ such that

$$\boldsymbol{y}^i - (x_1^{(t)}, 0) \in \operatorname{int}(S_3), \qquad \boldsymbol{y}^i - (x_1^{(t')}, 0) \in \operatorname{bd}(S_3).$$

As $\operatorname{int}(S_3)$ and $\operatorname{bd}(S_3)$ are disjoint, we must have $t \neq t'$. We compute

$$\begin{aligned}
\left|x_1^{(t)} - x_1^{(t')}\right| &= \left\|(x_1^{(t)}, 0) - (x_1^{(t')}, 0)\right\|_1\\
&\leq \left\|\boldsymbol{y}^i - (x_1^{(t)}, 0)\right\|_1 + \left\|\boldsymbol{y}^i - (x_1^{(t')}, 0)\right\|_1\\
&\leq 2\cdot(3\eta + 3\eta) = 12\eta. && \text{(Lemma 3)}
\end{aligned}$$

However, noting that $\sigma \geq 32\eta$, we get

$$\left|x_1^{(t)} - x_1^{(t')}\right| \geq \min_{1\leq i<j\leq T'} \left|x_1^{(i)} - x_1^{(j)}\right| \geq \sigma \geq 32\eta > 0,$$

which gives a contradiction.

Now, we rewrite $\boldsymbol{g}$ by averaging within P_i for $i = 1, 2, 3$; i.e.,

$$\boldsymbol{g} = \sum_{i\in P_1} \lambda_i \boldsymbol{g}^i + \sum_{j\in P_2} \lambda_j \boldsymbol{g}^j + \sum_{k\in P_3} \lambda_k \boldsymbol{g}^k = \sum_{i=1}^{3} \theta_i \boldsymbol{g}^{P_i},$$

where $\theta_i := \sum_{j\in P_i} \lambda_j$ and $\boldsymbol{g}^{P_i} := \sum_{j\in P_i} \frac{\lambda_j}{\theta_i} \boldsymbol{g}^j$ for $i \in \{1, 2, 3\}$. Consider the following cases:

- Averaging within P_1: $\boldsymbol{g}^{P_1} = \boldsymbol{e}_1$. To see this, note that for any $i \in P_1$, there exists a $t_i \in [T']$ such that $\boldsymbol{y}^i - (x_1^{(t_i)}, 0) \in \operatorname{int}(S_3)$. By Lemma 5, for any $i \in P_1$, we have
$$\{t_i\} = \operatorname*{arg\,max}_{t'\in[T']} \tilde{h}\left(y_1^i - x_1^{(t')}, y_2^i\right)$$
and $\tilde{h}\left(y_1^i - x_1^{(t_i)}, y_2^i\right) > y_2^i - \frac{\eta}{2}$. Thus, by [12, Proposition 2.3.12] and the piecewise linear representation of h in (PL), we get $\emptyset \neq \partial\widetilde{h_w}(\boldsymbol{y}^i) \subseteq \partial\tilde{h}\left(y_1^i - x_1^{(t_i)}, y_2^i\right) = \{\boldsymbol{e}_1\}$, which implies that $\boldsymbol{g}^i = \boldsymbol{e}_1$ for any $i \in P_1$.
- Averaging within P_2: $\boldsymbol{g}^{P_2} \in \begin{bmatrix}[-1,0]\\ [\frac{1}{2},2]\end{bmatrix}$ by Lemma 4.
- Averaging within P_3: $\boldsymbol{g}^{P_3} \in \operatorname{conv}\left(\boldsymbol{e}_1, \begin{bmatrix}[-1,0]\\ [\frac{1}{2},2]\end{bmatrix}\right)$ by [44, Theorem 9.61].

From the above, we conclude that

$$\boldsymbol{g} = \sum_{i=1}^{3} \theta_i \boldsymbol{g}^{P_i} \in \operatorname{conv}\left(\boldsymbol{e}_1, \begin{bmatrix}[-1,0]\\ [\frac{1}{2},2]\end{bmatrix}\right).$$

To complete the proof, it remains to estimate $\operatorname{dist}(\mathbf{0}, \partial h_w(\boldsymbol{x})) = \|\boldsymbol{g}\|$. The following lemma will be useful for this purpose.

Lemma 9 *We have*

$$\frac{1}{17} = \min_{t,v_1,v_2} (t+(1-t)v_1)^2+(1-t)^2v_2^2 \quad \text{s.t.} \quad t \in [0,1], v_1 \in [-1,0], v_2 \in [1/2,2].$$

Proof Let q be the objective function of and (t^*, v_1^*, v_2^*) be an optimal solution to the above optimization problem. It is easy to see that $v_2^* = \frac{1}{2}$. Now, note that

$$q(t, v_1, v_2^*) = \left(\frac{1}{4} + (v_1 - 1)^2\right) \cdot t^2 - 2\left(v_1 - \frac{1}{2}\right)^2 \cdot t + v_1^2 + \frac{1}{4}.$$

By the first-order optimality condition and the constraint $v_1 \in [-1, 0]$, we have

$$0 < t^* = \frac{(2v_1 - 1)^2}{1 + 4(v_1 - 1)^2} = 1 - \frac{4(1 - v_1)}{1 + 4(1 - v_1)^2} < 1.$$

This implies that

$$q(t^*, v_1, v_2^*) = \frac{1}{1 + 4(v_1 - 1)^2} \geq \frac{1}{17},$$

and equality holds when $v_1 = v_1^* = -1$. □

Armed with Lemma 9, we compute

$$\begin{aligned}
\|\boldsymbol{g}\| &\geq \operatorname{dist}\left(\mathbf{0}, \operatorname{conv}\left(\boldsymbol{e}_1, \begin{bmatrix} [-1, 0] \\ [\frac{1}{2}, 2] \end{bmatrix}\right)\right) \\
&= \min_{t \in [0,1]} \left\| \begin{bmatrix} t + (1-t) \cdot [-1, 0] \\ (1-t) \cdot [\frac{1}{2}, 2] \end{bmatrix} \right\| \\
&= \min_{\substack{t:0 \leq t \leq 1, \\ v_1: -1 \leq v_1 \leq 0, \\ v_2: \frac{1}{2} \leq v_2 \leq 2}} \sqrt{(t + (1-t)v_1)^2 + (1-t)^2 v_2^2} \\
&= \frac{1}{\sqrt{17}}. \qquad \text{(by Lemma 9)}
\end{aligned}$$

It follows that $\operatorname{dist}\big(\mathbf{0}, \partial_\delta h_w(\boldsymbol{x})\big) = \operatorname{dist}\left(\mathbf{0}, \partial_\delta \widetilde{h_w}(x_1, x_2)\right) = \|\boldsymbol{g}\| \geq \frac{1}{\sqrt{17}}$, as required. □

4.2 Hardness Results

In this subsection, we put everything together. We will first prove Theorem 2, as its proof is conceptually easier and can be reused in that of Theorem 1.

4.2.1 Deterministic General Zero-Respecting Algorithms

Proof (of Theorem 2) Fix any $T \in \mathbb{N}_+$, $d \geq 2$, and $A \in \mathcal{A}_{\text{det-gzr}}$. By applying the resisting oracle argument in Section 4.1.1 to A, we obtain a resisting function $f_s : \mathbb{R}^d \to \mathbb{R}$ and a sequence $\left\{\boldsymbol{x}^{A[f_s],(t)}\right\}_{t=1}^T$ with $\boldsymbol{x}^{A[f_s],(1)} = \mathbf{0}$. By Lemma 1 and the definition of $\mathcal{A}_{\text{det-gzr}}$, with a simple induction on t, we have $2 \notin \operatorname{supp}\left(\boldsymbol{x}^{A[f_s],(t)}\right)$ for all $t \in [T]$. Furthermore, by Lemma 6, we know that the algorithm A cannot distinguish between the resisting function f_s and its associated "wedge" function h_w by querying the local oracle at the points $\left\{\boldsymbol{x}^{A[f_s],(t)}\right\}_{t=1}^T$. Formally, there exists a $\nu > 0$ such that $\mathcal{O}_{f_s}\left(\boldsymbol{x}^{A[f_s],(t)}\right)(\boldsymbol{y}) = \mathcal{O}_{h_w}\left(\boldsymbol{x}^{A[f_s],(t)}\right)(\boldsymbol{y})$ for all $\boldsymbol{y} \in \mathbb{B}_\nu^d\left(\boldsymbol{x}^{A[f_s],(t)}\right)$ and $t \in [T]$. Since A is deterministic, we have $\boldsymbol{x}^{A[f_s],(t)} = \boldsymbol{x}^{A[h_w],(t)}$ for all $t \in [T]$, which

implies that $h_w\left(\boldsymbol{x}^{A[h_w],(t)}\right) = f_s\left(\boldsymbol{x}^{A[f_s],(t)}\right) = 0 > -1$ for all $t \in [T]$. Thus, by Lemma 8 and the assumption that $0 \leq \varepsilon < \frac{1}{2\sqrt{17}}$, we get

$$\min_{t\in[T]} \operatorname{dist}\Big(\mathbf{0}, \partial_\delta h_w\left(\boldsymbol{x}^{A[h_w],(t)}\right)\Big) \geq \frac{1}{\sqrt{17}} > \varepsilon + \frac{1}{2\sqrt{17}}.$$

Upon noting that $h_w \in \mathcal{F}_{C,d}^{\mathrm{Lip}}$ by Lemma 6, the proof is complete. □

4.2.2 Deterministic Algorithms

The case of general deterministic algorithms can be reduced to that of deterministic general zero-respecting algorithms using a classic adversarial rotation argument [39, 9, 48].

Proof (of Theorem 1) Fix any $T \in \mathbb{N}_+$, $d \geq T+1$, and $A \in \mathcal{A}_{\mathrm{det}}$. By applying the resisting oracle argument in Section 4.1.1 to A, we obtain a resisting function $f_s : \mathbb{R}^d \to \mathbb{R}$ and a sequence $\left\{\boldsymbol{x}^{A[f_s],(t)}\right\}_{t=1}^T$ with $\boldsymbol{x}^{A[f_s],(1)} = \mathbf{0}$. Let $\boldsymbol{V} := \left[\boldsymbol{e}_1 \middle| \boldsymbol{x}^{A[f_s],(2)} \middle| \cdots \middle| \boldsymbol{x}^{A[f_s],(T)}\right] \in \mathbb{R}^{d\times T}$. Furthermore, let $\boldsymbol{u}_2 \in \mathbb{R}^d$ be such that $\boldsymbol{V}^\top \boldsymbol{u}_2 = \mathbf{0}$ and $\|\boldsymbol{u}_2\| = 1$. Note that such an $\boldsymbol{u}_2$ exists because $d > T$. By choosing $\widetilde{\boldsymbol{U}} \in \mathbb{R}^{d\times(d-2)}$ to be an orthonormal basis of the orthogonal complement of $\operatorname{span}\{\boldsymbol{e}_1, \boldsymbol{u}_2\}$, we can define an orthogonal matrix $\boldsymbol{U} := \left[\boldsymbol{e}_1 \middle| \boldsymbol{u}_2 \middle| \widetilde{\boldsymbol{U}}\right] \in \mathbb{R}^{d\times d}$.

Now, let h_w be the "wedge" function associated with the resisting function f_s and $g_w(\boldsymbol{x}) := h_w(\boldsymbol{U}^\top \boldsymbol{x})$. We claim that there exists a $\nu \in (0, \sigma/256]$ such that $\mathcal{O}_{f_s}\left(\boldsymbol{x}^{A[f_s],(t)}\right)(\boldsymbol{y}) = \mathcal{O}_{g_w}\left(\boldsymbol{x}^{A[f_s],(t)}\right)(\boldsymbol{y})$ for all $\boldsymbol{y} \in \mathbb{B}_\nu^d\left(\boldsymbol{x}^{A[f_s],(t)}\right)$ and $t \in [T]$. To see this, let ν be the constant in Lemma 6. Fix $t \in [T]$ and $\boldsymbol{y} \in \mathbb{B}_\nu^d\left(\boldsymbol{x}^{A[f_s],(t)}\right)$. By Lemma 1, we know that $f_s(\boldsymbol{y}) = y_1 - x_1^{A[f_s],(t)}$. Observe that $\boldsymbol{U}^\top \boldsymbol{y} \in \mathbb{B}_\nu^d\left(\boldsymbol{U}^\top \boldsymbol{x}^{A[f_s],(t)}\right)$, as $\left\|\boldsymbol{U}^\top\left(\boldsymbol{y} - \boldsymbol{x}^{A[f_s],(t)}\right)\right\| \leq \nu$. Moreover, we have $2 \notin \operatorname{supp}\left(\boldsymbol{U}^\top \boldsymbol{x}^{A[f_s],(t)}\right)$ for all $t \in [T]$ by the construction of $\boldsymbol{U}$. It follows from Lemma 6 that $h_w\left(\boldsymbol{U}^\top \boldsymbol{y}\right) = f_s\left(\boldsymbol{U}^\top \boldsymbol{y}\right)$. Since $\boldsymbol{U}^\top \boldsymbol{y} \in \mathbb{B}_\nu^1\left(x_1^{A[f_s],(t)}\right) \otimes \mathbb{R}^{d-1}$, using Lemma 1 again yields

$$g_w(\boldsymbol{y}) = h_w(\boldsymbol{U}^\top \boldsymbol{y}) = f_s(\boldsymbol{U}^\top \boldsymbol{y}) = \boldsymbol{e}_1^\top \boldsymbol{y} - \boldsymbol{e}_1^\top \boldsymbol{x}^{A[f_s],(t)} = y_1 - x_1^{A[f_s],(t)} = f_s(\boldsymbol{y}).$$

Thus, we see that A cannot distinguish between f_s and g_w by querying the local oracle at $\left\{\boldsymbol{x}^{A[f_s],(t)}\right\}_{t=1}^T$, thus establishing the claim. In particular, since A is deterministic, we have $\boldsymbol{x}^{A[f_s],(t)} = \boldsymbol{x}^{A[g_w],(t)}$ for all $t \in [T]$.

Next, observe that

$$\begin{aligned}
\partial_\delta g_w(\boldsymbol{x}) &= \operatorname{conv}\left(\bigcup_{\boldsymbol{y}\in\mathbb{B}_\delta^d(\boldsymbol{x})} \boldsymbol{U}\partial h_w\left(\boldsymbol{U}^\top \boldsymbol{y}\right)\right) \\
&= \boldsymbol{U}\operatorname{conv}\left(\bigcup_{\boldsymbol{y}\in\mathbb{B}_\delta^d(\boldsymbol{x})} \partial h_w\left(\boldsymbol{U}^\top \boldsymbol{y}\right)\right) \\
&= \boldsymbol{U}\operatorname{conv}\left(\bigcup_{\boldsymbol{z}\in\mathbb{B}_\delta^d(\boldsymbol{U}^\top\boldsymbol{x})} \partial h_w(\boldsymbol{z})\right) = \boldsymbol{U}\partial_\delta h_w\left(\boldsymbol{U}^\top \boldsymbol{x}\right),
\end{aligned}$$

where the first equality follows from [44, Theorem 8.49, Exercise 10.7] (see also [12, Theorem 2.3.10]); the second can be deduced from the bijectivity of $\boldsymbol{U}$ [38, Chapter 1, §2, Exercise 2(b)] and [25, Chapter A, Proposition 1.3.4]; and the third is due to $\left\{\boldsymbol{U}^\top \boldsymbol{y} : \boldsymbol{y} \in \mathbb{B}_\nu^d(\boldsymbol{x})\right\} = \mathbb{B}_\nu^d\left(\boldsymbol{U}^\top \boldsymbol{x}\right)$. Besides, for any $t \in [T]$, we have $h_w\left(\boldsymbol{U}^\top \boldsymbol{x}^{A[g_w],(t)}\right) = f_s\left(\boldsymbol{x}^{A[f_s],(t)}\right) = 0 > -1$. By Lemma 8 and the assumption that $0 \leq \varepsilon < \frac{1}{2\sqrt{17}}$, we obtain

$$\begin{aligned}\min_{t\in[T]} \operatorname{dist}\Big(\mathbf{0}, \partial_\delta g_w\left(\boldsymbol{x}^{A[g_w],(t)}\right)\Big) &= \min_{t\in[T]} \operatorname{dist}\Big(\mathbf{0}, \partial_\delta h_w\left(\boldsymbol{U}^\top \boldsymbol{x}^{A[g_w],(t)}\right)\Big) \\ &\geq \frac{1}{\sqrt{17}} > \varepsilon + \frac{1}{2\sqrt{17}}.\end{aligned}$$

Moreover, as a simple corollary of Lemma 6, we have $g_w \in \mathcal{F}_{C,d}^{\mathrm{Lip}}$. This completes the proof. □

5 Concluding Remarks

In this paper, we showed that no deterministic algorithm for computing GAS points of Lipschitz functions has dimension-free finite-time complexity. This settles an open question posed by Zhang et al. in [49]. Furthermore, even without the dimension-free requirement, we showed that any finite-time deterministic method cannot be general zero-respecting. In particular, this implies that any natural derandomization of the algorithms in [49, 46, 18] cannot have finite-time complexity. Our results shed light on the hardness of nonconvex nonsmooth optimization problems in modern large-scale settings. As for further research, it would be interesting to study the complexity of approximate stationarity concepts based on other subdifferential constructions for different classes of structured nonconvex nonsmooth optimization problems.

References

1. Arora, R., Basu, A., Mianjy, P., Mukherjee, A.: Understanding deep neural networks with rectified linear units. In: International Conference on Learning Representations (2018)
2. Benaïm, M., Hofbauer, J., Sorin, S.: Stochastic approximations and differential inclusions. SIAM Journal on Control and Optimization **44**(1), 328–348 (2005)
3. Blyth, T.S.: Set Theory and Abstract Algebra. Longman Publishing Group (1975)
4. Böckenhauer, H.J., Hromkovič, J., Komm, D., Krug, S., Smula, J., Sprock, A.: The string guessing problem as a method to prove lower bounds on the advice complexity. Theoretical Computer Science **554**, 95–108 (2014)
5. Braun, G., Guzmán, C., Pokutta, S.: Lower bounds on the oracle complexity of nonsmooth convex optimization via information theory. IEEE Transactions on Information Theory **63**(7), 4709–4724 (2017)

6. Burke, J.V., Curtis, F.E., Lewis, A.S., Overton, M.L., Simões, L.E.: Gradient sampling methods for nonsmooth optimization. Numerical Nonsmooth Optimization: State of the Art Algorithms pp. 201–225 (2020)
7. Burke, J.V., Lewis, A.S., Overton, M.L.: A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. SIAM Journal on Optimization **15**(3), 751–779 (2005)
8. Carmon, Y., Duchi, J.C., Hinder, O., Sidford, A.: "Convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. In: International Conference on Machine Learning, pp. 654–663 (2017)
9. Carmon, Y., Duchi, J.C., Hinder, O., Sidford, A.: Lower bounds for finding stationary points I. Mathematical Programming **184**(1–2), 71–120 (2020)
10. Carmon, Y., Duchi, J.C., Hinder, O., Sidford, A.: Lower bounds for finding stationary points II: First-order methods. Mathematical Programming **185**(1), 315–355 (2021)
11. Cartis, C., Gould, N.I.M., Toint, Ph.L.: On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization problems. SIAM Journal on Optimization **20**(6), 2833–2852 (2010)
12. Clarke, F.H.: Optimization and Nonsmooth Analysis. SIAM (1990)
13. Conn, A.R., Gould, N.I., Toint, P.L.: Trust Region Methods. SIAM (2000)
14. Cui, Y., Pang, J.S.: Modern Nonconvex Nondifferentiable Optimization. SIAM (2021)
15. Daniilidis, A., Drusvyatskiy, D.: Pathological subgradient dynamics. SIAM Journal on Optimization **30**(2), 1327–1338 (2020)
16. Davis, D., Drusvyatskiy, D.: Stochastic model-based minimization of weakly convex functions. SIAM Journal on Optimization **29**(1), 207–239 (2019)
17. Davis, D., Drusvyatskiy, D., Kakade, S., Lee, J.D.: Stochastic subgradient method converges on tame functions. Foundations of Computational Mathematics **20**(1), 119–154 (2020)
18. Davis, D., Drusvyatskiy, D., Lee, Y.T., Padmanabhan, S., Ye, G.: A gradient sampling method with complexity guarantees for Lipschitz functions in high and low dimensions. In: Advances in Neural Information Processing Systems, vol. 35, pp. 6692–6703 (2022)
19. Davis, D., Grimmer, B.: Proximally guided stochastic subgradient method for nonsmooth, nonconvex problems. SIAM Journal on Optimization **29**(3), 1908–1930 (2019)
20. Dyer, M., Frieze, A.: Computing the volume of convex bodies: A case where randomness provably helps. Probabilistic Combinatorics and Its Applications **44**, 123–170 (1991)
21. Dyer, M., Frieze, A., Kannan, R.: A random polynomial-time algorithm for approximating the volume of convex bodies. Journal of the ACM **38**(1), 1–17 (1991)
22. Ghadimi, S., Lan, G.: Stochastic first-and zeroth-order methods for nonconvex stochastic programming. SIAM Journal on Optimization **23**(4),

2341–2368 (2013)
23. Goldstein, A.: Optimization of Lipschitz continuous functions. Mathematical Programming **13**(1), 14–22 (1977)
24. Hager, W.W., Zhang, H.: A survey of nonlinear conjugate gradient methods. Pacific Journal of Optimization **2**(1), 35–58 (2006)
25. Hiriart-Urruty, J.B., Lemaréchal, C.: Fundamentals of Convex Analysis. Springer Science & Business Media (2004)
26. Jin, C., Netrapalli, P., Ge, R., Kakade, S.M., Jordan, M.I.: On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. Journal of the ACM **68**(2) (2021)
27. Jordan, M., Kornowski, G., Lin, T., Shamir, O., Zampetakis, M.: Deterministic nonsmooth nonconvex optimization. In: Conference on Learning Theory, pp. 4570–4597 (2023)
28. Kakade, S.M., Lee, J.D.: Provably correct automatic subdifferentiation for qualified programs. In: Advances in Neural Information Processing Systems, p. 7125–7135 (2018)
29. Kiwiel, K.C.: Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization. SIAM Journal on Optimization **18**(2), 379–388 (2007)
30. Kiwiel, K.C.: A nonderivative version of the gradient sampling algorithm for nonsmooth nonconvex optimization. SIAM Journal on Optimization **20**(4), 1983–1994 (2010)
31. Kong, S., Lewis, A.: The cost of nonconvexity in deterministic nonsmooth optimization. arXiv preprint arXiv:2210.00652 (2022)
32. Kornowski, G., Shamir, O.: On the complexity of finding small subgradients in nonsmooth optimization. arXiv preprint arXiv:2209.10346 (2022)
33. Kornowski, G., Shamir, O.: Oracle complexity in nonsmooth nonconvex optimization. Journal of Machine Learning Research **23**(314), 1–44 (2022)
34. Li, J., So, A.M.C., Ma, W.K.: Understanding notions of stationarity in nonsmooth optimization: A guided tour of various constructions of subdifferential for nonsmooth functions. IEEE Signal Processing Magazine **37**(5), 18–31 (2020)
35. Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Mathematical Programming **45**(1), 503–528 (1989)
36. Majewski, S., Miasojedow, B., Moulines, E.: Analysis of nonsmooth stochastic approximation: The differential inclusion approach. arXiv preprint arXiv:1805.01916 (2018)
37. Metel, M.R., Takeda, A.: Perturbed iterate SGD for Lipschitz continuous loss functions. Journal of Optimization Theory and Applications pp. 1–44 (2022)
38. Munkres, J.R.: Topology: New International Edition. Pearson Prentice Hall (2000)
39. Nemirovskij, A.S., Yudin, D.B.: Problem Complexity and Method Efficiency in Optimization. Wiley-Interscience (1983)
40. Nesterov, Yu.: A method of solving a convex programming problem with convergence rate $O(1/k^2)$. In: Doklady Akademii Nauk, vol. 269, pp. 543–

547. Russian Academy of Sciences (1983)
41. Nesterov, Yu.: How to make the gradients small. Optima (88), 10–11 (2012)
42. Nesterov, Yu.: Lectures on Convex Optimization, vol. 137. Springer (2018)
43. Nesterov, Yu., Polyak, B.T.: Cubic regularization of Newton method and its global performance. Mathematical Programming **108**(1), 177–205 (2006)
44. Rockafellar, R.T., Wets, R.J.B.: Variational Analysis, vol. 317. Springer Science & Business Media (2009)
45. Tian, L., So, A.M.C.: On the hardness of computing near-approximate stationary points of Clarke regular nonsmooth nonconvex problems and certain DC programs. ICML Workshop on Beyond First-Order Methods in ML Systems (2021)
46. Tian, L., Zhou, K., So, A.M.C.: On the finite-time complexity and practical computation of approximate stationarity concepts of Lipschitz functions. In: International Conference on Machine Learning, vol. 162, pp. 21360–21379. PMLR (2022)
47. Vavasis, S.A.: Black-box complexity of local minimization. SIAM Journal on Optimization **3**(1), 60–80 (1993)
48. Woodworth, B., Srebro, N.: Tight complexity bounds for optimizing composite objectives. In: Advances in Neural Information Processing Systems, vol. 29, pp. 3646–3654 (2016)
49. Zhang, J., Lin, H., Jegelka, S., Jadbabaie, A., Sra, S.: Complexity of finding stationary points of nonsmooth nonconvex functions. In: International Conference on Machine Learning, pp. 11173–11182 (2020)