Deterministic Nonsmooth Nonconvex Optimization

Michael I. Jordan* Guy Kornowski‡ Tianyi Lin* Ohad Shamir‡ Manolis Zampetakis*

*University of California, Berkeley
‡Weizmann Institute of Science

February 17, 2023

Abstract

We study the complexity of optimizing nonsmooth nonconvex Lipschitz functions by producing (δ, ϵ)-Goldstein stationary points. Several recent works have presented randomized algorithms that produce such points using $\widetilde{O}(\delta^{-1}\epsilon^{-3})$ first-order oracle calls, independent of the dimension d. It has been an open problem as to whether a similar result can be obtained via a deterministic algorithm. We resolve this open problem, showing that randomization is necessary to obtain a dimension-free rate. In particular, we prove a lower bound of $\Omega(d)$ for any deterministic algorithm. Moreover, we show that unlike smooth or convex optimization, access to function values is required for any deterministic algorithm to halt within any finite time horizon.

On the other hand, we prove that if the function is even slightly smooth, then the dimension-free rate of $\widetilde{O}(\delta^{-1}\epsilon^{-3})$ can be obtained by a deterministic algorithm with merely a logarithmic dependence on the smoothness parameter. Motivated by these findings, we turn to study the complexity of deterministically smoothing Lipschitz functions. Though there are well-known efficient black-box randomized smoothings, we start by showing that no such deterministic procedure can smooth functions in a meaningful manner (suitably defined), resolving an open question in the literature. We then bypass this impossibility result for the structured case of ReLU neural networks. To that end, in a practical "white-box" setting in which the optimizer is granted access to the network's architecture, we propose a simple, dimension-free, deterministic smoothing of ReLU networks that provably preserves (δ, ϵ)-Goldstein stationary points. Our method applies to a variety of architectures of arbitrary depth, including ResNets and ConvNets. Combined with our algorithm for slightly-smooth functions, this yields the first deterministic, dimension-free algorithm for optimizing ReLU networks, circumventing our lower bound.

1 Introduction

We consider the problem of optimizing a Lipschitz continuous function, $f : \mathbb{R}^d \to \mathbb{R}$, which is potentially not smooth nor convex, using a first-order algorithm which utilizes values and derivatives of the function at various points. The theoretical analysis of nonsmooth and nonconvex optimization has long been a focus of research in economics, control theory and computer science [Clarke, 1990, Mäkelä and Neittaanmäki, 1992, Outrata et al., 1998]. In recent years, this area has received renewed attention stemming from the fact that essentially all optimization problems associated with training modern neural networks are neither smooth nor convex, due to their depth and the ubiquitous use of rectified linear units (ReLUs), among other nonsmooth components [Nair and Hinton, 2010, Glorot et al., 2011].

Since the minimization of a Lipschitz function f is well known to be intractable [Nemirovski and Yudin, 1983, Murty and Kabadi, 1987, Nesterov, 2018], a local measure of optimality is required in order to obtain any reasonable guarantees. Accordingly, it is common to make use of the generalized gradient $\partial f(x)$ due to Clarke [1974, 1975, 1981], which is a natural generalization of the gradient and the convex subgradient [Clarke et al., 2008, Rockafellar and Wets, 2009, Burke et al., 2020], and seek points with small subgradient. Although under certain regularity assumptions it is possible to *asymptotically* find an approximate Clarke stationary point of f,[1] the standard subgradient method fails to approach a Clarke stationary point of a Lipschitz function in general [Daniilidis and Drusvyatskiy, 2020]; moreover, it is not possible to find such points using any algorithm within finite time [Zhang et al., 2020, Theorem 1]. Moreover, even getting *near* an approximate Clarke stationary point of a Lipschitz function has been proven to be impossible unless the number of queries has an exponential dependence on the dimension [Kornowski and Shamir, 2021]. For an overview of relevant theoretical results in nonsmooth nonconvex optimization, we refer to Appendix A.

These negative results motivate rethinking the definition of local optimality in terms of a relaxed yet still meaningful notion. To this end, we consider the problem of finding a (δ, ϵ)-Goldstein stationary point of f [Goldstein, 1977], which are points for which there exists a convex combination of gradients in a δ-neighborhood whose norm is less than ϵ (see Section 2 for a formal definition). The breakthrough result of Zhang et al. [2020] proposed a randomized algorithm that finds such points with a dimension-free complexity of $\widetilde{O}(\delta^{-1}\epsilon^{-3})$ oracle calls. Though they make use of a non-standard first-order oracle that does not apply to all Lipschitz functions, subsequent work [Davis et al., 2022, Tian et al., 2022] has proposed variants of the algorithm that apply to any Lipschitz function using a standard first-order oracle.

It is important to note that all of the aforementioned algorithms are randomized. This state of affairs is unusual when contrasted with the regimes of smooth or convex optimization, where deterministic optimal dimension-free first-order algorithms exist and cannot be improved upon by randomized algorithms [Nesterov, 2018, Carmon et al., 2021]. This raises a fundamental question:

What is the role of randomization in dimension-free nonsmooth nonconvex optimization?

1.1 Our Contributions

This paper presents several results on the complexity of finding (δ, ϵ)-Goldstein stationary points using deterministic algorithms, providing a detailed answer to the question raised above. Our contributions can be summarized as follows:

1. **Necessity of randomness for dimension-free complexity (Theorem 3.1).** We show that deterministic algorithms cannot find (δ, ϵ)-Goldstein stationary points at *any* dimension-free rate, by proving a dimension-dependent lower bound of $\Omega(d)$ for any deterministic first-order algorithm, where $\delta, \epsilon > 0$ are smaller than given constants.

2. **Deterministic algorithms require a zeroth-order oracle (Theorem 3.2).** In sharp contrast to smooth or convex optimization, we prove that without access to function values, no deterministic algorithm can guarantee to return a (δ, ϵ)-Goldstein stationary point within *any* finite time, whenever $\delta, \epsilon > 0$ are smaller than given constants. On the other hand, we note that a gradient oracle is sufficient to obtain a finite-time guarantee using a randomized algorithm (Remark 3.1).

[1] Namely, $\mathbf{x} \in \mathbb{R}^d$ such that $\min\{\|\mathbf{g}\| : \mathbf{g} \in \partial f(\mathbf{x})\} \leq \epsilon$.

3. **Deterministic algorithm with logarithmic smoothness dependence (Theorem 4.1).** Considering cases in which the objective function is slightly smooth, we present a deterministic first-order algorithm that finds a (δ, ϵ)-Goldstein stationary point of any H-smooth function within $\widetilde{O}(\log(H)\delta^{-1}\epsilon^{-3})$ oracle calls.[2]

4. **Deterministic smoothing (Theorem 5.1 and Theorem 5.2).** We show that unlike randomized black-box smoothings, no deterministic black-box smoothing can produce a reasonable $\mathrm{poly}(d)$-smooth approximation using a dimension-free complexity, essentially solving an open question due to Kornowski and Shamir [2021]. On the other hand, in a practical white-box model of ReLU neural networks, we propose a simple, dimension-free, deterministic smoothing procedure which applies to a variety of architectures, while provably maintaining the set of (δ, ϵ)-Goldstein stationary points. Combined with the algorithm described in the previous bullet, we obtain the first deterministic, dimension-free algorithm for optimizing ReLU networks, circumventing our aforementioned lower bound.

Related work. Following an initial publication of our results, Tian and So [2022] have independently presented an alternative proof of our first result (Theorem 3.1). A more detailed account of previous results in nonsmooth nonconvex optimization is deferred to Appendix A.

2 Preliminaries and Technical Background

Notation. We denote $[d] := \{1, 2, \ldots, d\}$. We denote by $\mathbf{0}_d \in \mathbb{R}^d$ the zero vector and by $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_d \in \mathbb{R}^d$ the standard basis vectors. For any vector $\mathbf{x} \in \mathbb{R}^d$, we let $\|\mathbf{x}\|$ be its Euclidean norm, and denote by x_i its i^{th} coordinate. For a set $\mathcal{X} \subseteq \mathbb{R}^d$, we let $\mathrm{conv}(\mathcal{X})$ denote its convex hull. For a continuous function $f(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}$, we let $\nabla f(\mathbf{x})$ denote the gradient of f at $\mathbf{x}$ (if it exists). For a scalar $a \in \mathbb{R}$, we let $\lfloor a \rfloor$ and $\lceil a \rceil$ be the smallest integer that is larger than a and the largest integer that is smaller than a. In addition, we denote a closed ball of radius $r > 0$ around a point $\mathbf{x} \in \mathbb{R}^d$ by $B_r(\mathbf{x}) := \{\mathbf{y} \in \mathbb{R}^d : \|\mathbf{y} - \mathbf{x}\| \leq r\}$. Given a bounded segment $I \subset \mathbb{R}$, we denote by $\xi \sim U(I)$ a random variable distributed uniformly over I. Finally, we use the standard big-O notation, with $O(\cdot)$, $\Theta(\cdot)$ and $\Omega(\cdot)$ hiding absolute constants that do not depend on problem parameters, $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ hiding absolute constants and additional logarithmic factors, and also denote by $\mathrm{poly}(\cdot)$ polynomial factors.

Nonsmooth analysis. We call a function $f : \mathbb{R}^d \to \mathbb{R}$ L-Lipschitz if for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d : |f(\mathbf{x}) - f(\mathbf{y})| \leq L\|\mathbf{x} - \mathbf{y}\|$, and H-smooth if it is differentiable and $\nabla f : \mathbb{R}^d \to \mathbb{R}^d$ is H-Lipschitz, namely for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d : \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq H\|\mathbf{x} - \mathbf{y}\|$. By Rademacher's theorem, Lipschitz functions are differentiable almost everywhere (in the sense of Lebesgue). Hence, for any Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$ and point $\mathbf{x} \in \mathbb{R}^d$ the Clarke subgradient set [Clarke, 1990] can be defined as

$$\partial f(\mathbf{x}) := \mathrm{conv}\{\mathbf{g} : \mathbf{g} = \lim_{n \to \infty} \nabla f(\mathbf{x}_n),\ \mathbf{x}_n \to \mathbf{x}\} \ ,$$

namely, the convex hull of all limit points of $\nabla f(\mathbf{x}_n)$ over all sequences of differentiable points which converge to $\mathbf{x}$. Note that if the function is continuously differentiable at a point or convex, the Clarke subdifferential reduces to the gradient or subgradient in the convex analytic sense, respectively. We say

[2] A function $f : \mathbb{R}^d \to \mathbb{R}$ is called H-smooth if for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d : \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{x})\| \leq H\|\mathbf{x} - \mathbf{y}\|$.

that a point $\mathbf{x}$ is an ϵ-Clarke stationary point of $f(\cdot)$ if $\min\{\|\mathbf{g}\| : \mathbf{g} \in \partial f(\mathbf{x})\} \leq \epsilon$. Furthermore, given $\delta > 0$ the Goldstein δ-subdifferential [Goldstein, 1977] of f at $\mathbf{x}$ is the set

$$\partial_\delta f(\mathbf{x}) := \mathrm{conv}\left(\cup_{\mathbf{y} \in B_\delta(\mathbf{x})} \partial f(\mathbf{y})\right) \ ,$$

namely all convex combinations of gradients at points in a δ-neighborhood of $\mathbf{x}$. We say that a point $\mathbf{x}$ is a (δ, ϵ)-Goldstein stationary point of $f(\cdot)$ if

$$\min\{\|\mathbf{g}\| : \mathbf{g} \in \partial_\delta f(\mathbf{x})\} \leq \epsilon \ .$$

Note that a point is ϵ-Clarke stationary if and only if it is (δ, ϵ)-Goldstein stationary for all $\delta > 0$ [Zhang et al., 2020, Lemma 7].

Algorithms and complexity. Throughout this work we consider iterative first-order algorithms, from an oracle complexity perspective [Nemirovski and Yudin, 1983]. Such an algorithm first produces an initial point $\mathbf{x}_0 \in \mathbb{R}^d$ (possibly at random, if it is a randomized algorithm) and receives $(f(\mathbf{x}_0), \partial f(\mathbf{x}_0))$.[3] Then, for any $t \geq 1$ produces $\mathbf{x}_t$ possibly at random based on previously observed responses, and receives $(f(\mathbf{x}_t), \partial f(\mathbf{x}_t))$. We are interested in the minimal number T for which we can guarantee to produce some (δ, ϵ)-Goldstein stationary point, uniformly over the class of Lipschitz functions.

3 Lower bounds for deterministic algorithms

3.1 Dimension-dependent lower bound

As discussed earlier, [Zhang et al., 2020, Davis et al., 2022, Tian et al., 2022] have presented randomized first-order algorithms that given any L-Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$ and an initial point $\mathbf{x}_0$ that satisfies $f(\mathbf{x}_0) - \inf_{\mathbf{x}} f(\mathbf{x}) \leq \Delta$, produce a (δ, ϵ)-Goldstein stationary point of f within $\widetilde{O}(\Delta L^2/\delta\epsilon^3)$ oracle calls to f. We show that this rate, and indeed any dimension-free rate, cannot be achieved by a deterministic algorithm.

Theorem 3.1 *For any $\Delta, L > 0$, $d \geq 3$, any $T \leq d - 2$ and any deterministic first-order algorithm, there exists an L-Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$ such that $f(\mathbf{x}_0) - \inf_{\mathbf{x}} f(\mathbf{x}) \leq \Delta$, yet the first T iterates produced by the algorithm when applied to f are not (δ, ϵ)-stationary points for any $\delta < \frac{\Delta}{L}$, $\epsilon < \frac{L}{252}$.*

Our result highlights that even though finding a (δ, ϵ)-Goldstein stationary point in nonsmooth nonconvex optimization is computationally tractable using a randomized algorithm, it is essentially harder than finding an ϵ-stationary point in smooth nonconvex optimization without randomization as it requires $\Omega(d)$ oracle calls. We also note that Theorem 3.1 holds true regardless of the relationship between the dimension d and the parameters (δ, ϵ), in contrast to the dimension-independent lower bounds established for nonsmooth convex optimization [Nesterov, 2018], where the accuracy parameter must scale polynomially with $1/d$.

The full proof of Theorem 3.1 is deferred to Section 6.1, though we will now provide a proof sketch. For any deterministic first-order algorithm, if an oracle can always return the "uninformative" answer

[3] For the purpose of this work it makes no difference whether the algorithm gets to see some subgradient or the whole Clarke subgradient set. That is, the lower bounds to follow hold even if the algorithm has access to the *entire* subgradient set, while the upper bounds hold even if the algorithm receives a single arbitrary subgradient.

$f(\mathbf{x}_t) = 0, \nabla f(\mathbf{x}_t) = \mathbf{e}_1$ this fixes the iterates $\mathbf{x}_1, \ldots, \mathbf{x}_T$. Hence, it remains to construct a Lipschitz function that will be consistent with the oracle answers, yet all the queried points are not (δ, ϵ)-stationary. To that end, we construct a function which in a very small neighborhood of each queried point $\mathbf{x}_t$ looks like $\mathbf{x} \mapsto \mathbf{e}_1^\top(\mathbf{x} - \mathbf{x}_t)$, yet in most of the space looks like $\mathbf{x} \mapsto \max\{\mathbf{v}^\top \mathbf{x}, -1\}$, which has (δ, ϵ)-stationary points only when $\mathbf{x}$ is correlated with $-\mathbf{v}$. By letting $\mathbf{v}$ be some vector which is orthogonal to all the queried points (which is possible as long as $T < d - 1$), we obtain the result.

This construction relies crucially on the function being highly nonsmooth—essentially interpolating between two orthogonal linear functions in an arbitrarily small neighborhood. As it will turn out, if the function to be optimized is even slightly smooth, then the theorem can be bypassed, as we will show in Section 4.

3.2 Lower bound for gradient-only oracle

In this section, we demonstrate the importance of having access either to randomness or to a zeroth-order oracle, namely to the function value. In particular, we prove that any deterministic algorithm which has access only to a gradient oracle cannot return an approximate Goldstein stationary point within any finite number of iterations.

Theorem 3.2 *For any $0 < \delta < \epsilon < 1$, any $d \in \mathbb{N}$, $T < \infty$, and any deterministic algorithm which has access only to a gradient oracle, there exists a 1-Lipschitz function $f : \mathbb{R}^d \to [-1, 1]$ such that the algorithm cannot guarantee to return a (δ, ϵ)-Goldstein stationary point using T oracle calls.*

We will now sketch the proof; see Section 6.2 for the full proof. We can assume without loss of generality that $d = 1$ (otherwise we can simple apply the "hard" construction to the first coordinate). Suppose a deterministic algorithm has access only to a derivative oracle, which always returns the "uninformative" answer $f'(\mathbf{x}_t) = 1$. This fixes the algorithm's iterates $\mathbf{x}_1, \ldots, \mathbf{x}_T$, which then attempts to guarantee that some returned point $\hat{\mathbf{x}}$ is a (δ, ϵ)-stationary point. It remains to construct a Lipschitz function that will be consistent with the oracle answers, yet $\hat{\mathbf{x}}$ will not be (δ, ϵ)-stationary. To that end, we construct a function which looks like $\mathbf{x} \mapsto \mathbf{x} - \hat{\mathbf{x}}$ in a long enough segment around $\hat{\mathbf{x}}$, ensuring it is indeed not (δ, ϵ)-stationary. On the other hand, in a very small neighborhood of each queried point $\mathbf{x}_t$ we add a "bump" so that the function looks like $\mathbf{x} \mapsto \mathbf{x} - \mathbf{x}_t$, consistent with our resisting oracle. Finally, far away from all queried points we let the function be constant, so that its image remains in $[-1, 1]$.

Remark 3.1 *In contrast with Theorem 3.2, there exist randomized algorithms that access only a gradient oracle and guarantee to return a (δ, ϵ)-Goldstein stationary point in finite-time. Indeed, Lin et al. [2022, Theorem 3.1] have shown that for $f_\delta(\mathbf{x}) := \mathbb{E}_{\mathbf{u} \sim B_\delta(\mathbf{x})}[f(\mathbf{u})]$ it holds that $\nabla f_\delta(\mathbf{x}) = \mathbb{E}_{\mathbf{u} \sim B_\delta(\mathbf{x})}[\nabla f(\mathbf{u})] \in \partial_\delta f(\mathbf{x})$ (where $\mathbf{u} \sim B_\delta(\mathbf{x})$ is distributed uniformly over a Euclidean ball of radius δ centered at $\mathbf{x}$), thus it suffices to find an ϵ-stationary point of f_δ. But since $\nabla f(\mathbf{u})$ is an unbiased estimator of $\nabla f_\delta(\mathbf{x})$ and $\|\nabla f(\mathbf{u})\| \leq L$, this is well known to be possible using stochastic gradient descent [Ghadimi and Lan, 2013]. In particular, the same argument as in the proof of Lin et al. [2022, Theorem 3.2] shows that it is possible to find such a point within $O(\sqrt{d}(L^4\epsilon^{-4} + \Delta L^3 \delta^{-1} \epsilon^{-4}))$ calls to a gradient oracle.*

Remark 3.2 *Note that in nonsmooth convex optimization or in smooth nonconvex optimization, a deterministic algorithm can obtain (δ, ϵ)-Goldstein stationary points using only a gradient oracle, even at a dimension-free rate. Indeed, in the nonsmooth convex case gradient descent returns $\mathbf{x}$ such that*

Algorithm 1 BINARY-SEARCH(δ, $\nabla f(\cdot)$, $\mathbf{g}_0$, $\mathbf{x}$)

Initialization: Set $b \leftarrow \delta$, $a \leftarrow 0$ and $t \leftarrow b$.
while $-\nabla f(\mathbf{x} - t\frac{\mathbf{g}_0}{\|\mathbf{g}_0\|}) \cdot \frac{\mathbf{g}_0}{\|\mathbf{g}_0\|} + \frac{1}{2}\|\mathbf{g}_0\| \geq -\frac{\epsilon}{4}$ **do**
  Set $t \leftarrow \frac{a+b}{2}$.
  if $f(\mathbf{x} - b\frac{\mathbf{g}_0}{\|\mathbf{g}_0\|}) + \frac{b}{2}\|\mathbf{g}_0\| > f(\mathbf{x} - t\frac{\mathbf{g}_0}{\|\mathbf{g}_0\|}) + \frac{t}{2}\|\mathbf{g}_0\|$ **then**
    Set $a \leftarrow t$.
  else
    Set $b \leftarrow t$.
  end if
end while
Output: $\nabla f(\mathbf{x} - t\frac{\mathbf{g}_0}{\|\mathbf{g}_0\|})$.

$f(\mathbf{x}) - \inf_{\mathbf{x}} f(\mathbf{x}) < \delta\epsilon$ within $O(\delta^{-2}\epsilon^{-2})$ gradient evaluations, and any such point is in particular a (δ, ϵ)-Goldstein stationary point.[4] *Similarly, in the smooth nonconvex setting gradient descent returns an ϵ-stationary point within $O(\epsilon^{-2})$ gradient evaluations, which is trivially also a (δ, ϵ)-Goldstein stationary point.*

4 Deterministic algorithm for slightly smooth functions

In this section we show that if the objective function is even slightly smooth, then the dimension free rate of $\widetilde{O}(\delta^{-1}\epsilon^{-3})$ can be obtained by a deterministic first-order algorithm, incurring a mild logarithmic dependence on the smoothness parameter.

Theorem 4.1 *Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is L-Lipschitz, H-smooth, and $\mathbf{x}_0 \in \mathbb{R}^d$ is such that $f(\mathbf{x}_0) - \inf_{\mathbf{x}} f(\mathbf{x}) \leq \Delta$. Then* DETERMINISTIC-GOLDSTEIN-SG($\mathbf{x}_0, \delta, \epsilon$) *(Algorithm 2) is a deterministic first-order algorithm that given any $\delta, \epsilon \in (0, 1)$ returns a (δ, ϵ)-Goldstein stationary point of f within $T = O\left(\frac{\Delta L^2 \log(HL\delta/\epsilon)}{\delta\epsilon^3}\right)$ oracle calls.*

The idea behind this de-randomization is to replace a certain randomized line search in the algorithms of Zhang et al. [2020], Davis et al. [2022], Tian et al. [2022], which in turn are based on Goldstein [1977], with a deterministic binary search subroutine, Algorithm 1. This subroutine terminates within $O(\log(H\delta/\epsilon))$ steps provided that the function is H-smooth. We note that such a procedure was derived by Davis et al. [2022] for any H-weakly convex function along differentiable directions, and since any H-smooth function is H-weakly convex and differentiable along any direction, this can be applied in an identical manner. Although this algorithmic ingredient appears inside a proof of Davis et al. [2022], they use it in a different manner in order to produce a *randomized* algorithm for weakly convex functions in low dimension, with different guarantees suitable for that setting. We defer the full analyses of Algorithm 1, Algorithm 2 which lead to the proof of Theorem 4.1 to Section 6.3.

[4] Otherwise, let $\mathbf{g}$ be the minimal norm element in $\partial_\delta f(\mathbf{x})$, and assume by contradiction that $\|\mathbf{g}\| > \epsilon$. Then Goldstein [1977] ensures that $f(\mathbf{x} - \frac{\delta}{\|\mathbf{g}\|}\mathbf{g}) \leq f(\mathbf{x}) - \delta\|\mathbf{g}\| < f(\mathbf{x}) - \delta\epsilon < \inf_{\mathbf{x}} f(\mathbf{x})$ which is a contradiction.

Algorithm 2 DETERMINISTIC-GOLDSTEIN-SG($\mathbf{x}_0, \delta, \epsilon$)

Input: initial point $\mathbf{x}_0 \in \mathbb{R}^d$, accuracy parameters $\delta, \epsilon \in (0, 1)$.
for $t = 0, 1, 2, \ldots, T-1$ **do**
  Set $\mathbf{g}(\mathbf{x}_t) \leftarrow \nabla f(\mathbf{x}_t)$.
  while $f(\mathbf{x}_t - \delta \frac{\mathbf{g}(\mathbf{x}_t)}{\|\mathbf{g}(\mathbf{x}_t)\|}) - f(\mathbf{x}_t) > -\frac{\delta}{2}\|\mathbf{g}(\mathbf{x}_t)\|$ and $\|\mathbf{g}(\mathbf{x}_t)\| > \epsilon$ **do**
    Set $\mathbf{g}_{\text{new}} \leftarrow$ BINARY-SEARCH($\delta, \nabla f(\cdot), \mathbf{g}(\mathbf{x}_t), \mathbf{x}_t$).
    $\mathbf{h}_t \leftarrow \arg\min\{\|\mathbf{g}(\mathbf{x}_t) + \lambda(\mathbf{g}_{\text{new}} - \mathbf{g}(\mathbf{x}_t))\| : 0 \le \lambda \le 1\}$.
    $\mathbf{g}(\mathbf{x}_t) \leftarrow \mathbf{h}_t$.
  end while
  if $\|\mathbf{g}(\mathbf{x}_t)\| \le \epsilon$ **then**
    Stop.
  else
    $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - \delta \frac{\mathbf{g}(\mathbf{x}_t)}{\|\mathbf{g}(\mathbf{x}_t)\|}$.
  end if
end for
Output: $\mathbf{x}_t$.

5 Deterministic smoothings

Motivated by the mild smoothness dependence of DETERMINISTIC-GOLDSTEIN-SG (Algorithm 2) as proved in Theorem 4.1, we turn to the design of smoothing procedures. These are algorithms that act on a Lipschitz function, and return a smooth approximation—allowing the use of smooth optimization methods. Smoothing nonsmooth functions in order to allow the use of smooth optimization algorithms is a longstanding approach for nonsmooth nonconvex optimization, both in practice and in theoretical analyses. We refer to Appendix A for references. From a computational perspective, it is not clear what it means for an algorithm to "receive" a real function as an input. For this reason, we make the distinction between "black-box" smoothings which are granted oracle access to the original function, and "white-box" smoothings which are assumed to have access to additional structural information.

5.1 Black-box smoothings

Recently, Kornowski and Shamir [2021] have studied black-box smoothings from an oracle complexity viewpoint. One of their main results is that randomized smoothing [Duchi et al., 2012] is an optimal smoothing procedure, in the sense that no efficient black-box smoothing procedure can yield an approximation whose smoothness parameter is lower than $O(\sqrt{d})$, which is achieved by randomized smoothing. In particular, this implies that any efficient black-box smoothing unavoidably suffers from some dimension dependence. In that paper, the authors posed the open question of assessing what can be achieved by a deterministic black-box smoothing, since efficient randomized smoothing is only able to return stochastic estimates of the smoothed function. We solve this question for all "reasonable" smoothing procedures, as defined next. Without loss of generality, we consider functions whose Lipschitz constant is 1, since if the objective function is L-Lipschitz the algorithm can simply rescale it by L.

Definition 5.1 *An algorithm $\mathcal{S}$ is called a black-box smoothing with complexity $T \in \mathbb{N}$, if it uses a first-order oracle of a 1-Lipschitz function $f : \mathbb{R}^d \to \mathbb{R}$, such that given any $\mathbf{x} \in \mathbb{R}^d$ it sequentially*

queries f's oracle at T points and returns $\widetilde{f}(\mathbf{x})$, $\mathbf{g}_{\mathbf{x}} = \nabla\widetilde{f}(\mathbf{x})$ for some smooth $\widetilde{f} : \mathbb{R}^d \to \mathbb{R}$. We say that the smoothing algorithm is ***meaningful*** *if $\widetilde{f}$ is* $\mathrm{poly}(d)$*-smooth, and any (δ, ϵ)-Goldstein stationary point of $\widetilde{f}$ is a $(\mathrm{poly}(\delta, \epsilon), \mathrm{poly}(\delta, \epsilon))$-Goldstein stationary point of f.*

In other words, a smoothing fails to be meaningful if either the smooth approximation has super-polynomial smoothness (thus can hardly be treated as smooth), or introduces completely "fake" approximately-stationary points of f.[5] The latter case implies that running a nonconvex optimization algorithm over $\widetilde{f}$ fails to provide any meaningful guarantee for the original function f. Note that these assumptions are extremely permissive, as we allow for any polynomial parameter blow-up, and do not even quantify the requirement regarding the accuracy of the approximation. Notably, all black-box smoothings considered in the literature, including randomized smoothing and the Moreau-Yosida smoothing for weakly-convex functions [Davis and Drusvyatskiy, 2019], are easily verified to be meaningful (and, indeed, satisfy more stringent conditions with respect to the original function). Further note that the randomized complexity of these procedures is dimension-free.

Under this mild assumption, our previous theorems readily imply a answer to the question posed by Kornowski and Shamir [2021].

Theorem 5.1 *There is no deterministic, black-box meaningful smoothing algorithm with dimension-free complexity.*

Proof. Assuming towards contradiction there is such a smoothing algorithm $\mathcal{S}$, we compose it with Deterministic-Goldstein-SG. Namely, given any Lipschitz f, we consider the first-order algorithm obtained by applying Algorithm 2 to $\widetilde{f} = \mathcal{S}(f)$. Since $\widetilde{f}$ is $\mathrm{poly}(d)$-smooth, and any (δ, ϵ)-Goldstein stationary point of $\widetilde{f}$ is a $(\mathrm{poly}(\delta, \epsilon), \mathrm{poly}(\delta, \epsilon))$-Goldstein stationary point of f, by Theorem 4.1 we obtain overall a deterministic algorithm that finds a $(\mathrm{poly}(\delta, \epsilon), \mathrm{poly}(\delta, \epsilon))$-Goldstein stationary point of f within $O(\log(\mathrm{poly}(d)) \cdot \mathrm{poly}(\delta^{-1}, \epsilon^{-1})) = O(\log(d) \cdot \mathrm{poly}(\delta^{-1}, \epsilon^{-1}))$ first-order oracle calls—contradicting Theorem 3.1. □

5.2 Deterministic smoothing of ReLU networks

In this section we introduce a smoothing technique that can be applied to optimization of non-smooth functions, provided that they are expressed as ReLUs in a neural network accessible to the smoothing procedure. The idea of utilizing the representation of a function, as opposed to just having oracle access to it, has been commonly used across diverse domains, from purely theoretical applications e.g., computational complexity theory, Daskalakis and Papadimitriou, 2011, Fearnley et al., 2021 to practical applications e.g., deep neural networks, LeCun et al., 2015, Goodfellow et al., 2016. We refer to this function representation as the *white-box model* to contrast it with the previously discussed black-box model. Our results demonstrate that having such a white box access is powerful enough to allow for meaningful deterministic smoothing, as opposed to the black-box model whose insufficiency is established in Theorem 5.1.

We start by giving a brief overview of the key observation underlying our deterministic smoothing approach. Consider a single ReLU neuron with a bias term, namely for some point $\mathbf{x} \in \mathbb{R}^d$, weight $\mathbf{w} \in \mathbb{R}^d$ and bias $b \in \mathbb{R}$:

$$(\mathbf{x}, \mathbf{w}, b) \mapsto \mathrm{relu}(\mathbf{w}^\top \mathbf{x} + b) := \max\{\mathbf{w}^\top \mathbf{x} + b, 0\} \ . \tag{5.1}$$

[5] It is important to recall that for smooth $\widetilde{f}$ the notions of approximate-Clarke stationarity and approximate-Goldstein stationarity coincide [Zhang et al., 2020, Proposition 6].

We replace the nonsmooth ReLU with a smooth, carefully chosen "Huberized" function:

$$\text{softrelu}_\gamma(z) = \mathbb{E}_{\xi\sim U[-\gamma,\gamma]}[\text{relu}(z+\xi)] = \begin{cases} z\ , & z \geq \gamma \\ \frac{(z+a)^2}{4a}\ , & -\gamma \leq z < \gamma \\ 0\ , & z < -\gamma, \end{cases}$$

for some small $\gamma > 0$. Accordingly, we obtain the "smoothed" neuron of the form

$$\begin{aligned}(\mathbf{x}, \mathbf{w}, b) &\mapsto \text{softrelu}_a(\mathbf{w}^\top \mathbf{x} + b) \\ &= \mathbb{E}_{\xi\sim U[-\gamma,\gamma]}[\text{relu}(\mathbf{w}^\top \mathbf{x} + (b+\xi))] = \begin{cases} \mathbf{w}^\top \mathbf{x} + b\ , & \mathbf{w}^\top \mathbf{x} + b \geq \gamma \\ \frac{(\mathbf{w}^\top \mathbf{x}+b+a)^2}{4a}\ , & -\gamma \leq \mathbf{w}^\top \mathbf{x} + b < \gamma \\ 0\ , & \mathbf{w}^\top \mathbf{x} + b < -\gamma \end{cases}.\end{aligned}$$

Optimizing the function above (as a component of a larger neural network) with respect to $(\mathbf{w}, b)$ is the goal of any optimizer seeking to "train" the network's parameters to fit its input $\mathbf{x}$. We see that on one hand the smoothed neuron is a closed-form smooth approximation of the ReLU neuron in Eq. (5.1), yet is mathematically equivalent to randomized smoothing *over the bias term*. Hence, we obtain the meaningful guarantees of randomized smoothing, namely that optimizing the smoothed model corresponds to optimizing the original nonsmooth function, without the need for randomization. Moreover, as opposed to plain randomized smoothing which would smooth with respect to $(\mathbf{w}, b) \in \mathbb{R}^{d+1}$, thus suffering from a dimension dependence in the smoothness parameter, smoothing over b alone avoids dependence on d. Overall, replacing all ReLU neurons of a network with smoothed neurons is mathematically equivalent to randomized smoothing over the parameter subspace corresponding to all bias terms, reducing the dimension-dependence to a dependence on the number of biases, roughly the size of the network.[6]

We now formally describe the class of representations that our smoothing procedure will apply to. It is easy to see that this class contains ReLU neural networks with biases of arbitrary depth and width, including many architectures used in practice.

Definition 5.2 (Neural Arithmetic Circuits (NAC)) *We say that $\mathcal{C}$ is a* neural arithmetic circuit with biases *if it is represented as a directed acyclic graph with four different group of nodes: (i) input nodes; (ii) bias nodes; (iii) output nodes; and (iv) gate nodes. The gate node can be one of $\{+, \text{relu}, \times, \text{const(c)}\}$, where const(c) stands for a constant $c \in [-1, 1]$. Moreover, a valid NAC with biases satisfies the following conditions:*

1. *There is at least one input node. Every input node has 0 incoming edges and any number of outgoing edges.*
2. *The number of bias nodes is equal to the number of* relu *gates. Every bias node has 0 incoming edges, and only one outgoing edge.*
3. *The gate nodes in $\{+, \times\}$ have two incoming edges, and any number of outgoing edges.*[7]

[6] Note that due to dependencies between neurons at different layers, this does not correspond to standard randomized smoothing with respect to an isotropic distribution, but rather to a nontrivial distribution capturing the dependencies among different bias terms. We remark that this is a major technical challenge in proving Theorem 5.2 to follow.

[7] We can generalize it to the case of any finite number of inputs. Focusing on two incoming edges does not lack the generality since we can always compose these gates to simulate addition and maximum with many inputs by just increasing the size and the depth of the circuit by a logarithmic factor.

4. *The gate node* const(c) *has 0 incoming edges, and any number of outgoing edges.*

5. *The gate node* relu *has 1 incoming edge but any number of outgoing edges. We also assume that all the* relu *gates have biases, i.e., the predecessor vertex of a* relu *gate is always a "+" gate connected to a bias node that is unique for every* relu *gate.*

6. *There is only one output node that has 1 incoming edge and 0 outgoing edges.*

We denote by $s(\mathcal{C})$ the size of $\mathcal{C}$ (i.e., the number of nodes in the graph of $\mathcal{C}$).

The interpretation of $\mathcal{C}$ as a function $f : \mathbb{R}^d \to \mathbb{R}$ is very intuitive. The input nodes correspond to the input variables $x_1, \ldots, x_d$, followed by a gate node defining arithmetic operations over their input, finally producing $f(\mathbf{x})$ in the output node.

Example 5.1 *Consider training a neural network $\Phi_{\mathbf{W},\mathbf{b}}$ to fit a labeled dataset $(\mathbf{x}_i, y_i)_{i=1}^n$ with respect to the quadratic loss, where $\mathbf{W}, \mathbf{b}$ are the vectors of weights and biases of Φ, respectively. This task corresponds to minimizing the following function:*

$$f(\mathbf{W}, \mathbf{b}) = \sum_{i=1}^{n} \left(\Phi_{\mathbf{W},\mathbf{b}}(\mathbf{x}_i) - y_i\right)^2 .$$

It is easy to see that this function can be expressed as a neural arithmetic circuit according to Definition 5.2. The only requirement for $\Phi_{\mathbf{W},\mathbf{b}}$ is that every relu *gate has a unique bias variable. Examples for such $\Phi_{\mathbf{W},\mathbf{b}}$ include feed-forward ReLU networks, convolutional networks, and residual neural network with skip connections.*

Following the example above, we see that the problem of finding a (δ, ϵ)-Goldstein stationary point of a function represented by a neural arithmetic circuit $\mathcal{C}$ captures a wide range of important nonsmooth and nonconvex problems. To prove the efficiency of our proposed method, we need to impose the following assumption, measuring the extent to which function values increase throughout the neural arithmetic circuit. We note in Remark 5.1 that this assumption is satisfied by the practical design of deep neural networks.

Assumption 5.1 *For $G > 0$, we say that $h : \mathbb{R}^d \to \mathbb{R}$ is G-bounded over $\mathcal{R}$ if $|h|_{\mathcal{R}}| \leq G$. Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is represented as a linear arithmetic circuit $\mathcal{C}$. Let $v_1, \ldots, v_n$ be the nodes in $\mathcal{C}$ and f_i be the function that will be computed if v_i would the output of the neural circuit. We assume that there is a set $\mathcal{R} \subseteq \mathbb{R}^d$, such that for all $i \in [n]$: f_i is L_i-Lipschitz and G_i-bounded over $\mathcal{R}$, according to the following composition rules:*

- ***v_i is a $+$ gate:*** *if $f_i = f_j + f_k$ then $L_i = L_j + L_k$ and $G_i = G_j + G_k$.*

- ***v_i is a* relu *gate:*** *if $f_i = \text{relu}\{f_j\}$ then $L_i = L_j$ and $G_i = G_j$.*

- ***v_i is a* const(c) *gate:*** *$L_i = 0$ and $G_i = c$.*

- ***v_i is a $\times$ gate:*** *if $f_i = f_j \cdot f_k$ then $L_i = L_j \cdot G_k + G_j \cdot L_k$ and $G_i = G_j \cdot G_k$.*

- ***v_i is a input or a bias node:*** *$L_i = 1$, $G_i = \text{diam}(\mathcal{R})$ (the diameter of $\mathcal{R}$).*

In particular, we assume that f is L-Lipschitz and G bounded over $\mathcal{R}$ according to the rules above. In this case, we say that f is L-recursively Lipschitz and G-recursively bounded in $\mathcal{R}$.

Remark 5.1 *Note that the recursive rules used in Assumption 5.1 always provide an upper bound on $L > 0$, however this bound can be much larger than the true Lipschitz constant L in the worst-case. To bypass these bad cases, we impose Assumption 5.1. Notably, this assumption is not theoretically artificial but is satisfied by generic constructions of neural networks in the context of deep learning. Indeed, since the $+$ and $\times$ gates are often used consecutively, leading to a bad Lipschitz constant in the worst case, practitioners often force these upper bounds to be as small as possible by employing normalization techniques in order to stabilize the training [Ioffe and Szegedy, 2015, Miyato et al., 2018].*

As previously discussed, our deterministic smoothing idea is to replace the relu activation function with its carefully chosen smooth alternative softrelu. We emphasize that this smoothing procedure is simple, implementable and inspired by techniques that are widely accepted in practice (e.g Tatro et al., 2020, Shamir et al., 2020). While proving that this results in a smooth approximation of the original function is relatively straightforward, the main novelty of our proof is showing that any (δ, ϵ)-Goldstein stationary point of the smoothed model is a (δ, ϵ)-Goldstein stationary point of the original, following from our observation of the equivalence to randomized smoothing with respect to a low dimensional subspace. This is crucial, as it allows optimization of the original function to be carried through the smoothed model. We are now ready to state our main theorem in this section, whose proof is deferred to Section 6.4.

Theorem 5.2 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a L-recursively Lipschitz and G-recursively bounded function in $\mathcal{R} \subseteq \mathbb{R}^d$ (see Assumption 5.1), represented by a neural arithmetic circuit $\mathcal{C}$. For every $\gamma > 0$, we can construct a function $\widetilde{f} : \mathbb{R}^d \to \mathbb{R}$ such that for all $\mathbf{x} \in \mathcal{R}$ it holds that:*

1. $|f(\mathbf{x}) - \widetilde{f}(\mathbf{x})| \leq \gamma$.
2. *$\widetilde{f}$ is L-Lipschitz and G-bounded.*
3. *$\widetilde{f}$ is $\frac{(G \cdot L)^{O(s(\mathcal{C}))}}{\min\{\epsilon, \delta, \gamma\}}$-smooth.*
4. *Every (δ, ϵ)-Goldstein stationary point of $\widetilde{f}$ is a (δ', ϵ')-Goldstein stationary point of f with $\epsilon' = 2\epsilon$ and $\delta' = 2\delta$.*

At first glance, the smoothness parameter provided by the theorem above, though dimension-independent, may seem overwhelming as it depends exponentially on the size of the network. Luckily, this brings us back to Theorem 4.1 where we have proved that it is possible to incur merely a logarithmic dependence on this parameter, resulting in the following corollary by setting $\gamma = \min\{\epsilon, \delta\}$.

Corollary 5.3 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a L-recursively Lipschitz and G-recursively bounded function in $\mathcal{R} \subseteq \mathbb{R}^d$ (see Assumption 5.1), represented by a neural arithmetic circuit $\mathcal{C}$. Then if we apply* Deterministic-Goldstein-SG *(Algorithm 2) to the function $\widetilde{f}$ defined in Theorem 5.2 and $\mathcal{R}$ is such that the algorithm's iterates do not escape $\mathcal{R}$, the algorithm is guaranteed to return a (δ, ϵ)-Goldstein stationary point of f using $O\left(\frac{GL^2 s(\mathcal{C}) \log(GL\delta/\epsilon)}{\delta \epsilon^3}\right)$ first-order oracle calls.*

6 Proofs

6.1 Proof of Theorem 3.1

Fix $d \geq 3,\ \Delta, L > 0$ and let $T \leq d-2$. Consider the case that for any $t \in [T-1]$ the first-order oracle response is $f(\mathbf{x}_t) = 0, \nabla f(\mathbf{x}_t) = \mathbf{e}_1$. Since the algorithm is deterministic this fixes the iterate sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$. We will show this resisting strategy is indeed consistent with a function which satisfies the conditions in the theorem.

To that end, we denote $r := \min_{1 \leq i \neq j \leq T} \|\mathbf{x}_i - \mathbf{x}_j\|/4 > 0$ (without loss of generality) and fix some $\mathbf{v} \in (\mathrm{span}\{\mathbf{e}_1, \mathbf{x}_1, \ldots, \mathbf{x}_T\})^\perp$ with $\|\mathbf{v}\| = 1$ (which exists since $d \geq T+2$). For any $\mathbf{z} \in \mathbb{R}^d$ we define

$$g_{\mathbf{z}}(\mathbf{x}) := \min\{\|\mathbf{x}-\mathbf{z}\|^2/r^2, 1\}\mathbf{v}^\top \mathbf{x} + (1 - \min\{\|\mathbf{x}-\mathbf{z}\|^2/r^2, 1\})\mathbf{e}_1^\top(\mathbf{x}-\mathbf{z}) \ ,$$

and further define

$$h(\mathbf{x}) := \begin{cases} \mathbf{v}^\top \mathbf{x}\,, & \forall t \in [T] : \|\mathbf{x}-\mathbf{x}_t\| \geq r \\ g_{\mathbf{x}_t}(\mathbf{x})\,, & \exists t \in [T] : \|\mathbf{x}-\mathbf{x}_t\| < r \end{cases} .$$

Note that h is well defined since by definition of r there cannot be $i \neq j$ such that $\|\mathbf{x}-\mathbf{x}_i\| < r$ and $\|\mathbf{x}-\mathbf{x}_j\| < r$.

Lemma 6.1 *$h : \mathbb{R}^d \to \mathbb{R}$ as defined above is 7-Lipschitz, satisfies for any $t \in [T]$: $h(\mathbf{x}_t) = 0, \nabla h(\mathbf{x}_t) = \mathbf{e}_1$ and has no $(\delta, \frac{1}{36})$-stationary points for any $\delta > 0$.*

Proof. We start by noting that h is continuous, since for any $\mathbf{z}$ and $(\mathbf{y}_n)_{n=1}^\infty \subset B_r(\mathbf{z})$, $\mathbf{y}_n \overset{n\to\infty}{\longrightarrow} \mathbf{y}$ such that $\|\mathbf{y}-\mathbf{z}\| = r$ we have

$$\begin{aligned}
\lim_{n\to\infty} h(\mathbf{y}_n) &= \lim_{n\to\infty} g_{\mathbf{z}}(\mathbf{y}_n) \\
&= \lim_{n\to\infty} \left(\min\{\|\mathbf{y}_n-\mathbf{z}\|^2/r^2, 1\}\mathbf{v}^\top \mathbf{y}_n + (1 - \min\{\|\mathbf{y}_n-\mathbf{z}\|^2/r^2, 1\})\mathbf{e}_1^\top(\mathbf{y}_n-\mathbf{z})\right) \\
&= \lim_{n\to\infty} \left(\frac{\|\mathbf{y}_n-\mathbf{z}\|^2}{r^2} \cdot \mathbf{v}^\top \mathbf{y}_n + \left(1 - \frac{\|\mathbf{y}_n-\mathbf{z}\|^2}{r^2}\right)\mathbf{e}_1^\top(\mathbf{y}_n-\mathbf{z})\right) \\
&= \mathbf{v}^\top \mathbf{y} \ .
\end{aligned}$$

Having established continuity, since $\mathbf{x} \mapsto \mathbf{v}^\top \mathbf{x}$ is clearly 1-Lipschitz (in particular 7-Lipschitz), in order to prove Lipschitzness of h it is enough to show that $g_{\mathbf{x}_t}(\mathbf{x})$ is 7-Lipschitz in $\|\mathbf{x}-\mathbf{x}_t\| < r$ for any $\mathbf{x}_t$. For any such $\mathbf{x}, \mathbf{x}_t$ we have

$$\begin{aligned}
\nabla g_{\mathbf{x}_t}(\mathbf{x}) &= \frac{2\mathbf{v}^\top \mathbf{x}}{r^2}(\mathbf{x}-\mathbf{x}_t) + \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\mathbf{v} - \frac{2\mathbf{e}_1^\top(\mathbf{x}-\mathbf{x}_t)}{r^2}(\mathbf{x}-\mathbf{x}_t) - \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\mathbf{e}_1 + \mathbf{e}_1 \\
&\overset{\mathbf{v}\perp\mathbf{x}_t}{=} \frac{2\mathbf{v}^\top(\mathbf{x}-\mathbf{x}_t)}{r^2}(\mathbf{x}-\mathbf{x}_t) + \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\mathbf{v} - \frac{2\mathbf{e}_1^\top(\mathbf{x}-\mathbf{x}_t)}{r^2}(\mathbf{x}-\mathbf{x}_t) - \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\mathbf{e}_1 + \mathbf{e}_1 \ , \quad (6.1)
\end{aligned}$$

hence

$$\begin{aligned}
\|\nabla g_{\mathbf{x}_t}(\mathbf{x})\| &= \left\| \frac{2\mathbf{v}^\top(\mathbf{x}-\mathbf{x}_t)}{r^2}(\mathbf{x}-\mathbf{x}_t) + \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\mathbf{v} - \frac{2\mathbf{e}_1^\top(\mathbf{x}-\mathbf{x}_t)}{r^2}(\mathbf{x}-\mathbf{x}_t) - \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\mathbf{e}_1 + \mathbf{e}_1 \right\| \\
&\leq \frac{2\|\mathbf{v}\| \cdot \|\mathbf{x}-\mathbf{x}_t\|^2}{r^2} + \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\|\mathbf{v}\| + \frac{2\|\mathbf{e}_1\| \cdot \|\mathbf{x}-\mathbf{x}_t\|^2}{r^2} + \frac{\|\mathbf{x}-\mathbf{x}_t\|^2}{r^2}\|\mathbf{e}_1\| + \|\mathbf{e}_1\| \\
&\leq 2+1+2+1+1 = 7 \ ,
\end{aligned}$$

which proves the desired Lipschitz bound. The fact that for any $t \in [T] : h(\mathbf{x}_t) = 0, \nabla h(\mathbf{x}_t) = \mathbf{e}_1$ is easily verified by construction and by Eq. (6.1). In order to finish the proof, we need to show that h has no $(\delta, \frac{1}{36})$ stationary-points. By construction we have

$$\partial h(\mathbf{x}) = \begin{cases} \mathbf{v}\,, & \forall t \in [T] : \|\mathbf{x} - \mathbf{x}_t\| > r \\ \nabla g_{\mathbf{x}_t}(\mathbf{x})\,, & \exists t \in [T] : \|\mathbf{x} - \mathbf{x}_t\| < r \end{cases},$$

while for $\|\mathbf{x} - \mathbf{x}_t\| = r$ we would get convex combinations of the two cases.[8] Inspecting the set $\{\nabla g_{\mathbf{x}_t}(\mathbf{x}) : \|\mathbf{x} - \mathbf{x}_t\| < r\}$ through Eq. (6.1), we see that it depends on $\mathbf{x}, \mathbf{x}_t$ only through $\mathbf{x} - \mathbf{x}_t$ and that actually

$$\{\nabla g_{\mathbf{x}_t}(\mathbf{x}) : \|\mathbf{x} - \mathbf{x}_t\| < r\} = \{\nabla g_{\mathbf{0}_d}(\mathbf{x}) : \|\mathbf{x}\| < r\}\,,$$

which is convenient since the latter set does not depend on $\mathbf{x}_t$. Overall, we see that *any* convex combination of gradients of h is in the set

$$\begin{aligned}
&\left\{\lambda_1\mathbf{v} + \lambda_2\left(\frac{2\mathbf{v}^\top\mathbf{x}}{r^2}\mathbf{x} + \frac{\|\mathbf{x}\|^2}{r^2}\mathbf{v} - \frac{2\mathbf{e}_1^\top\mathbf{x}}{r^2}\mathbf{x} - \frac{\|\mathbf{x}\|^2}{r^2}\mathbf{e}_1 + \mathbf{e}_1\right) : \lambda_1, \lambda_2 \geq 0, \lambda_1 + \lambda_2 = 1, \|\mathbf{x}\| \leq r\right\} \\
&= \left\{\lambda_1\mathbf{v} + \lambda_2\left(2\mathbf{v}^\top\mathbf{x}\cdot\mathbf{x} + \|\mathbf{x}\|^2\mathbf{v} - 2\mathbf{e}_1^\top\mathbf{x}\cdot\mathbf{x} - \|\mathbf{x}\|^2\mathbf{e}_1 + \mathbf{e}_1\right) : \lambda_1, \lambda_2 \geq 0, \lambda_1 + \lambda_2 = 1, \|\mathbf{x}\| \leq 1\right\} \\
&= \left\{(\lambda_1 + \lambda_2\|\mathbf{x}\|^2)\mathbf{v} + 2\lambda_2((\mathbf{v} - \mathbf{e}_1)^\top\mathbf{x})\mathbf{x} + \lambda_2(1 - \|\mathbf{x}\|^2)\mathbf{e}_1 : \lambda_1, \lambda_2 \geq 0, \lambda_1 + \lambda_2 = 1, \|\mathbf{x}\| \leq 1\right\}\,.
\end{aligned}$$

We aim to show that the set above does not contain any vectors of norm smaller than $\frac{1}{36}$. Let $\mathbf{u}$ be an element in the set with corresponding $\lambda_1, \lambda_2, \mathbf{x}$ as above. If $\|\mathbf{u}\| \geq 1$ then there is nothing to show, so we can assume $\|\mathbf{u}\| < 1$. We have

$$\begin{aligned}
\mathbf{u}^\top\mathbf{v} &= \lambda_1 + \lambda_2\|\mathbf{x}\|^2 + 2\lambda_2(\mathbf{v} - \mathbf{e}_1)^\top\mathbf{x}\cdot\mathbf{x}^\top\mathbf{v} \\
&= \lambda_1 + \lambda_2\|\mathbf{x}\|^2 + 2\lambda_2(\mathbf{v}^\top\mathbf{x})^2 - 2\lambda_2\mathbf{e}_1^\top\mathbf{x}\cdot\mathbf{x}^\top\mathbf{v} \\
&\geq \lambda_1 + \lambda_2(\mathbf{v}^\top\mathbf{x})^2 + \lambda_2(\mathbf{e}_1^\top\mathbf{x})^2 + 2\lambda_2(\mathbf{v}^\top\mathbf{x})^2 - 2\lambda_2\mathbf{e}_1^\top\mathbf{x}\cdot\mathbf{x}^\top\mathbf{v} \\
&= \lambda_1 + \lambda_2(\mathbf{v}^\top\mathbf{x} - \mathbf{e}_1^\top\mathbf{x})^2 + 2\lambda_2(\mathbf{v}^\top\mathbf{x})^2 \\
&\geq \lambda_1 + \lambda_2(\mathbf{v}^\top\mathbf{x} - \mathbf{e}_1^\top\mathbf{x})^2 \\
&\geq \lambda_2(\mathbf{v}^\top\mathbf{x} - \mathbf{e}_1^\top\mathbf{x})^2\,,
\end{aligned} \tag{6.2}$$

which gives

$$\begin{aligned}
\mathbf{u}^\top(\mathbf{e}_1 + \mathbf{v}) &= \lambda_1 + \lambda_2 + 2\lambda_2(\mathbf{v}^\top\mathbf{x} - \mathbf{e}_1^\top\mathbf{x})(\mathbf{e}_1^\top\mathbf{x} + \mathbf{v}^\top\mathbf{x}) \\
&\geq 1 - 4\lambda_2|\mathbf{v}^\top\mathbf{x} - \mathbf{e}_1^\top\mathbf{x}| \\
&\overset{(6.2)}{\geq} 1 - 4\sqrt{\lambda_2\mathbf{u}^\top\mathbf{v}}\,.
\end{aligned}$$

[8] Since we are interested in analyzing the δ-subdifferential set which consists of convex combinations of subgradients, and subgradients are defined as convex combinations of gradients at differentiable points - it is enough to consider convex combinations of gradients at differentiable points in the first place.

Hence

$$\begin{aligned} 1 &\le |\mathbf{u}^\top(\mathbf{e}_1+\mathbf{v})| + 4\sqrt{\lambda_2 \mathbf{u}^\top \mathbf{v}} \le \|\mathbf{u}\| \cdot \|\mathbf{e}_1+\mathbf{v}\| + 4\sqrt{\|\mathbf{u}\|} \\ &\le \sqrt{2}\|\mathbf{u}\| + 4\sqrt{\|\mathbf{u}\|} \overset{\|\mathbf{u}\|<1}{\le} \sqrt{2\|\mathbf{u}\|} + 4\sqrt{\|\mathbf{u}\|} \\ \implies \|\mathbf{u}\| &\ge \frac{1}{(\sqrt{2}+4)^2} > \frac{1}{36} \,. \end{aligned}$$

□

Given the previous lemma we can easily finish the proof of the theorem by looking at

$$f(\mathbf{x}) := \max\left\{\frac{L}{7}h(\mathbf{x}), -\Delta\right\} \,.$$

f is L-Lipschitz (since h is 7-Lipschitz) and satisfies $f(\mathbf{x}_0) - \inf_{\mathbf{x}} f(\mathbf{x}) \le 0 - (-\Delta) = \Delta$, as required. Furthermore, for any $t \in [T] : h(\mathbf{x}_t) = 0 > -\Delta \implies f(\mathbf{x}_t) = \frac{L}{7}h(\mathbf{x}_t) = 0$. Since f is L-Lipschitz, this further implies that for any $\mathbf{x}$ such that $\|\mathbf{x} - \mathbf{x}_t\| < \frac{\Delta}{L} : \; f(\mathbf{x}) > -\Delta \implies \partial f(\mathbf{x}) = \frac{L}{7}\partial h(\mathbf{x})$. In particular, $\partial_\delta f(\mathbf{x}_t) = \frac{L}{7}\partial_\delta h(\mathbf{x}_t)$ for any $\delta < \frac{\Delta}{L}$, so the lemma shows that $\mathbf{x}_t$ is not a (δ, ϵ) stationary point of f for $\epsilon < \frac{L}{7} \cdot \frac{1}{36} = \frac{L}{252}$.

6.2 Proof of Theorem 3.2

Let $0 < \delta < \epsilon < 1$, and let $T < \infty$. It is enough to prove the case $d = 1$, since otherwise we can simply look at $\mathbf{x} \mapsto f(x_1)$ with f being the lower bound construction in one dimension.

Suppose that an algorithm has access only to a derivative oracle, and consider the case that for any $t \in [T]$ the oracles response is $f'(\mathbf{x}_t) = 1$. Since the algorithm is deterministic this fixes the iterate sequence $Q := (\mathbf{x}_1, \ldots, \mathbf{x}_T)$. Afterwards, the algorithm returns the candidate solution $\hat{\mathbf{x}}$ for being a (δ, ϵ)-Goldstein stationary point. We remark that $\hat{\mathbf{x}}$ might not be in Q. We will show that the described resisting strategy is indeed consistent with a function which satisfies the conditions in the theorem. Namely, it suffices to construct a 1-Lipschitz function f such that $f'(\mathbf{x}_t) = 1$ for all $t \in [T]$ yet $\hat{\mathbf{x}}$ is not a (δ, ϵ)-Goldstein stationary point.

To that end, let $\eta \in (0, 1-\delta)$ be such that $\hat{\mathbf{x}} + \delta + \eta \notin Q$ and $\hat{\mathbf{x}} - \delta - \eta \notin Q$ (recall that Q is finite, thus such η exists). We set $f(\mathbf{x}) = \mathbf{x} - \hat{\mathbf{x}}$ for all $\mathbf{x} \in [\hat{\mathbf{x}} - \delta + \eta, \hat{\mathbf{x}} + \delta + \eta]$, which ensures $\partial_\delta f(\hat{\mathbf{x}}) = \{1\}$, and in particular the norm of the minimal-norm element in $\partial_\delta f(\hat{\mathbf{x}})$ is 1. Since $\epsilon < 1$, we get that $\hat{\mathbf{x}}$ is not a (δ, ϵ)-Goldstein stationary, as required. Moreover, for all $\mathbf{x}_t \in Q \cap [\hat{\mathbf{x}} - \delta + \eta, \hat{\mathbf{x}} + \delta + \eta]$, we have $f'(\mathbf{x}_t) = 1$. Thus, for these query points that lie in the interval $[\hat{\mathbf{x}} - \delta + \eta, \hat{\mathbf{x}} + \delta + \eta]$, we satisfy the resisting oracle condition,

We continue on to define the function $f(\mathbf{x})$ for any $\mathbf{x} > \hat{\mathbf{x}} + \delta + \eta$. The idea is to simply keep $f(\mathbf{x}) = \delta + \eta$ in this range while adding some small bumps to guarantee that $f'(\mathbf{x}_t) = 1$ for all $\mathbf{x}_t \in Q \cap (\hat{\mathbf{x}} + \delta + \eta, \infty)$. Let $\bar{Q} = Q \cup \{\hat{\mathbf{x}} - \delta + \eta, \hat{\mathbf{x}} + \delta + \eta\}$ and $r_1 = \frac{1}{10}\min_{\mathbf{x},\mathbf{x}' \in \bar{Q}, \mathbf{x} \neq \mathbf{x}'}\{|\mathbf{x} - \mathbf{x}'|\}$, we define $r = \min\{r_1, \delta\}$ and

$$f(\mathbf{x}) = \begin{cases} \delta + \eta \,, & \forall \mathbf{x}' \in Q : |\mathbf{x} - \mathbf{x}'| > r \\ \delta + \eta - \mathbf{x} \,, & \exists \mathbf{x}' \in Q : |\mathbf{x} - \mathbf{x}'| \le r \;\text{ and }\; \mathbf{x} \le \mathbf{x}' - \frac{r}{2} \\ \delta + \eta - r + \mathbf{x} \,, & \exists \mathbf{x}' \in Q : |\mathbf{x} - \mathbf{x}'| \le r \;\text{ and }\; \mathbf{x} > \mathbf{x}' - \frac{r}{2} \end{cases} \,.$$

We see from the above definition that $0 \le f(\mathbf{x}) \le \delta + \eta$ for all $\mathbf{x} > \hat{\mathbf{x}} + \delta + \eta$ and $f'(\mathbf{x}) = 1$ for all $\mathbf{x} \in Q \cap (\hat{\mathbf{x}} + \delta + \eta, \infty)$. Similarly, we define $f(\mathbf{x})$ for any $\mathbf{x} < -\hat{\mathbf{x}} - \delta - \eta$ as:

$$f(\mathbf{x}) = \begin{cases} -\delta - \eta \ , & \forall \mathbf{x}' \in Q : |\mathbf{x} - \mathbf{x}'| > r \\ -\delta - \eta + \mathbf{x} \ , & \exists \mathbf{x}' \in Q : |\mathbf{x} - \mathbf{x}'| \le r \ \text{ and } \ \mathbf{x} \le \mathbf{x}' + \frac{r}{2} \\ -\delta - \eta + r - \mathbf{x} \ , & \exists \mathbf{x}' \in Q : |\mathbf{x} - \mathbf{x}'| \le r \ \text{ and } \ \mathbf{x} > \mathbf{x}' + \frac{r}{2} \end{cases} .$$

Putting all the pieces together, we get that f is a 1-Lipschitz function satisfying $f'(\mathbf{x}_t) = 1$ for all $t \in [T]$, and $\hat{\mathbf{x}}$ is not a (δ, ϵ)-Goldstein stationary point for any $0 < \delta < \epsilon < 1$, yielding the desired result.

6.3 Proof of Theorem 4.1

We start by concretely stating the purpose of the binary search given by Algorithm 1.

Lemma 6.2 *Suppose* $\mathbf{x} \in \mathbb{R}^d$ *and* $\mathbf{g}_0 \in \partial_\delta f(\mathbf{x})$ *are such that* $f(\mathbf{x} - \delta \frac{\mathbf{g}_0}{\|\mathbf{g}_0\|}) - f(\mathbf{x}) > -\frac{\delta}{2}\|\mathbf{g}_0\|$ *and* $\|\mathbf{g}_0\| > \epsilon$. *Then* BINARY-SEARCH$(\delta, \nabla f(\cdot), \mathbf{g}_0, \mathbf{x})$ *terminates within* $O(\log(H\delta/\epsilon))$ *first-order oracle calls and returns* $\mathbf{g}_{\text{new}} \in \partial_\delta f(\mathbf{x})$ *such that* $\mathbf{g}_{\text{new}}^\top \mathbf{g}_0 \le \frac{3}{4}\|\mathbf{g}_0\|^2$.

Proof. Using the first assumption on $\mathbf{x}$, $\mathbf{g}_0$ we apply the fundamental theorem of calculus to see that

$$\begin{aligned} \frac{1}{2}\|\mathbf{g}_0\|^2 &\ge \frac{\|\mathbf{g}_0\|}{\delta}\left(f(\mathbf{x}) - f\left(\mathbf{x} - \delta\tfrac{\mathbf{g}_0}{\|\mathbf{g}_0\|}\right)\right) = \frac{1}{\delta}\int_0^\delta \left\langle \nabla f\left(\mathbf{x} - \frac{r}{\|\mathbf{g}_0\|}\mathbf{g}_0\right), \mathbf{g}_0 \right\rangle dr \\ &= \mathbb{E}_{\xi \sim U[\mathbf{x}, \mathbf{x} - \frac{\delta}{\|\mathbf{g}_0\|}\mathbf{g}_0]}\left[\langle \nabla f(\mathbf{x} + \xi), \mathbf{g}_0 \rangle\right] \ , \end{aligned}$$

hence in expectation with respect to the uniform measure over the segment $[\mathbf{x}, \mathbf{x} - \frac{\delta}{\|\mathbf{g}_0\|}\mathbf{g}_0]$, sampling a gradient will indeed satisfy the required condition. Applying the fundamental theorem again, it is easy to see that the **if** condition in Algorithm 1 checks whether the average gradient along the right half of the segment has larger inner product with $\mathbf{g}_0$ than other half or vise versa, and then continues examining the half with the smaller expected inner product. Thus, after k iterations of this process we are left with a segment I_k of length $2^{-k}\delta$ along which

$$\mathbb{E}_{\xi \sim U[I_k]}\left[\langle \nabla f(\mathbf{x} + \xi), \mathbf{g}_0 \rangle\right] \le \frac{1}{2}\|\mathbf{g}_0\|^2.$$

But recalling that ∇f is H-Lipschitz, we get that all gradients of f over I_k are at distance smaller than $H \cdot 2^{-k}\delta$ from one another. In particular, for $k = O(\log(H\delta/\epsilon))$ we get that all $\xi \in I$ satisfy

$$\langle \nabla f(\mathbf{x} + \xi), \mathbf{g}_0 \rangle \le \frac{1}{2}\|\mathbf{g}_0\|^2 + \frac{\epsilon^2}{4} \le \frac{3}{4}\|\mathbf{g}_0\|^2 \ ,$$

where we have applied the second assumption on $\mathbf{g}_0$. Thus the algorithm terminates, and returns $\mathbf{g}_{\text{new}}$ satisfying the required condition. □

Having established the complexity and guarantee produced by the binary search subroutine, we are now ready to analyze DETERMINISTIC-GOLDSTEIN-SG$(\mathbf{x}_0, \delta, \epsilon)$. Since $\mathbf{g}_{\text{new}} \in \partial_\delta f(\mathbf{x}_t)$ and $\partial_\delta f(\mathbf{x})$ is a convex set, we observe that $\mathbf{g}(\mathbf{x}_t) \in \partial_\delta f(\mathbf{x}_t)$. Accordingly, we see that whenever the **while** loop

terminates then either $\|\mathbf{g}(\mathbf{x}_t)\| \leq \epsilon$, meaning that $\mathbf{x}_t$ is a (δ, ϵ)-Goldstein stationary point, or else $f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t) - \frac{\delta}{2}\|\mathbf{g}(\mathbf{x}_t)\| < f(\mathbf{x}_t) - \frac{\delta\epsilon}{2}$. If the former occurs we are done, while the latter can occur at most $\frac{2\Delta}{\delta\epsilon} = O(\frac{\Delta}{\delta\epsilon})$ times by the assumption that $f(\mathbf{x}_0) - \inf_{\mathbf{x}} f(\mathbf{x}) \leq \Delta$.

Hence, it remains to show that the inner loop (lines 5-7) is repeated at most $O\left(\frac{L^2 \log(L/\epsilon)}{\epsilon^2}\right)$ times per outer loop (namely, per t) in order to obtain the desired complexity overall. To that end, assume that $f\left(\mathbf{x}_t - \delta\frac{\mathbf{g}(\mathbf{x}_t)}{\|\mathbf{g}(\mathbf{x}_t)\|}\right) - f(\mathbf{x}_t) > -\frac{\delta}{2}\|\mathbf{g}(\mathbf{x}_t)\|$ and $\|\mathbf{g}(\mathbf{x}_t)\| > \epsilon$. By the previous lemma, we know that $\mathbf{g}_{\text{new}}^\top \mathbf{g}(\mathbf{x}_t) \leq \frac{3}{4}\|\mathbf{g}(\mathbf{x}_t)\|^2$. But that being the case, we get by definition of $\mathbf{h}_t$ that for all $\lambda \in [0,1]$:

$$\begin{aligned}
\|\mathbf{h}_t\|^2 &\leq \|\mathbf{g}(\mathbf{x}_t) + \lambda(\mathbf{g}_{\text{new}} - \mathbf{g}(\mathbf{x}_t))\|^2 \\
&= \|\mathbf{g}(\mathbf{x}_t)\|^2 + 2\lambda\mathbf{g}(\mathbf{x}_t)^\top(\mathbf{g}_{\text{new}} - \mathbf{g}(\mathbf{x}_t)) + \lambda^2\|\mathbf{g}_{\text{new}} - \mathbf{g}(\mathbf{x}_t)\|^2 \\
&\leq (1-2\lambda)\|\mathbf{g}(\mathbf{x}_t)\|^2 + 2\lambda\mathbf{g}(\mathbf{x}_t)^\top\mathbf{g}_{\text{new}} + 4L^2 \\
&\leq \left(1 - \frac{\lambda}{2}\right)\|\mathbf{g}(\mathbf{x}_t)\|^2 + 4L^2 \ .
\end{aligned}$$

By letting $\lambda = \frac{\|\mathbf{g}(\mathbf{x})\|^2}{16L^2}$ and recalling that $\epsilon \leq \|\mathbf{g}(\mathbf{x})\| \leq L$ we get

$$\|\mathbf{h}_t\|^2 \leq \left(1 - \frac{\epsilon^2}{64L^2}\right)\|\mathbf{g}(\mathbf{x}_t)\|^2 \ .$$

Hence each iteration shrinks $\|\mathbf{g}(\mathbf{x}_t)\|^2$ by a factors of $\left(1 - \frac{\epsilon^2}{64L^2}\right)$. Since initially $\|\mathbf{g}(\mathbf{x}_t)\|^2 \leq L^2$, this can happen at most $O\left(\frac{L^2 \log(L/\epsilon)}{\epsilon^2}\right)$ times before having $\|\mathbf{g}(\mathbf{x}_t)\|^2 < \epsilon^2$, as claimed.

6.4 Proof of Theorem 5.2

We construct the function g by using exactly the same neural arithmetic circuit of f, where we replace all the **relu** gates with the **softrelu** gates:

$$\text{softrelu}_a(z) = \begin{cases} z \ , & z \geq a \\ \frac{(z+a)^2}{4a} \ , & -a \leq z < a \\ 0 \ , & z < -a \end{cases} \ ,$$

and note that

$$\text{softrelu}_a(z) = \mathbb{E}_{\xi \sim U[-a,a]}[\text{relu}(z+\xi)] \ .$$

The following summarizes the properties of softrelu gates, and can be easily verified.

Lemma 6.3 *We have that (i)* $|\text{relu}(z) - \text{softrelu}_a(z)| \leq \frac{a}{4}$, *(ii)* $\text{softrelu}_a(\cdot)$ *is* 1*-Lipschitz, and (iii)* softrelu_a *is* $\frac{1}{2a}$*-smooth.*

We prove the theorem inductively, going through the gates of $\mathcal{C}$ one by one with respect to a topological sorting of $\mathcal{C}$ which will remain fixed throughout the proof. Our goal is to compare the evaluation of the nodes, in the order of this topological sorting, in the circuit of f and in the circuit of g under Assumption 5.1. We denote by f_i be the function evaluated in the node i of the circuit of f, in the topological sorting of the circuit of f, and by g_i the corresponding function evaluated in the node

i of the circuit of g. Let also $L_i > 0$ be the corresponding Lipschitz parameter of f_i and $G_i > 0$ the corresponding value bound parameter.

As to the base of our induction, we examine the input nodes. The input nodes and the constant gates for f and g both have the same value, and have gradient $\mathbf{e}_j$ for some j. Thus, they are 0-smooth. Also, the input nodes are 1-Lipschitz and are bounded in value by the diameter of $\mathcal{R}$, while the constant nodes are clearly 0-Lipschitz. This will serve as the basis of our induction.

Our inductive hypothesis is that the following is satisfied for i : For any $\mathbf{x} \in \mathcal{R}$, any $j < i$, it holds that (i) $|f_j(\mathbf{x}) - g_j(\mathbf{x})| \leq \gamma_j$, (ii) g_j is S_j-smooth, (iii) g_j is L_j-Lipschitz, and (iv) g_j is G_j-bounded. Then, we seek to prove that (i) $|f_i(\mathbf{x}) - g_i(\mathbf{x})| \leq \gamma_i$ for all $\mathbf{x} \in \mathcal{R}$, (ii) g_i is S_i-smooth, (iii) g_i is L_i-Lipschitz, and (iv) g_i is G_i-bounded, while bounding γ_i, S_i, L_i, G_i as functions of the previous parameters.

We consider different cases according to the type of node i:

- **output node.** In this case, the value and Lipschitz constant of node i is the same as of a node $j < i$. Thus, we have $|f_i(\mathbf{x}) - g_i(\mathbf{x})| \leq \gamma_j =: \gamma_i$ and obtain that $S_i = S_j$, $L_i = L_j$, and $G_i = G_j$.

- $\times$ **node.** In this case, there exists $j, k < i$ such that $f_i(\mathbf{x}) = f_j(\mathbf{x}) \cdot f_k(\mathbf{x})$ and $g_i(\mathbf{x}) = g_j(\mathbf{x}) \cdot g_k(\mathbf{x})$ which means that

$$\begin{aligned} |f_i(\mathbf{x}) - g_i(\mathbf{x})| &= |f_j(\mathbf{x}) \cdot f_k(\mathbf{x}) - g_j(\mathbf{x}) \cdot g_k(\mathbf{x})| \\ &\leq |f_j(\mathbf{x})| \cdot |f_k(\mathbf{x}) - g_k(\mathbf{x})| + |g_k(\mathbf{x})| \cdot |f_j(\mathbf{x}) - g_j(\mathbf{x})| \\ &\leq G_j \cdot \gamma_k + \gamma_j \cdot G_k =: \gamma_i \ . \end{aligned}$$

  Also, $S_i \leq S_j \cdot G_k + G_j \cdot S_k + 2L_j \cdot L_k$. The Lipschitz constant of g_i is upper bounded by $L_j \cdot G_k + G_j \cdot L_k$ which is equal to L_i by Assumption 5.1. Thus, g_i is L_i-Lipschitz. It is also easy to see that g_i is $G_i = G_j \cdot G_k$ bounded.

- $+$ **node.** In this case, there exist $j, k < i$ such that $f_i(\mathbf{x}) = f_j(\mathbf{x}) + f_k(\mathbf{x})$ and $g_i(\mathbf{x}) = g_j(\mathbf{x}) + g_k(\mathbf{x})$ which means that $|f_i(\mathbf{x}) - g_i(\mathbf{x})| \leq \gamma_j + \gamma_k =: \gamma_i$. Also, $S_i \leq S_j + S_k$. Then, the Lipschitz constant of g_i is upper bounded by $L_j + L_k$ which is equal to L_i by Assumption 5.1. Thus, g_i is L_i-Lipschitz and is also easy to see that it is $G_i = G_j + G_k$ bounded.

- relu **node.** In this case, there exists $j < i$ such that $f_i(\mathbf{x}) = \text{relu}(f_j(\mathbf{x}))$ and $g_i(\mathbf{x}) = \text{softrelu}(g_j(\mathbf{x}))$. Using Lemma 6.3, the triangle inequality and the fact that relu is 1-Lipschitz, we have $|f_i(\mathbf{x}) - g_i(\mathbf{x})| \leq \frac{a}{4} + \gamma_j =: \gamma_i$. The next is to bound the smoothness S_i. By definition, we have

$$\nabla g_i(\mathbf{x}) = \nabla \text{softrelu}_a(g_j(\mathbf{x})) = \text{softrelu}'_a(g_j(\mathbf{x})) \nabla g_j(\mathbf{x}) \ ,$$

  hence

$$\begin{aligned} \|\nabla g_i(\mathbf{x}) - \nabla g_i(\mathbf{y})\| &= \|\text{softrelu}'_a(g_j(\mathbf{x})) \nabla g_j(\mathbf{x}) - \text{softrelu}'_a(g_j(\mathbf{y})) \nabla g_j(\mathbf{y})\| \\ &\leq |\text{softrelu}'_a(g_j(\mathbf{x}))| \cdot \|\nabla g_j(\mathbf{x}) - \nabla g_j(\mathbf{y})\| \\ &\quad + |\text{softrelu}'_a(g_j(\mathbf{x})) - \text{softrelu}'_a(g_j(\mathbf{y}))| \cdot \|\nabla g_j(\mathbf{y})\| \\ &\leq (G_j \cdot S_j + \frac{1}{2a} L_j) \|\mathbf{x} - \mathbf{y}\|. \end{aligned}$$

  So, $S_i \leq G_j \cdot S_j + \frac{1}{2a} L_j$. Also, due to the fact that softrelu is 1-Lipschitz, the Lipschitzness of g_i is upper bounded by L_j which is equal to L_i by Assumption 5.1. Thus, g_i is L_i-Lipschitz. Finally, g_i is obviously G_j bounded and hence G_i bounded by Assumption 5.1.

Note that the sequence of errors $\{\gamma_i\}_{i\geq 1}$ is increasing. Going through all the cases we considered above, we see that $\gamma_i \leq \frac{a}{4} + G_k\gamma_j + G_j\gamma_k \leq \frac{1}{a} + 2G\gamma_{i-1}$ which implies that $\gamma_i \leq a \cdot (2G)^i$. Thus, we have

$$|f(\mathbf{x}) - g(\mathbf{x})| \leq a \cdot (2G)^{s(\mathcal{C})} \ .$$

Similarly, we have that $S_i \leq 2G \cdot S_{i-1} + 2L^2 + \frac{L}{a}$, so as long we set $\frac{1}{a}$ large enough compared to $2L^2$ (which will indeed be the case later on) we can simplify this to $S_i \leq 2G \cdot S_{i-1} + 2\frac{L}{a}$, hence $S_i \leq 2\frac{L}{a} \cdot (2G)^i$. Thus, we have

$$S_i \leq \frac{2L}{a} \cdot (2G)^{s(\mathcal{C})} \ .$$

Next, we proceed to prove the equivalence between the Goldstein stationary points of f and g. Towards this goal we introduce some notation: let $n = s(\mathcal{C})$, let $\mathcal{B}_i$ be all the bias variables that are used in the topological ordering of the neural circuit in nodes before the node i and b be the total number of bias variables, i.e., $b = |\mathcal{B}_n|$. We relate any (δ_i, ϵ_i)-Goldstein stationary of g_i to those of f_i through the following lemma.

Lemma 6.4 *Let $\mathbf{x} \in \mathcal{R}$. Then there exists two positive sequences $\delta_1, \ldots, \delta_n > 0$ and $\epsilon_1, \ldots, \epsilon_n > 0$, a sequence of vectors $\mathbf{s}_1, \ldots, \mathbf{s}_n$, and a sequence of distributions $\mathcal{D}_1^{\mathbf{x}}, \ldots, \mathcal{D}_n^{\mathbf{x}}$ supported on $[-a, a]^{s(\mathcal{C})}$ such that for all $i \in [s(\mathcal{C})]$ the following hold:*

- $\delta_i \leq (40 \cdot L \cdot G)^{s(\mathcal{C})} \cdot a, \quad \epsilon_i \leq (160 \cdot L \cdot G)^{3s(\mathcal{C})} \cdot a$ *and* $\|\mathbf{s}_i\|_2 \leq \epsilon_i$.
- $\nabla g_i(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim \mathcal{D}_i^{\mathbf{x}}}[\nabla f_i(\mathbf{y})] + \mathbf{s}_i$.
- *The support of $\mathcal{D}_i^{\mathbf{x}}$ has diameter δ_i and contains only points where f_k is differentiable for all k. Moreover, the support of $\mathcal{D}_i^{\mathbf{x}}$ only contains points $\mathbf{y}$ for which either $y_i = x_i$ for all i that correspond to input variables that are not biases, or are biases which are not used in the computation of f_i, g_i.*

Proof. We prove this lemma by induction on i. For the base of the induction we observe that all the input and constant nodes are in the beginning of the topological ordering, and satisfy $g_i = f_i$, and that f_i is differentiable. For this reason, input and constant nodes satisfy the lemma with $\mathcal{D}_i^{\mathbf{x}}$ equal to the Dirac delta distribution at $\mathbf{x}$, $\mathbf{s}_i = 0$, $\delta_i = 0$, $\epsilon_i = 0$. Now for the inductive step we assume that the lemma holds for all $j < i$ and we split into the following cases for the node i, depending on the type of the node in the circuit.

- **output node.** In this case, we have that $g_i = g_j$, $f_i = f_j$ hence the lemma follows immediately by the inductive hypothesis.
- **+ node.** In this case, we have that $g_i = g_j + g_k$, $f_i = f_j + f_k$. By the inductive hypothesis we get

  $$\nabla g_j(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim \mathcal{D}_j^{\mathbf{x}}}[\nabla f_j(\mathbf{y})] + \mathbf{s}_j \ , \qquad \nabla g_k(\mathbf{x}) = \mathbb{E}_{\mathbf{y} \sim \mathcal{D}_k^{\mathbf{x}}}[\nabla f_k(\mathbf{y})] + \mathbf{s}_k \ .$$

  Now let's assume without loss of generality that $j \geq k$, then we set $\mathcal{D}_i^{\mathbf{x}} = \mathcal{D}_j^{\mathbf{x}}$ and from the fact that $\mathcal{B}_j$ is a super set of $\mathcal{B}_k$ and from linearity of expectation we get

  $$\begin{aligned}\nabla g_i(\mathbf{x}) = \nabla g_j(\mathbf{x}) + \nabla g_k(\mathbf{x}) &= \mathbb{E}_{\mathbf{y} \sim \mathcal{D}_i^{\mathbf{x}}}[\nabla f_j(\mathbf{y}) + \nabla f_k(\mathbf{y})] + \mathbf{s}_j + \mathbf{s}_k \\ &= \mathbb{E}_{\mathbf{y} \sim \mathcal{D}_i^{\mathbf{x}}}[\nabla f_i(\mathbf{y})] + \mathbf{s}_j + \mathbf{s}_k \ ,\end{aligned}$$

  hence the lemma holds for this i with $\delta_i \leq \max\{\delta_j, \delta_k\}$, $\mathbf{s}_i = \mathbf{s}_j + \mathbf{s}_k$, $\epsilon_i = \epsilon_j + \epsilon_k$.

- $\times$ **node.** In this case, we have $g_i = g_j \cdot g_k$, $f_i = f_j \cdot f_k$ and by inductive hypothesis we get that

$$\nabla g_j(\mathbf{x}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[\nabla f_j(\mathbf{y})] + \mathbf{s}_j, \qquad \nabla g_k(\mathbf{x}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_k^{\mathbf{x}}}[\nabla f_k(\mathbf{y})] + \mathbf{s}_k.$$

Recall that g_i is differentiable, hence

$$\nabla g_i(\mathbf{x}) = g_j(\mathbf{x})\nabla g_k(\mathbf{x}) + g_k(\mathbf{x})\nabla g_j(\mathbf{x}) \ .$$

Also, because we will only consider points $\mathbf{x}$ for which all f_i's are also differentiable we have that

$$\nabla f_i(\mathbf{x}) = f_j(\mathbf{x})\nabla f_k(\mathbf{x}) + f_k(\mathbf{x})\nabla f_j(\mathbf{x}) \ .$$

Let's assume without loss of generality that $j \geq k$, then we set $\mathcal{D}_i^{\mathbf{x}} = \mathcal{D}_j^{\mathbf{x}}$ and from the fact that $\mathcal{B}_j$ is a super set of $\mathcal{B}_k$ we have that

$$\nabla g_j(\mathbf{x}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_j(\mathbf{y})] + \mathbf{s}_j \ , \qquad \nabla g_k(\mathbf{x}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_k(\mathbf{y})] + \mathbf{s}_k \ . \tag{6.3}$$

Using Eq. (6.3), the gradient of g_i and linearity of expectation we have

$$\begin{aligned}\nabla g_i(\mathbf{x}) &= g_j(\mathbf{x})\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_k(\mathbf{y})] + g_k(\mathbf{x})\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_k(\mathbf{y})] + g_k(\mathbf{x})\mathbf{s}_j + g_j(\mathbf{x})\mathbf{s}_k \\ &= \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[g_j(\mathbf{x})\nabla f_k(\mathbf{y}) + g_k(\mathbf{x})\nabla f_k(\mathbf{y})] + g_k(\mathbf{x})\mathbf{s}_j + g_j(\mathbf{x})\mathbf{s}_k \ . \end{aligned} \tag{6.4}$$

At this point we invoke Assumption 5.1 to utilize that g_j, g_k are G-bounded and that f_j, f_k are L-Lipschitz and hence we have that

$$\begin{aligned}\|\nabla g_i(\mathbf{x}) - \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_i(\mathbf{y})]\|_2 \leq L \cdot \left(\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|] + \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[|g_k(\mathbf{x}) - f_k(\mathbf{y})|]\right) \\ + G \cdot (\epsilon_j + \epsilon_k) \ . \end{aligned} \tag{6.5}$$

So it remains to bound $\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|]$ and $\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[|g_k(\mathbf{x}) - f_k(\mathbf{y})|]$. We will prove an upper bound on $\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|]$, while the same upper bound will work for for k as well. First, observe that because of the structure of the biases and the definition of $\mathcal{D}_i^{\mathbf{x}}$ we have that

$$\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|] = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|] \ .$$

Now we get that

$$\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|] \leq \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[|g_j(\mathbf{x}) - g_j(\mathbf{y})|] + \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[|g_j(\mathbf{y}) - f_j(\mathbf{y})|] \ . \tag{6.6}$$

We can now use the first statement of the theorem that we have already proved to recall that $|g_j(\mathbf{y}) - f_j(\mathbf{y})| \leq \gamma_j \leq (2G)^j \cdot a$, and we can also use the Lipschitz constant of g_j to get that

$$\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[|g_j(\mathbf{x}) - f_j(\mathbf{y})|] \leq L \cdot \delta_j + (2G)^i \cdot a \ . \tag{6.7}$$

Combining Eq. (6.5) and Eq. (6.7) we obtain

$$\|\nabla g_i(\mathbf{x}) - \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_i(\mathbf{y})]\|_2 \leq L^2 \cdot \delta_j + L \cdot (2G)^i \cdot a + G \cdot (\epsilon_j + \epsilon_k) \ ,$$

and the lemma follows for this case as well with $\delta_i \leq \max\{\delta_j, \delta_k\}$ and $\epsilon_i \leq L^2 \cdot \delta_j + L \cdot (2G)^i \cdot a + G \cdot (\epsilon_j + \epsilon_k)$.

- relu **node.** In this case we use the structure of the bias variables, and see that $g_i(\mathbf{x}) = \text{softrelu}(g_j(\mathbf{x}) + x_{b_i})$ and $f_i(\mathbf{x}) = \text{relu}(f_j(\mathbf{x}) + x_{b_i})$ where b_i is the index of the vector $\mathbf{x}$ that corresponds to the bias variable appearing only in the relu-node i. From the definition of softrelu we have that $g_i(\mathbf{x}) = \mathbb{E}_{u\sim U[-a,a]}[\text{relu}(g_j(\mathbf{x}) + x_{b_i} + u)]$. Using the fact that we only focus on $\mathbf{x}$ for which f_j is differentiable we get that

$$\begin{aligned}\nabla g_i(\mathbf{x}) &= \mathbb{E}_{u\sim U[-a,a]}[\mathbf{1}\{g_j(\mathbf{x}) + x_{b_i} + u \geq 0\}(\nabla(g_j(\mathbf{x}) + x_{b_i}))] \\ &= \mathbb{E}_{u\sim U[-a,a]}[\mathbf{1}\{g_j(\mathbf{x}) + x_{b_i} + u \geq 0\}](\nabla(g_j(\mathbf{x}) + x_{b_i})) \\ &= \mathbb{P}_{u\sim U[-a,a]}(g_j(\mathbf{x}) + x_{b_i} + u \geq 0)(\nabla(g_j(\mathbf{x}) + x_{b_i}))\ ,\end{aligned}$$

and also

$$\nabla f_i(\mathbf{x}) = \mathbf{1}\{f_j(\mathbf{x}) + x_{b_i} \geq 0\}(\nabla(f_j(\mathbf{x}) + x_{b_i}))\ .$$

From the inductive hypothesis and the fact that every bias variable appears only once we get

$$\nabla(g_j(\mathbf{x}) + x_{b_i}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla(f_j(\mathbf{y}) + x_{b_i})] + \mathbf{s}_j\ ,$$

so by denoting $\zeta_i(\mathbf{x}) = \mathbb{P}_{u\sim U[-a,a]}(g_j(\mathbf{x}) + x_{b_i} + u \geq 0)$ the above implies that

$$\nabla g_i(\mathbf{x}) = \zeta_i(\mathbf{x}) \cdot (\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla(f_j(\mathbf{y}) + x_{b_i})] + \mathbf{s}_j)\ .$$

We now need to distinguish several cases: (1) $g_j(\mathbf{x}) + x_{b_i} \geq a + \gamma_j + L\delta_j$, (2) $g_j(\mathbf{x}) + x_{b_i} \leq -a - \gamma_j - L\delta_j$, and (3) $|g_j(\mathbf{x}) + x_{b_i}| \leq a + \gamma_j + L\delta_j$.

We start with the first case. If $g_j(\mathbf{x}) + x_{b_i} \geq a + \gamma_j + L\delta_j$ then this means that $\zeta_i(\mathbf{x}) = 1$ and that $f_j(\mathbf{y}) + x_{b_i} \geq 0$ for all $\mathbf{y}$ that are δ_j-close to $\mathbf{x}$. This implies that we can choose $\mathcal{D}_i^{\mathbf{x}} = \mathcal{D}_j^{\mathbf{x}}$ and we immediately get

$$\nabla g_i(\mathbf{x}) = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_i(\mathbf{y})] + \mathbf{s}_j\ ,$$

hence the lemma holds with $\delta_i = \delta_j$, $\epsilon_i = \epsilon_j$ and $\mathbf{s}_i = \mathbf{s}_j$.

Similarly, for the second case we have that $\zeta_i(\mathbf{x}) = 0$ and $f_j(\mathbf{y}) + x_{b_i} \leq 0$ for all $\mathbf{y}$ that are δ_j-close to $\mathbf{x}$. In this case we have that

$$\nabla g_i(\mathbf{x}) = 0 = \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i\mathbf{x}}[\nabla f_i(\mathbf{y})]\ ,$$

hence the lemma holds with $\delta_i = \delta_j$, $\epsilon_i = \epsilon_j$ and $\mathbf{s}_i = 0$.

Finally, we consider the case $|g_j(\mathbf{x}) + x_{b_i}| \leq a + \gamma_j + L\delta_j$. This implies that $|f_j(\mathbf{y}) + x_{b_i}| \leq a + 2\gamma_j + 2L\delta_j$ for all $\mathbf{y}$ that are δ_j-close to $\mathbf{x}$. We define the distribution $\mathcal{D}_i^{\mathbf{x}}$ as follows: we first sample $\mathbf{y}$ from $\mathcal{D}_j^{\mathbf{x}}$, and let $\mathbf{y}_{-b_j}$ be the vector $\mathbf{y}$ with all the coordinates but b_j. We then sample y'_{b_j} from a distribution such that with probability $\zeta_i(\mathbf{x})$ it holds that $f(\mathbf{y}) + y'_{b_j} \geq \eta$ for some value $\eta > 0$ and with probability $1 - \zeta_i(\mathbf{x})$ it holds that $f(\mathbf{y}) + y'_{b_j} \leq -\eta$. We then observe the following: (a) by Rademacher's theorem we now that all f_j's are almost everywhere differentiable, so for an arbitrarily small value η and for every $\mathbf{y}$ we can find values y'_{b_j} such that what we want holds and also all the functions f_j are differentiable in $(\mathbf{y}_{-b_j}, y'_{b_j})$, and (b) the desired values y'_{b_j} are at most

$a + 2\gamma_j + 2L\delta_j$ away from x_{b_j}, hence at most $2a + 4\gamma_j + 4L\delta_j$ away from each other. Also, from the definition of $\mathcal{D}_i^{\mathbf{x}}$ we have that

$$\begin{aligned}
\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\nabla f_i(\mathbf{y})] &= \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_i^{\mathbf{x}}}[\mathbf{1}\{f_j(\mathbf{y}) + y_{b_i} \geq 0\}(\nabla f_j(\mathbf{y}) + \mathbf{e}_{b_i})] \\
&= \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[\mathbb{E}_{y'_{b_i}}[\mathbf{1}\{f_j(\mathbf{y}_{-b_i}) + y'_{b_i} \geq 0\}(\nabla f_j(\mathbf{y}) + \mathbf{e}_{b_i})]] \\
&= \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[\mathbb{E}_{y'_{b_i}}[\mathbf{1}\{f_j(\mathbf{y}_{-b_i}) + y'_{b_i} \geq 0\}](\nabla f_j(\mathbf{y}) + \mathbf{e}_{b_i})] \\
&= \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[\mathbb{P}_{y'_{b_i}}(f_j(\mathbf{y}_{-b_i}) + y'_{b_i} \geq 0)(\nabla f_j(\mathbf{y}) + \mathbf{e}_{b_i})] \\
&= \mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[\zeta_i(\mathbf{x})(\nabla f_j(\mathbf{y}) + \mathbf{e}_{b_i})] \\
&= \zeta_i(\mathbf{x})\mathbb{E}_{\mathbf{y}\sim\mathcal{D}_j^{\mathbf{x}}}[(\nabla f_j(\mathbf{y}) + \mathbf{e}_{b_i})] \\
&= \zeta_i(\mathbf{x})(\nabla g_j(\mathbf{y}) + \mathbf{e}_{b_i}) - \zeta_i(\mathbf{x})\mathbf{s}_j \\
&= \nabla g_i(\mathbf{x}) - \zeta_i(\mathbf{x})\mathbf{s}_j \ ,
\end{aligned}$$

where in the first line we used the definition of the distribution $\mathcal{D}_i^{\mathbf{x}}$, in the second line we use the fact that the bias variable y_{b_i} does not appear in the computation of f_j, in the fourth line we use the definition of the distribution of y'_{b_i} given $\mathbf{y}$, and in the rest we use the definition of ζ_i and our inductive hypothesis. Overall we get that the lemma follows with $\delta_i \leq \delta_j + 2a + 4\gamma_j + 4L\delta_j$, $\epsilon_i \leq \epsilon_j$, and $\mathbf{s}_i = \zeta_i(\mathbf{x}) \cdot \mathbf{s}_j$.

To conclude, using the fact that both δ_i and ϵ_i are increasing according to the definitions above and using the worst bounds from all these cases we get

$$\delta_i \leq \delta_{i-1}(4L + 1) + (8G)^{s(\mathcal{C})}a \ ,$$

implying that

$$\delta_n \leq (40 \cdot L \cdot G)^{s(\mathcal{C})} \cdot a \ .$$

Using this bound we can compute the worst possible bound for ϵ_i, as we have

$$\begin{aligned}
&\epsilon_i \leq L^2 \cdot \delta_j + L \cdot (2G)^i \cdot a + 2G \cdot \epsilon_{i-1} \\
\implies &\epsilon_i \leq (80 \cdot L \cdot G)^{2s(\mathcal{C})} \cdot a + 2G \cdot \epsilon_{i-1}
\end{aligned}$$

implying that

$$\epsilon_n \leq (160 \cdot L \cdot G)^{3s(\mathcal{C})} \cdot a \ ,$$

hence the lemma follows. □

Overall, for all the analyzed quantities we get

$$\begin{aligned}
\gamma_n &\leq a \cdot (2G)^{s(\mathcal{C})} \ , \\
\delta_n &\leq a \cdot (40LG)^{s(\mathcal{C})} \ , \\
\epsilon_n &\leq a \cdot (160LG)^{3s(\mathcal{C})} \ , \\
S_n &\leq \frac{2L}{a} \cdot (2G)^{s(\mathcal{C})} \ .
\end{aligned}$$

Setting $a = (160 \cdot L \cdot G)^{3s(\mathcal{C})} \cdot \gamma$ finishes the proof.

7 Conclusion

We have provided lower and upper bounds on the complexity of finding an approximate (δ, ϵ)-Goldstein stationary point of a Lipschitz function in deterministic nonsmooth and nonconvex optimization. We have shown that unlike dimension-free randomized algorithms, any deterministic first-order algorithm must suffer from a nontrivial dimension dependence, by establishing a lower bound of $\Omega(d)$ for any dimension d, whenever $\delta, \epsilon > 0$ are smaller than given constants. Furthermore, we established the importance of a zeroth-order oracle in deterministic nonsmooth nonconvex optimization, by proving that any deterministic algorithm that uses only a gradient oracle cannot guarantee to return an adequate point within any finite time. Both lower bounds stand in contrast to randomized algorithms, as well as deterministic smooth nonconvex and nonsmooth convex settings, emphasizing the unique difficulty of nonsmooth nonconvex optimization.

We have also provided a deterministic algorithm that achieves the best known dimension-free rate with merely a logarithmic smoothness dependence, allowing de-randomization for slightly-smooth functions. This motivated the study of deterministic smoothings, in order to apply our algorithm for nonsmooth problems. We proved that unlike existing randomized smoothings, no efficient deterministic black-box smoothing can provide any meaningful guarantees, providing an answer to an open question raised in the literature. Moreover, we have bypassed this impossibility result in a practical white-box model, providing a deterministic smoothing for a wide variety of widely used neural network architectures which is provably meaningful from an optimization viewpoint. Combined with our algorithm, this yields the first deterministic, dimension-free algorithm for optimizing such networks, circumventing our lower bound.

As to future directions, it is interesting to note that our lower bound for deterministic first-order optimization is linear with respect to the dimension, though we are not aware of any such algorithm with sub-exponential dimension dependence (namely, better than exhaustive grid-search). Therefore, we pose the following question:

> **Open problem:** *Is there a deterministic first-order algorithm for nonsmooth nonconvex optimization that returns a (δ, ϵ)-Goldstein stationary point using $\mathrm{poly}(d, \delta^{-1}, \epsilon^{-1})$ oracle calls?*

Acknowledgments

MJ and TL were supported in part by the Mathematical Data Science program of the Office of Naval Research under grant number N00014-18-1-2764 and by the Vannevar Bush Faculty Fellowship program under grant number N00014-21-1-2941. MZ was supported by the Army Research Office (ARO) under contract W911NF-17-1-0304 as part of the collaboration between US DOD, UK MOD and UK Engineering and Physical Research Council (EPSRC) under the Multidisciplinary University Research Initiative (MURI). GK and OS were supported in part by the European Research Council (ERC) grant 754705, and by an Israeli Council for Higher Education grant via the Weizmann Data Science Research Center.

References

Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, A. Sekhari, and K. Sridharan. Second-order information in non-convex stochastic optimization: Power and limitations. In *COLT*, pages 242–299. PMLR, 2020. (Cited on page 27.)

Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, N. Srebro, and B. Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, pages 1–50, 2022. (Cited on page 27.)

H. Attouch and D. Aze. Approximation and regularization of arbitrary functions in Hilbert spaces by the Lasry-Lions method. *Annales de l'Institut Henri Poincaré C, Analyse non linéaire*, 10(3): 289–312, 1993. (Cited on page 27.)

A. Beck and M. Teboulle. Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization*, 22(2):557–580, 2012. (Cited on page 27.)

M. Benaïm, J. Hofbauer, and S. Sorin. Stochastic approximations and differential inclusions. *SIAM Journal on Control and Optimization*, 44(1):328–348, 2005. (Cited on page 26.)

J. Bolte and E. Pauwels. Conservative set valued fields, automatic differentiation, stochastic gradient methods and deep learning. *Mathematical Programming*, 188(1):19–51, 2021. (Cited on pages 26 and 27.)

G. Braun, C. Guzmán, and S. Pokutta. Lower bounds on the oracle complexity of nonsmooth convex optimization via information theory. *IEEE Transactions on Information Theory*, 63(7):4709–4724, 2017. (Cited on page 27.)

J. V. Burke, A. S. Lewis, and M. L. Overton. Approximating subdifferentials by random sampling of gradients. *Mathematics of Operations Research*, 27(3):567–584, 2002a. (Cited on page 26.)

J. V. Burke, A. S. Lewis, and M. L. Overton. Two numerical methods for optimizing matrix stability. *Linear Algebra and its Applications*, 351:117–145, 2002b. (Cited on page 26.)

J. V. Burke, A. S. Lewis, and M. L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM Journal on Optimization*, 15(3):751–779, 2005. (Cited on page 26.)

J. V. Burke, F. E. Curtis, A. S. Lewis, M. L. Overton, and L. E. A. Simões. Gradient sampling methods for nonsmooth optimization. *Numerical Nonsmooth Optimization: State of the Art Algorithms*, pages 201–225, 2020. (Cited on pages 2 and 26.)

Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points I. *Mathematical Programming*, 184(1):71–120, 2020. (Cited on page 27.)

Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points II: first-order methods. *Mathematical Programming*, 185(1):315–355, 2021. (Cited on pages 2 and 27.)

C. Cartis, N. I. M. Gould, and P. L. Toint. 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. (Cited on page 27.)

C. Cartis, N. I. M. Gould, and P. L. Toint. Complexity bounds for second-order optimality in unconstrained optimization. *Journal of Complexity*, 28(1):93–108, 2012. (Cited on page 27.)

C. Cartis, N. I. M. Gould, and P. L. Toint. Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization. In *Proceedings of the International Congress of Mathematicians: Rio de Janeiro*, pages 3711–3750. World Scientific, 2018. (Cited on page 27.)

X. Chen. Smoothing methods for nonsmooth, nonconvex minimization. *Mathematical Programming*, 134(1):71–99, 2012. (Cited on page 27.)

F. H. Clarke. Necessary conditions for nonsmooth variational problems. In *Optimal Control Theory and Its Applications*, pages 70–91. Springer, 1974. (Cited on page 2.)

F. H. Clarke. Generalized gradients and applications. *Transactions of the American Mathematical Society*, 205:247–262, 1975. (Cited on page 2.)

F. H. Clarke. Generalized gradients of Lipschitz functionals. *Advances in Mathematics*, 40(1):52–67, 1981. (Cited on page 2.)

F. H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990. (Cited on pages 1 and 3.)

F. H. Clarke, Y. S. Ledyaev, R. J. Stern, and P. R. Wolenski. *Nonsmooth Analysis and Control Theory*, volume 178. Springer Science & Business Media, 2008. (Cited on page 2.)

A. Daniilidis and D. Drusvyatskiy. Pathological subgradient dynamics. *SIAM Journal on Optimization*, 30(2):1327–1338, 2020. (Cited on pages 2 and 26.)

C. Daskalakis and C. Papadimitriou. Continuous local search. In *SODA*, pages 790–804. SIAM, 2011. (Cited on page 8.)

D. Davis and D. Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019. (Cited on page 8.)

D. Davis, D. Drusvyatskiy, S. Kakade, and J. D. Lee. Stochastic subgradient method converges on tame functions. *Foundations of Computational Mathematics*, 20(1):119–154, 2020. (Cited on pages 26 and 27.)

D. Davis, D. Drusvyatskiy, Y. T. Lee, S. Padmanabhan, and G. Ye. A gradient sampling method with complexity guarantees for Lipschitz functions in high and low dimensions. In *NeurIPS*, 2022. (Cited on pages 2, 4, 6, and 27.)

J. C. Duchi, P. L. Bartlett, and M. J. Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012. (Cited on pages 7 and 27.)

J. Fearnley, P. W. Goldberg, A. Hollender, and R. Savani. The complexity of gradient descent: CLS = PPAD ∩ PLS. In *STOC*, pages 46–59, 2021. (Cited on page 8.)

S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013. (Cited on page 5.)

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *AISTATS*, pages 315–323, 2011. (Cited on page 1.)

A. Goldstein. Optimization of Lipschitz continuous functions. *Mathematical Programming*, 13(1):14–22, 1977. (Cited on pages 2, 4, and 6.)

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. (Cited on page 8.)

C. Guzmán and A. Nemirovski. On lower complexity bounds for large-scale smooth convex optimization. *Journal of Complexity*, 31(1):1–14, 2015. (Cited on page 27.)

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456. PMLR, 2015. (Cited on page 11.)

K. C. Kiwiel. Restricted step and Levenberg-Marquardt techniques in proximal bundle methods for nonconvex nondifferentiable optimization. *SIAM Journal on Optimization*, 6(1):227–249, 1996. (Cited on page 26.)

K. C. Kiwiel. Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization. *SIAM Journal on Optimization*, 18(2):379–388, 2007. (Cited on page 26.)

G. Kornowski and O. Shamir. Oracle complexity in nonsmooth nonconvex optimization. In *NeurIPS*, pages 324–334, 2021. (Cited on pages 2, 3, 7, 8, and 27.)

G. Kornowski and O. Shamir. Oracle complexity in nonsmooth nonconvex optimization. *Journal of Machine Learning Research*, 23(314):1–44, 2022. (Cited on page 27.)

J-M. Lasry and P-L. Lions. A remark on regularization in Hilbert spaces. *Israel Journal of Mathematics*, 55(3):257–266, 1986. (Cited on page 27.)

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. (Cited on page 8.)

T. Lin, Z. Zheng, and M. I. Jordan. Gradient-free methods for deterministic and stochastic nonsmooth nonconvex optimization. In *NeurIPS*, 2022. (Cited on page 5.)

M. M. Mäkelä and P. Neittaanmäki. *Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control*. World Scientific, 1992. (Cited on page 1.)

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. (Cited on page 11.)

K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39(2):117–129, 1987. (Cited on page 2.)

V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *ICML*, pages 807–814, 2010. (Cited on page 1.)

A. S. Nemirovski and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience, 1983. (Cited on pages 2, 4, and 27.)

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005. (Cited on page 27.)

Y. Nesterov. How to make the gradients small. *Optima. Mathematical Optimization Society Newsletter*, 88:10–11, 2012. (Cited on page 27.)

Y. Nesterov. *Lectures on Convex Optimization*, volume 137. Springer, 2018. (Cited on pages 2, 4, and 27.)

J. Outrata, M. Kocvara, J. Zowe, and J. Zowe. *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints: Theory, Applications and Numerical Results*, volume 28. Springer Science & Business Media, 1998. (Cited on page 1.)

R. T. Rockafellar and R. J-B. Wets. *Variational Analysis*, volume 317. Springer Science & Business Media, 2009. (Cited on pages 2 and 27.)

G. I. Shamir, D. Lin, and L. Coviello. Smooth activations and reproducibility in deep networks. *arXiv preprint arXiv:2010.09931*, 2020. (Cited on page 11.)

N. Tatro, P. Chen, P. Das, I. Melnyk, P. Sattigeri, and R. Lai. Optimizing mode connectivity via neuron alignment. *Advances in Neural Information Processing Systems*, 33:15300–15311, 2020. (Cited on page 11.)

L. Tian and A. M-C. So. No dimension-free deterministic algorithm computes approximate stationarities of Lipschitzians. *ArXiv Preprint: 2210.06907*, 2022. (Cited on page 3.)

L. Tian, K. Zhou, and A. M-C. So. On the finite-time complexity and practical computation of approximate stationarity concepts of Lipschitz functions. In *ICML*, pages 21360–21379. PMLR, 2022. (Cited on pages 2, 4, 6, and 27.)

S. A. Vavasis. Black-box complexity of local minimization. *SIAM Journal on Optimization*, 3(1):60–80, 1993. (Cited on page 27.)

J. Zhang, H. Lin, S. Jegelka, S. Sra, and A. Jadbabaie. Complexity of finding stationary points of nonconvex nonsmooth functions. In *ICML*, pages 11173–11182. PMLR, 2020. (Cited on pages 2, 4, 6, 8, and 27.)

A Related Work

To appreciate the difficulty and the scope of research agenda in nonsmooth nonconvex optimization, we describe the relevant literature. In this context, existing research is mostly devoted to establishing the asymptotic convergence of optimization algorithms, including the gradient sampling (GS) method [Burke et al., 2002a,b, 2005, Kiwiel, 2007, Burke et al., 2020], bundle methods [Kiwiel, 1996] and subgradient methods [Benaïm et al., 2005, Davis et al., 2020, Daniilidis and Drusvyatskiy, 2020, Bolte and Pauwels, 2021]. More specifically, Burke et al. [2002a] provided the systematic investigation of approximating a generalized gradient through a simple yet novel random sampling scheme, motivating the subsequent development of celebrated gradient bundle method [Burke et al., 2002b]. Then, Burke et al. [2005] and Kiwiel [2007] proposed the modern GS method by incorporating key modifications into the scheme of the aforementioned gradient bundle method and proved that any cluster point of the iterates generated by the GS method is a Clarke stationary point. For an overview of GS methods, we refer to Burke et al. [2020].

There has been recent progress in the investigation of different subgradient methods for nonsmooth nonconvex optimization. It was shown by Daniilidis and Drusvyatskiy [2020] that the standard subgradient method fails to find any Clarke stationary point of a Lipschitz function, as witnessed by the existence of pathological examples. Benaïm et al. [2005] established the asymptotic convergence guarantee of

stochastic approximation methods from a differential inclusion point of view under additional conditions and Bolte and Pauwels [2021] justified *automatic differentiation* as used in deep learning. Davis et al. [2020] proved the asymptotic convergence of subgradient methods if the objective function is assumed to be Whitney stratifiable. Turning to nonasymptotic convergence guarantee, Zhang et al. [2020] proposed a randomized variant of Goldstein's subgradient method and proved a dimension-independent complexity bound of $\widetilde{O}(\delta^{-1}\epsilon^{-3})$ for finding a (δ, ϵ)-Goldstein stationary point of a Hadamard directionally differentiable function. For the more broad class of Lipschitz functions, Davis et al. [2022] and Tian et al. [2022] have proposed two other randomized variants of Goldstein's subgradient method and proved the same complexity guarantee. Comparing to their randomized counterparts, deterministic algorithms are relatively scarce in nonsmooth nonconvex optimization.

In convex optimization, we have a deep understanding of the complexity of finding an ϵ-optimal point (i.e., $\mathbf{x} \in \mathbb{R}^d$ such that $f(\mathbf{x}) - \inf_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) \leq \epsilon$) [Nemirovski and Yudin, 1983, Guzmán and Nemirovski, 2015, Braun et al., 2017, Nesterov, 2018]. In smooth nonconvex optimization, various lower bounds have been established for finding an ϵ-stationary point (i.e., $\mathbf{x} \in \mathbb{R}^d$ such that $\|\nabla f(\mathbf{x})\| \leq \epsilon$) [Vavasis, 1993, Nesterov, 2012, Carmon et al., 2020, 2021]. Further extensions to nonconvex stochastic optimization were given in Arjevani et al. [2020, 2022] while algorithm-specific lower bounds for finding an ϵ-stationary point were derived in Cartis et al. [2010, 2012, 2018]. However, these proof techniques can not be extended to nonsmooth nonconvex optimization due to different optimality notions. In this vein, Zhang et al. [2020] and Kornowski and Shamir [2021] have demonstrated that neither an ϵ-Clarke stationary point nor a near ϵ-Clarke stationary point can be obtained in a $\text{poly}(d, \epsilon^{-1})$ number of queries when $\epsilon > 0$ is smaller than some constant. Our analysis is inspired by their construction and techniques but focus on establishing lower bounds for finding a (δ, ϵ)-Goldstein stationary point.

The smoothing viewpoint starts with Rockafellar and Wets [2009, Theorem 9.7], which states that any approximate Clarke stationary point of a Lipschitz function is the asymptotic limit of appropriate approximate stationary points of smooth functions. In particular, given a Lipschitz function f, we can attempt to construct a smooth function $\tilde{f}$ that is δ-close to f (i.e., $\|f - g\|_\infty \leq \delta$), and apply a smooth optimization algorithm on $\tilde{f}$. Such smoothing approaches have been used in convex optimization [Nesterov, 2005, Beck and Teboulle, 2012] and found the application in structured nonconvex optimization [Chen, 2012]. For a general Lipschitz function, Duchi et al. [2012] proposed a randomized smoothing approach that can transform the original problem to a smooth nonconvex optimization where the objective function is given in the expectation form and the smoothness parameter is dimension-dependent. Moreover, there are deterministic smoothing approaches that yield dimension-independent smoothness parameters but they are computationally intractable [Lasry and Lions, 1986, Attouch and Aze, 1993]. Recently, Kornowski and Shamir [2021, 2022] have explored the trade-off between computational tractability and smoothing, ruling out the existence of any (possibly randomized) smoothing approach that achieves computational tractability and a dimension-independent smoothness parameter.