The Adaptive Complexity of Finding a Stationary Point

Huanjian Zhou ZHOU@MS.K.U-TOKYO.AC.JP
University of Tokyo, RIKEN AIP

Andi Han ANDI.HAN@SYDNEY.EDU.AU
RIKEN AIP, University of Sydney

Akiko Takeda TAKEDA@MIST.I.U-TOKYO.AC.JP
University of Tokyo, RIKEN AIP

Masashi Sugiyama SUGI@K.U-TOKYO.AC.JP
RIKEN AIP, University of Tokyo

Editors: Nika Haghtalab and Ankur Moitra

Abstract

In large-scale applications, such as machine learning, it is desirable to design non-convex optimization algorithms with a high degree of parallelization. In this work, we study the adaptive complexity of finding a stationary point, which is the minimal number of sequential rounds required to achieve stationarity given polynomially many queries executed in parallel at each round.

For the high-dimensional case, *i.e.*, $d = \widetilde{\Omega}(\varepsilon^{-(2+2p)/p})$, we show that for any (potentially randomized) algorithm, there exists a function with Lipschitz p-th order derivatives such that the algorithm requires at least $\varepsilon^{-(p+1)/p}$ iterations to find an ε-stationary point. Our lower bounds are tight and show that even with $\mathrm{poly}(d)$ queries per iteration, no algorithm has better convergence rate than those achievable with one-query-per-round algorithms. In other words, gradient descent, the cubic-regularized Newton's method, and the p-th order adaptive regularization method are adaptively optimal. Our proof relies upon novel analysis with the characterization of the output for the hardness potentials based on a chain-like structure with random partition.

For the constant-dimensional case, *i.e.*, $d = \Theta(1)$, we propose an algorithm that bridges grid search and gradient flow trapping, finding an approximate stationary point in constant iterations. Its asymptotic tightness is verified by a new lower bound on the required queries per iteration. We show there exists a smooth function such that any algorithm running with $\Theta(\log(1/\varepsilon))$ rounds requires at least $\widetilde{\Omega}((1/\varepsilon)^{(d-1)/2})$ queries per round. This lower bound is tight up to a logarithmic factor, and implies that the gradient flow trapping is adaptively optimal.

Keywords: Non-convex optimization, Stationary points, Adaptive complexity

1. Introduction

Let $f : \mathbb{R}^d \to \mathbb{R}$ be a smooth function (*i.e.*, the map $\boldsymbol{x} \mapsto \nabla f(\boldsymbol{x})$ is Lipschitz, and f is possibly non-convex). The problem of minimizing f is one of the most fundamental problems across a wide range of scientific and engineering disciplines, with particular importance in modern machine learning (Jain et al., 2017; Bottou et al., 2018). Generally, without additional structured assumptions on f, it is $\mathcal{NP}$-hard to find approximate global minima or even test if a point is a local minimum (Nemirovskij and Yudin, 1983) or a high-order saddle point (Murty and Kabadi, 1985). As an alternative measure of optimization convergence, we study the problem of finding an ε-approximate stationary point (for $\varepsilon > 0$), *i.e.*, a point $\boldsymbol{x} \in \mathbb{R}^d$ such that

$$\|\nabla f(\boldsymbol{x})\| \le \varepsilon.$$

This is motivated by a line of research that identifies sub-classes of non-convex problems for which all (first-order or second-order) stationary points are globally optimal (Choromanska et al., 2015; Ge et al., 2016; Kawaguchi, 2016; Ge et al., 2015; Sun et al., 2018; Ma et al., 2018; Allen-Zhu et al., 2019). In addition, practically efficient (gradient-based) algorithms have been developed to find stationary points (Kingma, 2014; Jin et al., 2017; Zaheer et al., 2018; Fang et al., 2018). Furthermore, significant progress has been made in developing sequential algorithms for this problem and proving non-asymptotic convergence rates (Cartis and Roberts, 2023; Nesterov and Polyak, 2006; Nesterov, 2013; Birgin et al., 2017; Cartis et al., 2020b,a).

The algorithms underlying the above results are highly sequential and fail to fully exploit contemporary parallel computing resources such as multi-core central processing units (CPUs) and many-core graphics processing units (GPUs). Nevertheless, given the widespread application of non-convex optimization in machine learning and the continual growth in dataset sizes (Bottou et al., 2018), there is a persistent need to accelerate non-convex optimization through parallelization (Dean et al., 2012; You et al., 2017; Recht et al., 2011; You et al., 2020).

A convenient metric for parallelization in black-box oracle models is *adaptivity*, which is the number of sequential rounds it makes when each round can execute polynomial independent queries in parallel. Over the past several years, there have been breakthroughs in the study of adaptivity in various problems including sorting (Valiant, 1975; Cole, 1988; Braverman et al., 2016), multi-armed bandits (Agarwal et al., 2017), property testing (Canonne and Gur, 2018; Chen et al., 2018), submodular optimization (Balkanski and Singer, 2018a; Chakrabarty et al., 2024; Li et al., 2020), convex optimization (Balkanski and Singer, 2018b; Diakonikolas and Guzmán, 2019; Bubeck et al., 2019; Carmon et al., 2023), and log-concave sampling (Zhou and Sugiyama, 2024; Anari et al., 2024; Zhou et al., 2024).

While the adaptive complexity of convex optimization has been extensively studied—most notably with (Balkanski and Singer, 2018b) demonstrating that parallelization does not provide acceleration for convex optimization—the adaptive complexity of *non-convex* optimization has not been explored. This gap motivates our investigation into the question:

Whether parallelization can fundamentally accelerate non-convex optimization in both constant- and high-dimensional settings?

1.1. Our results

In this paper, we make significant progress by establishing new lower bounds for the high-dimensional setting, as well as both upper and lower bounds in the constant-dimensional case. Notably, our lower bounds either match the best known upper bounds or match our proposed new upper bounds up to logarithmic factors. As a result, we obtain the first *tight* adaptive complexity characterizations for finding stationary points.

Lower bounds in high dimension. We derive the following adaptive complexity in the high-dimensional setting $d = \widetilde{\Omega}\left(\varepsilon^{-(2+2p)/p}\right)$.

Theorem 1 (informal, see Theorem 4) *For $d = \widetilde{\Omega}\left(\varepsilon^{-(2+2p)/p}\right)$ and p-th order smooth function, any randomized optimizer needs $\Omega(\varepsilon^{-(1+p)/p})$ sequential rounds to find ε-stationary points even with $\mathsf{poly}(d)$ queries per round.*

Table 1: Comparisons of the state-of-the-art algorithms and our lower bounds. When $d = \Theta(1)$, we focus on the optimization over cube constraint $[0,1]^d$. When $d = \widetilde{\Omega}\left(\varepsilon^{-\frac{2+2p}{p}}\right)$, we consider p-th order smoothness and assume access to a p-th order oracle. Here, $\widetilde{\Omega}$ omits logarithmic factors in the parameter $1/\varepsilon$.

Problem setting		Adaptive complexity	Queries per iteration
(constant) $d = \Theta(1)$ $(d \geq 2)$	Upper bounds	$k = \Theta(1)$ (Theorem 10)	$\mathcal{O}\left(\varepsilon^{-\frac{(d+1)^k}{(2d)^k-(d+1)^k}\cdot\frac{d-1}{2}}\right)$
	Lower bounds	$k = \Theta(1)$ (Theorem 11)	$\widetilde{\Omega}\left(\varepsilon^{-\frac{d^k}{d^k-1}\cdot\frac{d-1}{2}}\right)$
	Upper bounds	$\Theta\left(\log\left(\frac{1}{\varepsilon}\right)\right)$ (Hollender and Zampetakis, 2023)	$\mathcal{O}\left(\varepsilon^{-(d-1)/2}\right)$
	Lower bounds	$\Theta(\log(\frac{1}{\varepsilon}))$ (Theorem 11)	$\widetilde{\Omega}\left(\varepsilon^{-(d-1)/2}\right)$
(high-dim.) $d = \widetilde{\Omega}\left(\varepsilon^{-\frac{2+2p}{p}}\right)$	Upper bounds	$\mathcal{O}\left(\varepsilon^{-\frac{1+p}{p}}\right)$ (Birgin et al., 2017)	1
	Lower bounds	$\Omega\left(\varepsilon^{-\frac{1+p}{p}}\right)$ (Carmon et al., 2020)	1
	Lower bounds	$\Omega\left(\varepsilon^{-\frac{1+p}{p}}\right)$ (Theorem 4)	$\mathsf{poly}(d)$

Our result significantly strengthens the lower bound established for one-query-per-round algorithms in (Carmon et al., 2020), even with only logarithmic scaling in the dimension dependence. Specifically, for $p = 1$, our lower bound implies that gradient descent (Nesterov, 2012) is adaptive optimal among all methods (even randomized, high-order, parallel methods) operating on functions with Lipschitz continuous gradient and bounded initial sub-optimality. Similarly, in the case $p = 2$, our result shows that the cubic regularization of Newton's method (Nesterov and Polyak, 2006; Cartis et al., 2010) is adaptive optimal and for general p, the p-th order adaptive regularization (ARp) algorithm (Birgin et al., 2017; Cartis et al., 2020a,b) is adaptive optimal. Consequently, our result highlights that *parallelization offers no acceleration* for high-dimensional non-convex optimization.

Our result can be also viewed as a specific setting of stochastic non-convex optimization with zero variance, (*i.e.*, $\sigma = 0$). Combining with the lower bound for stochastic non-convex optimization with $\sigma > 0$, $\Omega(\sigma^2\varepsilon^{-4})$ (Arjevani et al., 2023), we can obtain a lower bound $\Omega(\varepsilon^{-2} + \sigma^2\varepsilon^{-4})$ matching the query complexity by stochastic gradient descent (SGD), *i.e.*, SGD is also adaptive optimal for stochastic non-convex optimization. We also note the construction in (Arjevani et al., 2023) cannot directly apply to our setting (see more discussions in Section 1.2).

Lower and upper bounds in constant dimension. Following (Bubeck and Mikulincer, 2020), we study the constraint set $[0,1]^d$ $(d \geq 2)$ for ease of exposition. For constant dimension d, the naïve grid search only requires a single round by querying $O(\varepsilon^{-d})$ points on an $O(\varepsilon)$-net of $[0,1]^d$. The first non-trivial improvement was proposed in (Bubeck et al., 2019), where they developed an algorithm running within $O(\mathsf{poly}(d)\log(\frac{1}{\varepsilon}))$ iterations and issuing $O(\mathsf{poly}(d, \log(\frac{1}{\varepsilon})) \cdot \varepsilon^{-(d-1)/2})$ queries per iteration. The number of queries per iteration has been later improved to $O(\mathsf{poly}(d) \cdot \varepsilon^{-(d-1)/2})$ by an algorithm called gradient flow parallel trap (GFPT) (Hollender and Zampetakis, 2023). In this work, we bridge grid search (1-round) and GFPT ($\Theta(\log(\frac{1}{\varepsilon}))$-round) by establishing the following upper bound on the query complexity for constant-round algorithms.

Theorem 2 (informal, see Theorem 10) *For $d = \Theta(1)$ (with $d \geq 2$) and any Lipschitz smooth function, there exists an algorithm running within $k = \Theta(1)$ sequential rounds to find ε-stationary points with $\varepsilon^{-\frac{d-1}{2}(1+\mathcal{O}(2^{-k}))}$ queries per iteration.*

This is the *first* result for constant-dimension and constant-iteration settings. For $k = 1$, our result recovers the ε^{-d} complexity of grid search. As k increases, our complexity approaches $\mathcal{O}(\varepsilon^{-\frac{d-1}{2}})$ exponentially fast in the exponent and becomes numerically close to the complexity of GFPT after a moderate number of iterations. Because the query complexity is non-increasing with the number of rounds, for $\omega(1) \leq k \leq \mathcal{O}(\log(\frac{1}{\varepsilon}))$, the number of required queries per round lies in $(\varepsilon^{-\frac{d-1}{2}}, \varepsilon^{-\frac{d-1}{2}(1+\mathcal{O}(2^{-k}))})$ which remains $\mathsf{poly}(\frac{1}{\varepsilon})$.

We also complement our upper bound in constant rounds by showing new lower bounds for any algorithm running within $\mathcal{O}(\log(\frac{1}{\varepsilon}))$ iterations.

Theorem 3 (informal, see Theorem 11) *For $d = \Theta(1)$ and any Lipschitz smooth function, to find ε-stationary points, any randomized optimizer running within $k = \mathcal{O}(\log(\frac{1}{\varepsilon}))$ needs at least $\widetilde{\Omega}\left(\varepsilon^{-\frac{d-1}{2}(1+\mathcal{O}(d^{-k}))}\right)$ queries per round.*

This lower bound asymptotically matches convergence rate of our algorithm as iteration $k \to \infty$. Our established lower bound also matches the upper bound of GFPT (Hollender and Zampetakis, 2023) with $\Theta(\log(\frac{1}{\varepsilon}))$ iterations. Our results partially address the question 3 in (Bubeck and Mikulincer, 2020) by showing $\mathsf{poly}(\frac{1}{\varepsilon})$ complexity of $\mathcal{O}(\log(\frac{1}{\varepsilon}))$-iterations algorithm, *i.e.*, the low-depth region is at least $\Omega(\log(\frac{1}{\varepsilon}))$.

1.2. Technical overview

In this section we summarize the main technical ideas used to prove our lower bounds. For details, see Section 3 for Theorem 1 and Section 4 for Theorem 2. Notably, the high-dimensional lower bound relies on concentration inequalities specific to high dimensions, whereas the constant-dimensional lower bound is based on a reduction whose cost remains constant only when the dimension itself is fixed.

1.2.1. HIGH-DIMENSIONAL CASE

Existing works (Woodworth and Srebro, 2017; Carmon et al., 2020, 2021; Yue et al., 2023; Kwon et al., 2024; Zhang et al., 2022; Arjevani et al., 2023) that proved lower bounds for continuous optimization are based on a family of hard functions, parameterized by random orthogonal vectors $(\boldsymbol{u}^1, \dots, \boldsymbol{u}^r)$, where r is the number of rounds. A crucial characteristic of such construction is that no algorithm can access information about the remaining vectors $\boldsymbol{u}^{i+1}, \dots, \boldsymbol{u}^r$ at round i, with high probability. Moreover, obtaining an approximate stationary point critically requires the knowledge of the final vector $\boldsymbol{u}^r$.

To derive our adaptive lower bounds, we follow a similar chaining-like construction. Nonetheless, compared to prior works, we design a family of hard functions against a *polynomial* number of parallel queries, which is the main technical innovation of this paper. Our new construction is based on *random partition* used in exploring adaptive complexity of log-concave sampling and submodular optimization (Zhou et al., 2024; Chakrabarty et al., 2022; Li et al., 2020). Such random partition based chaining structure allows us to simplify the proof and elegantly establish the adaptive complexity for non-convex optimization.

Prior arguement fails under polynomial queries. We first revisit the construction in (Carmon et al., 2020) and highlight why their approach fails to address cases involving a polynomial number of queries. Inspired by chaining-like quadratic functions for smooth convex optimization (Nesterov, 2013), their work designed a robust chaining-like function parameterized by an orthogonal matrix $\boldsymbol{U} = (\boldsymbol{u}^1, \dots, \boldsymbol{u}^r)$ such that (i) $\|\nabla f(\boldsymbol{U}\boldsymbol{x})\|$ is non-vanishing unless $|\langle \boldsymbol{u}^i, \boldsymbol{x}\rangle| > 1$ for all $i \in \{1, 2, ..., r\}$; (ii) the gradient $\nabla f(\boldsymbol{U}\boldsymbol{x})$ will be independent of $\boldsymbol{u}^{i+1}, \dots, \boldsymbol{u}^r$ if $|\langle \boldsymbol{u}^j, \boldsymbol{x}\rangle| < 1/2$, for $j \geq i+1$; (iii) without $\boldsymbol{u}^i$ being discovered, $|\langle \boldsymbol{u}^i, \boldsymbol{x}\rangle| < 1/2$ with high probability. Thus, any algorithms can discover at most one random vector in each iteration, and thus cannot reach an approximate stationary point without iterating r times.

Properties (i) and (ii) rely on the inherent structure of the chaining-like function, while the remaining objective is to establish property (iii) using random orthogonal vectors. To prove (iii), they analyzed the behavior at iteration t. Let $\widehat{\boldsymbol{u}}^j$ denote the projection of the future vector $\boldsymbol{u}^j$ with $j > t$ onto the orthogonal complement of the space $\operatorname{span}\{\boldsymbol{u}^1, \dots, \boldsymbol{u}^t, \boldsymbol{x}^1, \dots, \boldsymbol{x}^t\}$, where $\boldsymbol{x}^1, ..., \boldsymbol{x}^t$ denote the iterates up to t-th iteration. They showed that, conditioned on $\boldsymbol{u}^1, \dots, \boldsymbol{u}^t$, the random variable $\widehat{\boldsymbol{u}}^j$ follows a rotationally symmetric distribution and is independent of $\boldsymbol{x}^{t+1}$. Using a subspace concentration and an inductive argument (Ball et al., 1997), they verified that property (iii) holds. However, this relies on the projection, and when a set of queries is issued as a basis for the entire space (which can be done with polynomial queries), the argument fails to hold.

Our construction by random partition. Instead of using random projections on the space spanned by the discovered random orthogonal vectors and queried vectors, we adopt a specific random projection parameterized by a random partition over all coordinates. Specifically, we consider random partitions $\mathcal{P} = (P_1, \dots, P_{r+1})$ of $[d]$, and define the sum of each part $X^i(\boldsymbol{x}) = \sum_{j \in P_i} \boldsymbol{x}_j$ for query $\boldsymbol{x}$ (where we use $\boldsymbol{a}_i$ to denote the ith entry of $\boldsymbol{a}$). With such random partitions, we construct a similar chaining-like function $f_{\mathcal{P}}$ with the properties: (a) $\|\nabla f_{\mathcal{P}}(\boldsymbol{x})\|$ will not vanish unless $|X^i(\boldsymbol{x}) - X^{i+1}(\boldsymbol{x})| > 1$ for all $i \in [r]$ (Lemma 9), and (b) the gradient will be independent of $P_{i+1}, \dots, P_r$ if $|X^j(\boldsymbol{x}) - X^{j-1}(\boldsymbol{x})| < 1/2$, for $j \geq i$ (Lemma 8). Finally, the key property (c) is that without identifying the elements in P_i and P_{i+1}, the value $|X^i(\boldsymbol{x}) - X^{i+1}(\boldsymbol{x})| < 1/2$ with a probability $1 - d^{-\omega(1)}$. With such a high probability, even with polynomial times of "shots in the dark", the algorithm cannot process information of the future part of the partition. Thus, any parallel algorithm can only learn one part of the random partition with high probability. To prove property (c), we utilize the concentration bound of conditional Bernoulli random variables (Theorem 12).

Comparison with construction for stochastic settings. The only existing lower bound for nonconvex optimization with a large batch of queries pertains to the stochastic case. (Arjevani et al., 2023) utilized the same chaining-like function and perturbed the noiseless gradient oracle to a stochastic "zero-chaining" gradient oracle such that any algorithm can recover information about the future orthogonal vector $\boldsymbol{u}^j$ from the noisy oracle at iteration $t < j$ with an exponentially small probability. Furthermore, they effectively controlled the variance associated with the stochastic gradient oracle. However, the noise in the stochastic oracle is crucial for obscuring information, making it inapplicable to our noiseless case.

1.2.2. TRAP THE FLOW WITHIN A CONSTANT NUMBER OF ITERATIONS IN CONSTANT-DIMENSIONAL SPACE

Previous algorithms for finding stationary points in constant-dimensional spaces (Bubeck et al., 2019; Hollender and Zampetakis, 2023) employed a similar flow-trapping framework. At a high level, in each iteration, these algorithms identify a search domain containing an ε-stationary point and reduce either the volume or the diameter of the domain by a constant factor. By the Lipschitz property of the gradient, if the diameter of the domain is $O(\varepsilon)$, an ε-stationary point can be located

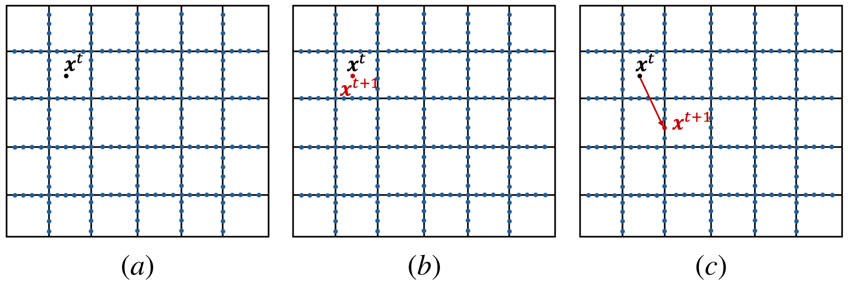

Figure 1: Illustration of **iterate update** on 2-dimensional space. We plot trap barriers as black grids and queried points on the trap barrier as blue dots. (a) Current iterate location. (b) If all the queried points are ε_t-unreachable from $\boldsymbol{x}_t$, then $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. ($c$) If some queried points are ε_t-reachable, we select $\boldsymbol{x}^{t+1}$ as the point with smallest function value.

within it. Moreover, the constant reduction in diameter ensures that $\Theta(\log(1/\varepsilon))$ iterations are sufficient to achieve the desired accuracy. Our algorithm (Algorithm 1) also proceeds with a similar identification-compression framework. However, to reduce the volume to $O(\varepsilon)$ in $\Theta(1)$ iterations, we design a new algorithm with more “traps” in each iteration to identify much smaller domains, which is one of our main technical contributions.

Boundary unreachability implies stationary point inside. We say a point $\boldsymbol{y}$ is ε-unreachable from a point $\boldsymbol{x}$ if $f(\boldsymbol{y}) > f(\boldsymbol{x}) - \varepsilon \|\boldsymbol{x} - \boldsymbol{y}\|$. By the mean value theorem, if $\boldsymbol{y}$ is ε-unreachable from $\boldsymbol{x}$, then the gradient flow starting from $\boldsymbol{x}$ and following the steepest descent direction cannot reach $\boldsymbol{y}$, unless it encounters an ε-stationary point along the path. Conversely, if all boundary points of a domain is ε-unreachable, this domain must contain an ε-stationary point (Lemma 26). With such a property, if we can compress the region such that the diameter is $O(\varepsilon)$, we can find an ε-stationary point.

Our method: Gradient Flow Grid Trapping (GFGT). The key technical challenge is to ensure the boundary remain ε-unreachable while compressing the search domain within a constant number of iterations. Inspired by the gradient flow trapping algorithm (Bubeck and Mikulincer, 2020; Hollender and Zampetakis, 2023), we try to trap the flow but with more barriers in each iteration, as shown in Algorithm 1. At a high-level, at iteration t, the algorithm first sets ℓ_t equally-spaced barriers on every d dimensions and queries a δ_t-net on every barrier. If no queried point is ε_t-reachable from current iterate $\boldsymbol{x}^t$, we do not update $\boldsymbol{x}^t$ and set $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. Otherwise, we set $\boldsymbol{x}^{t+1}$ to the reachable point with smallest objective value among queried points. We illustrate such an identification process in Figure 1. Subsequently, we compress the search domain centered around $\boldsymbol{x}^{t+1}$, while ensuring the boundary remains unreachable from the $\boldsymbol{x}^{t+1}$. This compression process is visualized in Figure 2. Such a construction allows the diameter of the hyperrectangle is reduced by at least $3/\ell_t$. Thus, it is sufficient to set $\ell_0 \times \cdots \times \ell_k = O(1/\varepsilon)$ to ensure that the diameter is bounded by $O(\varepsilon)$ with $k = \Theta(1)$ number of iterations.

Boundary unreachability during our algorithm. Let R_{t+1} denote the new hyperrectangle at the start of $(t+1)$-th iteration. We make the following two key observations regarding the iterate $\boldsymbol{x}^{t+1}$ and the hyperrectangle R_{t+1}. (a) any point on ∂R_{t+1} either belongs to a barrier controlled by a δ_t-net where every point has been queried or lies on the previous boundary ∂R_t and (b) the solution

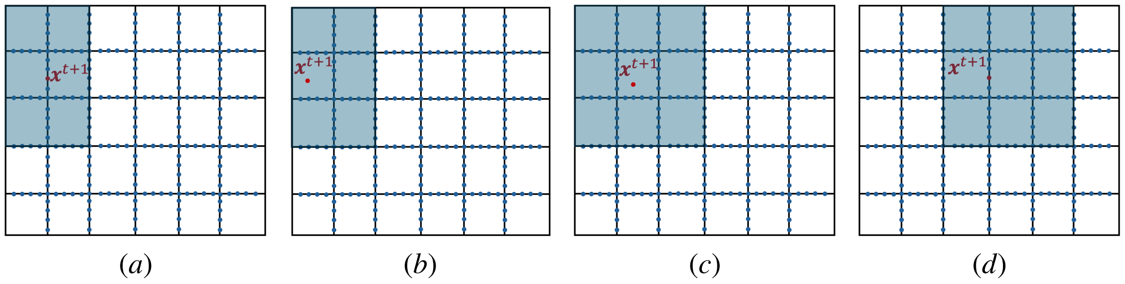

Figure 2: Illustration of **domain compression** on 2-dimensional space depending on the location of $\boldsymbol{x}^{t+1}$. The shaded areas represent the compressed domain for the next iteration. *(a,b)* If $\boldsymbol{x}^{t+1}$ is close to a boundary (in this case, the left boundary), the domain is extended only to the other directions (top, bottom, right). *(c,d)* If $\boldsymbol{x}^{t+1}$ is not close to any boundary, the domain is extended for to directions, depending on whether $\boldsymbol{x}^{t+1}$ is on the trap barrier.

$\boldsymbol{x}^{t+1}$ is always sufficiently far from the boundary ∂R_{t+1}. To leverage the finite δ_t-net in controlling an infinite number of points on ∂R_{t+1}, we slightly increase the accuracy $\varepsilon_{t+1} = \varepsilon_t + O(\varepsilon)$. We analyze the unreachability of boundary points under the following scenarios.

- If the point lies in the intersection $\partial R_t \cap \partial R_{t+1}$, we establish ε_{t+1}-unreachability by the triangle inequality, since any point on ∂R_t is already ε_t-unreachable.
- If the point lies on the new boundary, it can be controlled by a δ_t-net where every point has been queried.
  1. If the iterate is not updated, we show ε_{t+1}-unreachability by Lemma 30 and Observation (b).
  2. If the iterate is updated to the minimal-value one among reachable all queried points, then the other queried point must be ε_t-unreachable from the minimal-value solution. Then we apply the same argument to show the ε_{t+1}-unreachability.

Comparison with constant-iteration algorithms for local minimum search on grid graphs. Since the problem of local minimum search on grid graphs can be reduced to finding a stationary point (Bubeck and Mikulincer, 2020; Vavasis, 1993), constant-iteration algorithms for local minimum search on grid graphs share similar structural features. In particular, these algorithms query *all* points on the boundary of a subregion to progressively trap the local minimum within a significantly shrinking region. They then proceed to query all points within the final subregion. However, to trap a stationary point, one must ensure that all boundary points are unreachable. Since directly querying all points on the boundary of a sub-region would require infinitely many queries, we instead adopt the approach of (Hollender and Zampetakis, 2023), which ensures boundary unreachable by placing the next center point sufficiently far from the boundary. To implement this, we query a δ-net with spacing smaller than ε, thereby ensuring the stationary point remains unreachable via any finite number of queries.

Constant-dimensional lower bound. To analyze the constant-dimensional case, we reduce the problem to local search on grid graphs and leverage the hardness of random staircase construction. Specifically, we show that the problem of finding stationary points can be reduced to locating the local minimum of a monotone path function on a grid graph, following (Bubeck and Mikulincer, 2020; Vavasis, 1993). A smooth function is constructed to preserve the path's monotonicity, ensuring the stationary point lies near the local minima. Notably, such a reduction is round-preserving. The adaptive complexity is analyzed using random staircase construction (Brânzei and Li, 2022), where the staircase grows randomly at each step, restricting deterministic algorithms from accessing points beyond the current round with constant probability. We demonstrate that these hard instances hold for any algorithm running within $\Theta(\log(1/\varepsilon))$ rounds.

2. Preliminaries

Given a smooth objective function $f : \mathbb{R}^d \to \mathbb{R}$, the goal of non-convex optimization is to find a (possibly random) stationary point $\boldsymbol{x} \in \mathbb{R}^d$ such that $\mathbb{E}\left[\|\nabla f(\boldsymbol{x})\|\right] \le \varepsilon$. We further assume the initial point $\boldsymbol{x}^0$ satisfies $f(\boldsymbol{x}^0) - \inf_{\boldsymbol{x}} f(\boldsymbol{x}) \le \Delta$ for some $\Delta \ge 0$. In the constant dimensional setting, we also consider the cube-constrained minimization where f is restricted to $[0,1]^d$. In this case, we focus on finding an ε-KKT point, *i.e.*, the norm of the projected gradient instead of the gradient being bounded by ε (Hollender and Zampetakis, 2023).

Function class. A function $f : \mathbb{R}^d \to \mathbb{R}$ has L_p-Lipschitz p-th order derivatives if it is p times continuously differentiable, and for every $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{v} \in \mathbb{R}^d$, $\|\boldsymbol{v}\| = 1$, the directional projection $t \mapsto f_{\boldsymbol{x},\boldsymbol{v}}(t) := f(\boldsymbol{x} + t \cdot \boldsymbol{v})$ of f satisfies $\left| f_{\boldsymbol{x},\boldsymbol{v}}^{(p)}(t) - f_{\boldsymbol{x},\boldsymbol{v}}^{(p)}(t') \right| \le L_p \left| t - t' \right|$ for $t, t' \in \mathbb{R}$, where $f_{\boldsymbol{x},\boldsymbol{v}}^{(p)}(\cdot)$ is the pth derivative of $t \mapsto f_{\boldsymbol{x},\boldsymbol{v}}(t)$. We occasionally refer to a function with Lipschitz continuous p-th order derivatives as p-th order smooth and denote the class of such functions by $\mathcal{F}_p(L_p)$.

Oracle. In this work, we investigate the model where the algorithm queries points to the oracle $\mathcal{O}$. Given the potential function f, and a query $\boldsymbol{x} \in \mathbb{R}^d$, the 0-th order oracle answers the function value $f(\boldsymbol{x})$ and the p-th order oracle answers both $f(\boldsymbol{x})$ and its i-th order derivative $\nabla^i f(\boldsymbol{x})$, for any $i \in [p]$. In the following, we focus solely on the zeroth-order oracle, as higher-order oracles can be constructed using $\mathsf{poly}(d)$ queries to the zeroth-order oracle.

Adaptive algorithm class. The class of *adaptive* algorithms is formally defined as follows (Diakonikolas and Guzmán, 2019; Zhou et al., 2024). For any dimension d, an adaptive algorithm A takes $f : \mathbb{R}^d \to \mathbb{R}$ and a (possibly random) initial point $\boldsymbol{x}^0$ and iteration number r as input and returns an output $\boldsymbol{x}^{r+1}$, which is denoted as $\mathsf{A}[f, \boldsymbol{x}^0, r] = \boldsymbol{x}^{r+1}$. At iteration $i \in [r] := \{1, \dots, r\}$, A performs a batch of queries

$$Q^i = \{\boldsymbol{x}^{i,1}, \dots, \boldsymbol{x}^{i,k_i}\}, \quad \text{with } \boldsymbol{x}^{i,j} \in \mathbb{R}^d, \;\; j \in [k_i], \;\; k_i = \mathsf{poly}\big(d, 1/\varepsilon\big),$$

Given queries set Q^i, the oracle returns a batch of answers: $\mathcal{O}(Q^i) = \{\mathcal{O}(\boldsymbol{x}^{i,1}), \dots, \mathcal{O}(\boldsymbol{x}^{i,k_i})\}$.

An adaptive algorithm A is *deterministic* if in every iteration $i \in \{0, \dots, r\}$, A operates with the form $Q^{i+1} = \mathsf{A}^i(Q^0, \mathcal{O}(Q^0), \dots, Q^i, \mathcal{O}(Q^i))$, where A^i is mapping into $\mathbb{R}^{dk_{i+1}}$ with $Q^{r+1} = \boldsymbol{x}^{r+1}$ as output and $Q^0 = \boldsymbol{x}^0$ as an initial point. We denote the class of adaptive deterministic algorithms by $\mathcal{A}_{\text{det}}$. An adaptive *randomized* algorithm has the form $Q^{i+1} = \mathsf{A}^i(\xi_i, Q^0, \mathcal{O}(Q^0), \dots, Q^i, \mathcal{O}(Q^i))$, with access to a uniform random variable on $[0,1]$ (*i.e.*, infinitely many random bits), where A^i is mapping into $\mathbb{R}^{dk_{i+1}}$. We denote the class of adaptive randomized algorithms by $\mathcal{A}_{\text{rand}}$.

Notion of complexity Given $\varepsilon > 0$, $f \in \mathcal{F}$, and some algorithm A, define the running iteration $\mathsf{T}(\mathsf{A}, f, \boldsymbol{x}^0, \varepsilon)$ as the minimum number of rounds such that algorithm A outputs a solution $\boldsymbol{x}$ that satisfies $\mathbb{E}\left[\|\nabla f(\boldsymbol{x})\|\right] \leq \varepsilon$, *i.e.*, $\mathsf{T}(\mathsf{A}, f, \boldsymbol{x}^0, \varepsilon) = \inf\left\{t : \mathbb{E}\left[\left\|\nabla f\left(\mathsf{A}[f, \boldsymbol{x}^0, t]\right)\right\|\right] \leq \varepsilon\right\}$. We define the *worst case* complexity as

$$\mathsf{Comp}_{\mathsf{WC}}(\mathcal{F}, \varepsilon, \boldsymbol{x}^0) := \inf_{\mathsf{A}\in\mathcal{A}_{\text{det}}} \sup_{f\in\mathcal{F}} \mathsf{T}(\mathsf{A}, f, \boldsymbol{x}^0, \varepsilon).$$

For some randomized algorithm $\mathsf{A} \in \mathcal{A}_{\text{rand}}$, we define the *randomized* complexity as

$$\mathsf{Comp}_{\mathsf{R}}(\mathcal{F}, \varepsilon, \boldsymbol{x}^0) := \inf_{\mathsf{A}\in\mathcal{A}_{\text{rand}}} \sup_{f\in\mathcal{F}} \mathsf{T}(\mathsf{A}, f, \boldsymbol{x}^0, \varepsilon).$$

By definition, we have $\mathsf{Comp}_{\mathsf{WC}}(\mathcal{F}, \varepsilon, \boldsymbol{x}^0) \geq \mathsf{Comp}_{\mathsf{R}}(\mathcal{F}, \varepsilon, \boldsymbol{x}^0)$. In the rest of this paper, we only consider the randomized complexity and we lower-bound it by considering the *distributional* complexity:

$$\mathsf{Comp}_{\mathsf{D}}(\mathcal{F}, \varepsilon, \boldsymbol{x}^0) := \sup_{F\in\Delta(\mathcal{F})} \inf_{\mathsf{A}\in\mathcal{A}_{\text{rand}}} \mathbb{E}_{f\sim F}\, \mathsf{T}(\mathsf{A}, f, \boldsymbol{x}^0, \varepsilon),$$

where $\Delta(\mathcal{F})$ is the set of probability distributions over the class of functions $\mathcal{F}$.

3. High dimensional lower bounds for p-th order methods

In this section, we prove the adaptive lower bound for $d = \widetilde{\Omega}\big(\varepsilon^{-\frac{2+2p}{p}}\big)$ (Theorem 4), where the number of queries per iteration is dominated by $\mathsf{poly}(d)$. We first present our hard functions family and analyze their properties in Section 3.1. We characterize the output for any algorithm and give proof of Theorem 4 in Section 3.2 .

Theorem 4 *There exist numerical constants $0 < c_0, c_1, c_2 < \infty$ such that for all $p \geq 1, p \in \mathbb{N}$, and let Δ, L_p, $\varepsilon > 0$ and d with $\frac{d}{\log^2 d} \geq c_2\big(\frac{L_p}{l_p}\big)^{2/p}\varepsilon^{-\frac{2+2p}{p}}$, we have*

$$\mathsf{Comp}_{\mathsf{R}}(\mathcal{F}_p(L_p), \varepsilon, \Delta) \geq c_0 \cdot \Delta\left(\frac{L_p}{l_p}\right)^{1/p} \varepsilon^{-\frac{1+p}{p}},$$

where $l_p \leq e^{c_1 p \log p + c_1}$.

Compared to the query complexity presented in Theorem 2 of (Carmon et al., 2020), our adaptive complexity achieves the same query complexity but with only a logarithmic dependence on the dimension. This logarithmic scaling arises from our use of random partitions to hide information, where the logarithmic size of each part of partition is essential for ensuring concentration in high-dimensional space.

3.1. Hard functions and its properties

Let $d_0 \in \mathbb{N}_+$ with $d_0 \geq \log^2 d$, and $r = d/d_0 - 2$. Consider partition with fixed-size components as $P_1 \cup P_2 \cup \cdots \cup P_{r+2} = [d]$, where $|P_1| = d_0$, $|P_1| = |P_i|$ for all $i \in [r]$. The partition is uniformly random among all partitions with such fixed-size parts, and we denote it as $\mathcal{P}$. For any $\boldsymbol{x} \in \mathbb{R}^d$ let $X^i(\boldsymbol{x}) = (\sum_{s\in P_i} \boldsymbol{x}_s)/\sqrt{d_0}$, for all $i \in [r+1]$, and we denote $X(\boldsymbol{x}) = (X^1(\boldsymbol{x}), \ldots, X^{r+1}(\boldsymbol{x}))$. We define the (unscaled) hard function $f_{\mathcal{P}} : \mathbb{R}^d \to \mathbb{R}$ as,

$$f_{\mathcal{P}}(\boldsymbol{x}) = g_{\mathcal{P}}(\rho(\boldsymbol{x})) + \frac{1}{5}\|\boldsymbol{x}\|^2 ,$$

where $\rho(\boldsymbol{x}) = \frac{\boldsymbol{x}}{\sqrt{1+\|\boldsymbol{x}\|^2/R^2}}$ and $R = 230\sqrt{r+1}$ and

$$g_{\mathcal{P}}(\boldsymbol{x}) = -\Psi(1)\Phi(X^1) - \sum_{i=1}^{r}(-1)^i \left[\Psi(X^{i-1} - X^i)\Phi(X^{i+1} - X^i) - \Psi(X^i - X^{i-1})\Phi(X^i - X^{i+1})\right] ,$$

where $X^0 \equiv 0$ and the component functions are defined as (Carmon et al., 2020),

$$\Psi(x) := \begin{cases} 0 & x \le 1/2 \\ \exp\left(1 - \frac{1}{(2x-1)^2}\right) & x > 1/2 \end{cases} \quad \text{and} \quad \Phi(x) = \sqrt{e}\int_{-\infty}^{x} e^{-\frac{1}{2}t^2}\mathrm{d}t\,.$$

We enumerate all the relevant properties of the component functions Ψ and Φ in the following.

Lemma 5 ((Carmon et al., 2020, Lemma 1)) *The functions Ψ and Φ satisfy the following.*

1. *For all $x \le 1/2$ and all $k \in [N]$, $\Psi^{(k)}(x) = 0$.*
2. *For all $x \ge 1$ and $|y| < 1$, $\Psi(x)\Phi'(y) > 1$.*
3. *Both Ψ and Φ are infinitely differentiable, and for all $k \in [N]$ we have*

$$\sup_x |\Psi^{(k)}(x)| \le \exp\left(\frac{5k}{2}\log(4k)\right) \text{ and } \sup_x |\Phi^{(k)}(x)| \le \exp\left(\frac{3k}{2}\log\frac{3k}{2}\right).$$

4. *The functions and derivatives Ψ, Ψ', Φ and Φ' are non-negative and bounded, with*

$$0 \le \Psi < e, \ \ 0 \le \Psi' \le \sqrt{54/e}, \ \ 0 < \Phi < \sqrt{2\pi e}, \ \text{ and } \ 0 < \Phi' \le \sqrt{e}.$$

By careful calculation and Lemma 5, we can obtain the smoothness and boundedness of $f_{\mathcal{P}}$ and $g_{\mathcal{P}}$. We defer the proof in Appendix B.1 and Appendix B.2.

Lemma 6 (Smoothness and boundness of $g_{\mathcal{P}}$) *The function $g_{\mathcal{P}}$ satisfies the following.*

1. *We have $g_{\mathcal{P}}(\mathbf{0}) - \inf_{\boldsymbol{x}} g_{\mathcal{P}}(\boldsymbol{x}) < 12r$;*
2. *For all $\boldsymbol{x} \in \mathbb{R}^d$, $\|\nabla g_{\mathcal{P}}(\boldsymbol{x})\| \le 46\sqrt{r+1}$.*

Lemma 7 (Smoothness and boundness of $f_{\mathcal{P}}$) *The function $f_{\mathcal{P}}$ satisfies the following,*

1. *We have $f_{\mathcal{P}}(\mathbf{0}) - \inf_{\boldsymbol{x}} f_{\mathcal{P}}(\boldsymbol{x}) < 12r$.*
2. *For every $p \ge 1$, the p-th order derivatives of $f_{\mathcal{P}}$ are l_p-Lipschitz continuous, where $l_p \le \exp(cp\log p + c)$ for a numerical constant $c < \infty$.*

3.2. Characterization of output

We now turn to analyze the output when interacting with $f_{\mathcal{P}}$ in the following lemma. The proof can be found at Appendix B.3.

Lemma 8 (Characterization of output) *If there exists a constant $\alpha = \omega(1)$ such that $8R\sqrt{\frac{\alpha \log d}{d_0}} \leq \frac{1}{2}$, then for any randomized algorithm A, any $\tau \leq r$, and any initial point $\boldsymbol{x}^0$, $X(\mathsf{A}[f_{\mathcal{P}}, \boldsymbol{x}^0, \tau]) = (X^1(\rho(\boldsymbol{x})), \ldots, X^{r+1}(\rho(\boldsymbol{x})))$ takes a form as*

$$(x_1, \ldots, x_{2\tau}, x_{2\tau+1}, x_{2\tau+1}, \ldots, x_{2\tau+1}),$$

up to addictive error $1/4$ for every coordinate with probability $1 - d^{-\omega(1)}$ over $\mathcal{P}$.

Proof (Proof sketch of Lemma 8). We fix τ and prove the following by induction for $l \in [\tau]$: With high probability, the computation path of the (deterministic) algorithm A and the queries it issues in the l-th round are determined by $P_1, \ldots, P_{2l-2}$. We first consider deterministic algorithm and uniformly random partition.

To prove the inductive claim, we let $\mathcal{E}_l$ as event that any answer of query issued in iteration l, the answer only depends on $P_1, \ldots, P_{2l}$, *i.e.*, $\forall \boldsymbol{x} \in Q^l$, $f_{\mathcal{P}}(\boldsymbol{x}) = g^l_{\mathcal{P}}(\rho(\boldsymbol{x})) + \frac{1}{5}\|\boldsymbol{x}\|^2$. Combining the assumption that the queries in round l depend only on $P_1, \ldots, P_{2l-2}$, if $\mathcal{E}_l$ occurs, the computation path is determined by $P_1, \ldots, P_{2l}$. Thus if all of $\mathcal{E}_1, \ldots, \mathcal{E}_l$ occur, the computation path in round l is determined by $P_1, \ldots, P_{2l}$.

It is sufficient to show the conditional probability $\mathbb{P}\left[\mathcal{E}_l \mid \mathcal{E}_1, \ldots, \mathcal{E}_{l-1}\right]$ is at least $1 - d^{\omega(1)}$ to hide information from polynomial queries. Further, by the chaining structure of hardness function and $\Psi(x) = 0$ whenever $x < 1/2$, it is sufficient to show for any fixed $i \geq 2l$, with probability at least $1 - d^{\omega(1)}$,

$$\left|X^i(\rho(\boldsymbol{x})) - X^{i+1}(\rho(\boldsymbol{x}))\right| \leq \frac{1}{2}.$$

This can be guaranteed by uniformly random partition and the concentration of linear functions over the Boolean slice (Theorem 12). Finally, we note that by allowing the algorithm to use random bits, the results are a convex combination of the bounds above, so the same high-probability bounds are satisfied. ■

Given such characterization, the remaining goal is to show that without final part being discovered, the gradient norm of output will always be large. We summarize this in the following lemma and the proof can be found in Appendix B.4.

Lemma 9 (Small weighted partition implies large gradient norm) *For any $\boldsymbol{x} \in \mathbb{R}^d$, if $|X^{r+1}(\rho(\boldsymbol{x})) - X^r(\rho(\boldsymbol{x}))| < 1$, then $\|\nabla f_{\mathcal{P}}(\boldsymbol{x})\| \geq 0.08$.*

Now we are ready to prove Theorem 4 by leveraging Lemma 8 and Lemma 9.

Proof (Proof of Theorem 4). We consider the scaled hard function $f^0_{\mathcal{P}} : \mathbb{R}^d \to \mathbb{R}$ as

$$f^0_{\mathcal{P}}(\boldsymbol{x}) = \frac{L_p \sigma^{p+1}}{l_p} f_{\mathcal{P}}(\boldsymbol{x}/\sigma),$$

where scale parameter $\sigma > 0$ are to be determined, and the quantity $l_p \leq \exp(c_1 p \log p + c_1)$ for a numerical constant c_1 is defined in Lemma 7.

Combining Lemma 8 and Lemma 9, we can claim if there exists a constant $\alpha = \omega(1)$ such that $8R\sqrt{\frac{\alpha \log d}{d_0}} \leq \frac{1}{2}$, then

$$\|\nabla f_{\mathcal{P}}(\boldsymbol{x}/\sigma)\| \geq 0.08,$$

with probability $1 - d^{-\omega(1)}$ over $\mathcal{P}$. Taking $\sigma = \left(\frac{l_p \varepsilon}{0.08 L_p}\right)^{1/p}$, we have

$$\mathsf{Comp}_{\mathsf{R}}(\mathcal{F}, \varepsilon, \Delta) \geq (1 - d^{-\omega(1)}) \cdot r/2.$$

Then, we have

$$f^0_{\mathcal{P}}(\mathbf{0}) - \inf_{\boldsymbol{x}} f^0_{\mathcal{P}}(\boldsymbol{x}) \leq \frac{L_p \sigma^{p+1}}{l_p} 12r \leq 1857 \frac{l_p^{1/p}}{L_p^{1/p}} \varepsilon^{(p+1)/p} r.$$

Then we take $r = \lfloor \frac{\Delta}{1857} (L_p/l_p)^{1/p} \varepsilon^{-\frac{1+p}{p}} \rfloor$.

Finally, we check the requirement of dimension d to satisfy that there exists a constant $\alpha = \omega(1)$ such that $8R\sqrt{\frac{\alpha \log d}{d_0}} \leq \frac{1}{2}$. It is sufficient to let $16^2 \cdot 230^2 (r+1)^2 \log^2 d \leq d$. ■

4. Constant dimensional cases

In this section, we first establish the lower bound on the number of queries required per iteration for algorithms operating within $k < O(\log(1/\varepsilon))$ iterations, as detailed in Section 4.2. Subsequently, we derive the upper bound on the number of queries required per iteration for algorithms running within a constant number of iterations, as outlined in Section 4.1.

4.1. Upper bound for constant iteration case

Theorem 10 *For $\varepsilon > 0$, $d, k = \Theta(1)$ with $d \geq 2$, there is a deterministic algorithm finding ε-stationary points, running within k-round, with*

- $C_1(d, k, L, \Delta) \cdot \varepsilon^{-\frac{d-1}{\left(\frac{2d}{d+1}\right)^k - 1} - d - 1}$ *queries per iteration when f is unconstrained,*
- $C_2(d, k, L) \cdot \varepsilon^{-\frac{d-1}{2\left(\left(\frac{2d}{d+1}\right)^k - 1\right)} - \frac{d-1}{2}}$ *queries per iteration when f is constrained on $[0,1]^d$,*

where $C_1(d, k, L, \Delta)$ and $C_2(d, k, L)$ are uniformly constants depending on d, k, L, Δ and d, k, L respectively.

The unconstrained case has square number of queries due to the initialization region scale with $\frac{\Delta}{\varepsilon}$ instead of $[0,1]^d$. For constrained case, our upper bound is asymptotically tight compared to the lower bound $\Omega(\varepsilon^{-\frac{d-1}{2(d^k - 1)} - \frac{d-1}{2}})$ in Theorem 11. We note that numerically, $(d-1)/\left(\left(\frac{2d}{d+1}\right)^k - 1\right) < 0.05$ when $d = 2, k = 9$. Furthermore, when $d = 2$ and $k = 1$, the complexity is ε^{-2} which matches the complexity of a naïve grid search. These upper bounds are first non-trivial results for $d, k = \Theta(1)$. The algorithms and their proofs are deferred to Appendix D.

4.2. Lower bounds for $O(\log(1/\varepsilon))$ iteration case

Theorem 11 *For $\varepsilon > 0$, $d = \Theta(1)$ with $d \geq 2$ and $k \in \mathbb{N}$ with $k = O(\log(1/\varepsilon))$, any (possible randomized) algorithm running within k-round, and issuing $C(d) \cdot \varepsilon^{-\frac{d^k}{d^k-1} \cdot \frac{d-1}{2}} \cdot \frac{1}{k}$ queries per round fails to find ε-stationary points of a smooth function over $[0,1]^d$ with probability at least $7/40$.*

When $k = \Theta(\log(1/\varepsilon))$, the lower bound is

$$\Omega\left(\varepsilon^{-\left(1+\frac{1}{d^k-1}\right) \cdot \frac{d-1}{2}} \cdot \frac{1}{k}\right) = \Omega\left(\varepsilon^{-\frac{d-1}{2}} \cdot \varepsilon^{-\frac{d-1}{2(d^k-1)}} \cdot \frac{1}{k}\right) = \widetilde{\Omega}\left(\varepsilon^{-\frac{d-1}{2}}\right),$$

the last equality is implied from $\varepsilon^{-1} \leq (\log(1/\varepsilon))^{\frac{2(d^k-1)}{d-1}}$ and $\varepsilon^{-\frac{d-1}{2(d^k-1)}} \cdot \frac{1}{k} \geq \frac{1}{k}$. Thus our lower bound matches the upper bound established in (Hollender and Zampetakis, 2023), up to a logarithmic factor. The proof is based on the reduction to the problem of finding local minima in grid graph (Vavasis, 1993; Bubeck and Mikulincer, 2020) and the adaptive complexity of finding local minima in grid graph (Brânzei and Li, 2022). The details can be found in Appendix C.

5. Conclusions and future works

In this paper, we make significant progress towards understanding parallelization for non-convex optimization, in terms of both upper and lower bounds. In particular, when $d = \widetilde{\Omega}\left(\varepsilon^{-(2+2p)/p}\right)$, we discover that parallelization cannot accelerate non-convex optimization, leading to a complexity of $\Omega(\varepsilon^{-(1+p)/p})$ for finding a stationary point, which matches the lower bound for one-query-per-round algorithms. When $d = \Theta(1)$, our algorithm finds approximate stationary point within constant iteration and the asymptotically optimal query requirement which matches our lower bound. Furthermore we answer the open problem in (Bubeck and Mikulincer, 2020) by showing $\widetilde{\Omega}(\varepsilon^{-(d-1)/2})$ is necessary for $\log(1/\varepsilon)$ iteration algorithms.

There are several interesting directions for future exploration. Exploring the role of parallelization for intermediate dimension regime ($d = \omega(1) \leq d \leq O(\varepsilon^{-4})$) is an intriguing challenge that continues to be of significant academic interest. Naïve adaptation of lower bounds from high-dimensional cases is non-trivial, as the construction used to hide information from polynomial queries relies on high dimensionality to ensure concentration inequalities. Further, the reduction to finding local minima on a grid graph depends on constant dimensions, which breaks down when dimension increses. In addition, the query complexity of high-order methods in the constant-dimensional case remains unknown.

Acknowledgments

HZ was supported by International Graduate Program of Innovation for Intelligent World and Next Generation Artificial Intelligence Research Center. MS was supported by JST ASPIRE Grant Number JPMJAP2405. AT was supported by JSPS KAKENHI Grant Number 23H03351.

References

Arpit Agarwal, Shivani Agarwal, Sepehr Assadi, and Sanjeev Khanna. Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons. In *Conference on Learning Theory*, pages 39–75. PMLR, 2017.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019.

Nima Anari, Sinho Chewi, and Thuy-Duong Vuong. Fast parallel sampling under isoperimetry. In *Proceedings of Thirty Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pages 161–185. PMLR, 30 Jun–03 Jul 2024.

Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1):165–214, 2023.

Eric Balkanski and Yaron Singer. The adaptive complexity of maximizing a submodular function. In *Proceedings of the 50th annual ACM SIGACT symposium on theory of computing*, pages 1138–1151, 2018a.

Eric Balkanski and Yaron Singer. Parallelization does not accelerate convex optimization: Adaptivity lower bounds for non-smooth convex minimization. *arXiv preprint arXiv:1808.03880*, 2018b.

Keith Ball et al. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31 (1-58):26, 1997.

Ernesto G Birgin, JL Gardenghi, José Mario Martínez, Sandra Augusta Santos, and Ph L Toint. Worst-case evaluation complexity for unconstrained nonlinear optimization using high-order regularized models. *Mathematical Programming*, 163:359–368, 2017.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

Simina Brânzei and Jiawei Li. The query complexity of local search and brouwer in rounds. In *Conference on Learning Theory*, pages 5128–5145. PMLR, 2022.

Mark Braverman, Jieming Mao, and S Matthew Weinberg. Parallel algorithms for select and partition with noisy comparisons. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 851–862, 2016.

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

Sébastien Bubeck, Qijia Jiang, Yin-Tat Lee, Yuanzhi Li, and Aaron Sidford. Complexity of highly parallel non-smooth convex optimization. *Advances in neural information processing systems*, 32, 2019.

Clément L Canonne and Tom Gur. An adaptivity hierarchy theorem for property testing. *computational complexity*, 27:671–716, 2018.

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

Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Lower bounds for finding stationary points ii: first-order methods. *Mathematical Programming*, 185(1):315–355, 2021.

Yair Carmon, Arun Jambulapati, Yujia Jin, Yin Tat Lee, Daogao Liu, Aaron Sidford, and Kevin Tian. Resqueing parallel and private stochastic convex optimization. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2031–2058. IEEE, 2023.

Coralia Cartis and Lindon Roberts. Scalable subspace methods for derivative-free nonlinear least-squares optimization. *Mathematical Programming*, 199(1):461–524, 2023.

Coralia Cartis, Nicholas IM Gould, and Ph 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.

Coralia Cartis, Nicholas IM Gould, and Philippe L Toint. Sharp worst-case evaluation complexity bounds for arbitrary-order nonconvex optimization with inexpensive constraints. *SIAM Journal on Optimization*, 30(1):513–541, 2020a.

Coralia Cartis, Nick IM Gould, and Ph L Toint. A concise second-order complexity analysis for unconstrained optimization using high-order regularized models. *Optimization Methods and Software*, 35(2):243–256, 2020b.

Deeparnab Chakrabarty, Andrei Graur, Haotian Jiang, and Aaron Sidford. Improved lower bounds for submodular function minimization. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 245–254. IEEE, 2022.

Deeparnab Chakrabarty, Andrei Graur, Haotian Jiang, and Aaron Sidford. Parallel submodular function minimization. *Advances in Neural Information Processing Systems*, 36, 2024.

Xi Chen, Rocco A Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. *Journal of the ACM (JACM)*, 65(6):1–18, 2018.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204. PMLR, 2015.

Richard Cole. Parallel merge sort. *SIAM Journal on Computing*, 17(4):770–785, 1988.

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. Large scale distributed deep networks. *Advances in Neural Information Processing Systems*, 25, 2012.

Jelena Diakonikolas and Cristóbal Guzmán. Lower bounds for parallel and randomized convex optimization. In *Conference on Learning Theory*, pages 1132–1157. PMLR, 2019.

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

Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pages 797–842. PMLR, 2015.

Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. *Advances in neural information processing systems*, 29, 2016.

Alexandros Hollender and Emmanouil Zampetakis. The computational complexity of finding stationary points in non-convex optimization. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 5571–5572. PMLR, 2023.

Prateek Jain, Purushottam Kar, et al. Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, 10(3-4):142–363, 2017.

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

Kenji Kawaguchi. Deep learning without poor local minima. *Advances in Neural Information Processing Systems*, 29, 2016.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jeongyeol Kwon, Dohyun Kwon, and Hanbaek Lyu. On the complexity of first-order methods in stochastic bilevel optimization. *arXiv preprint arXiv:2402.07101*, 2024.

Wenzheng Li, Paul Liu, and Jan Vondrák. A polynomial lower bound on adaptive complexity of submodular maximization. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 140–152, 2020.

Cong Ma, Kaizheng Wang, Yuejie Chi, and Yuxin Chen. Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval and matrix completion. In *International Conference on Machine Learning*, pages 3345–3354. PMLR, 2018.

Katta G Murty and Santosh N Kabadi. Some np-complete problems in quadratic and nonlinear programming. Technical report, 1985.

Arkadij Semenovič Nemirovskij and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

Yurii Nesterov. How to make the gradients small. *Optima. Mathematical Optimization Society Newsletter*, (88):10–11, 2012.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical programming*, 108(1):177–205, 2006.

Bartłomiej Polaczyk. *Concentration of Measure and Functional Inequalities*. PhD thesis, University of Warsaw, 2023.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *Advances in Neural Information Processing Systems*, 24, 2011.

Ju Sun, Qing Qu, and John Wright. A geometric analysis of phase retrieval. *Foundations of Computational Mathematics*, 18:1131–1198, 2018.

Leslie G Valiant. Parallelism in comparison problems. *SIAM Journal on Computing*, 4(3):348–355, 1975.

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

Blake Woodworth and Nathan Srebro. Lower bound for randomized first order convex optimization. *arXiv preprint arXiv:1709.03594*, 2017.

Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Syx4wnEtvH.

Pengyun Yue, Cong Fang, and Zhouchen Lin. On the lower bound of minimizing polyak-łojasiewicz functions. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 2948–2968. PMLR, 2023.

Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. *Advances in Neural Information Processing systems*, 31, 2018.

Junyu Zhang, Mingyi Hong, and Shuzhong Zhang. On lower iteration complexity bounds for the convex concave saddle point problems. *Mathematical Programming*, 194(1):901–935, 2022.

Huanjian Zhou and Masashi Sugiyama. Parallel simulation for sampling under isoperimetry and score-based diffusion models. *arXiv preprint arXiv:2412.07435*, 2024.

Huanjian Zhou, Baoxiang Wang, and Masashi Sugiyama. Adaptive complexity of log-concave sampling. *arXiv preprint arXiv:2408.13045*, 2024.

Contents

1 Introduction **1**
- 1.1 Our results . . . 2
- 1.2 Technical overview . . . 4
  - 1.2.1 High-dimensional case . . . 4
  - 1.2.2 Trap the flow within a constant number of iterations in constant-dimensional space . . . 5

2 Preliminaries **8**

3 High dimensional lower bounds for p-th order methods **9**
- 3.1 Hard functions and its properties . . . 9
- 3.2 Characterization of output . . . 11

4 Constant dimensional cases **12**
- 4.1 Upper bound for constant iteration case . . . 12
- 4.2 Lower bounds for $O(\log(1/\varepsilon))$ iteration case . . . 13

5 Conclusions and future works **13**

A Useful facts **19**

B Missing proof in Section 3 **19**
- B.1 Proof of Lemma 6 . . . 19
- B.2 Proof of Lemma 7 . . . 20
- B.3 Proof of Lemma 8 . . . 23
- B.4 Proof of Lemma 9 . . . 24

C Missing details for constant dimensional lower bound **27**

D Constant iteration gradient flow trapping **28**
- D.1 Useful tools . . . 29
- D.2 Gradient flow fully parallel trapping . . . 29
- D.3 Analysis of Algorithm 1: Proof of the first part of Theorem 10 . . . 30
- D.4 Constrained setting: proof of second part of Theorem 10 . . . 32

Appendix A. Useful facts

Theorem 12 (Concentration of linear function of conditioned Bernoullis (Theorem 4.2.5 in Polaczyk (2023))) *Let $X_1, \ldots, X_n$ be $\{0,1\}$ random variables conditioned on $\sum_{i=1}^{n} X_i = k$. Let $f : \{0,1\}^n \to \mathbb{R}$ be $f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i$ with $\alpha_i \geq 0$ for all $i \in [n]$. Then for any $t > 0$,*

$$\mathbb{P}[|f(X_1, \ldots, X_n) - \mathbb{E}[f(X_1, \ldots, X_n)]| \geq t] \leq 2 \exp\left(- \frac{t^2}{16 \sum_{i=1}^{k} (\alpha_i^{\downarrow})^2} \right),$$

where for a finite sequence x, we denote by $x^{\downarrow}$ the non-increasing rearrangement of the elements of x.

Appendix B. Missing proof in Section 3

B.1. Proof of Lemma 6

Lemma 13 (Smoothness and boundness of $g_{\mathcal{P}}$) *The function $g_{\mathcal{P}}$ satisfies the following.*

1. *We have $g_{\mathcal{P}}(\mathbf{0}) - \inf_{\boldsymbol{x}} g_{\mathcal{P}}(\boldsymbol{x}) < 12r$;*
2. *For all $\boldsymbol{x} \in \mathbb{R}^d$, $\|\nabla g_{\mathcal{P}}(\boldsymbol{x})\| \leq 46\sqrt{r+1}$.*

Proof Part 1 follows because $g_{\mathcal{P}}(\mathbf{0}) = -\Psi(1)\Phi(0) < 0$ and since $0 < \Psi < e$ and $0 \leq \Phi \leq \sqrt{2\pi e}$, we have

$$g_{\mathcal{P}}(\boldsymbol{x}) \geq -re\sqrt{2\pi e} \geq -12r.$$

For part 2, we first bound the partial derivative as follows. If $l \in P_1$, we have

$$\begin{aligned}
&\sqrt{d_0}|\partial_l g_{\mathcal{P}}(\boldsymbol{y})| \\
&= \big| -\Psi(1)\Phi'(X^1) - \Psi'(-X^1)\Phi(X^2 - X^1) - \Psi(-X^1)\Phi'(X^2 - X^1) \\
&\quad - \Psi'(X^1)\Phi(X^1 - X^2) - \Psi(X^1)\Phi'(X^1 - X^2) \\
&\quad - \Psi'(X^1 - X^2)\Phi(X^3 - X^2) - \Psi'(-X^1 + X^2)\Phi(X^2 - X^3) \big| \\
&\leq 2e\sqrt{e} + 2\sqrt{54/e}\sqrt{2\pi e} \leq 46,
\end{aligned}$$

where the last inequality holds since $\Psi(x) = 0$ and $\Psi'(x) = 0$ on any $x < 1/2$, $0 \leq \Psi'(x) \leq \sqrt{54e^{-1}}$ and $0 \leq \Phi'(x) \leq \sqrt{e}$. If $l \in P_j$ and $1 < j < r+1$, we have

$$\begin{aligned}
&\sqrt{d_0}|\partial_l g_{\mathcal{P}}(\boldsymbol{y})| \\
&= \big| -(-1)^{j+1} \left[\Psi'(X^j - X^{j+1})\Phi(X^{j+2} - X^{j+1}) + \Psi'(X^{j+1} - X^j)\Phi(X^{j+1} - X^{j+2}) \right] \\
&\quad - (-1)^j \left[-\Psi'(X^{j-1} - X^j)\Phi(X^{j+1} - X^j) - \Psi(X^{j-1} - X^j)\Phi'(X^{j+1} - X^j) \right. \\
&\quad \left. -\Psi'(X^j - X^{j-1})\Phi(X^j - X^{j+1}) - \Psi(X^j - X^{j-1})\Phi'(X^j - X^{j+1}) \right] \\
&\quad -(-1)^{j-1} \left[\Psi(X^{j-2} - X^{j-1})\Phi'(X^j - X^{j-1}) + \Psi(X^{j-1} - X^{j-2})\Phi'(X^{j-1} - X^j) \right] \big|
\end{aligned}$$

$$
\begin{aligned}
&= \big|\Psi'(X^j - X^{j+1})\Phi(X^{j+2} - X^{j+1}) + \Psi'(X^{j+1} - X^j)\Phi(X^{j+1} - X^{j+2}) \\
&\quad + \Psi'(X^{j-1} - X^j)\Phi(X^{j+1} - X^j) + \Psi(X^{j-1} - X^j)\Phi'(X^{j+1} - X^j) \\
&\quad + \Psi'(X^j - X^{j-1})\Phi(X^j - X^{j+1}) + \Psi(X^j - X^{j-1})\Phi'(X^j - X^{j+1}) \\
&\quad + \Psi(X^{j-2} - X^{j-1})\Phi'(X^j - X^{j-1}) + \Psi(X^{j-1} - X^{j-2})\Phi'(X^{j-1} - X^j)\big| \\
&\le 2\sqrt{54/e}\sqrt{2\pi e} + 2e\sqrt{e} \le 46.
\end{aligned}
$$

If $l \in P_{r+1}$, we have

$$
\sqrt{d_0}|\partial_l g_{\mathcal{P}}(\boldsymbol{y})| \;=\; \left[\Psi(X^{r-1} - X^r)\Phi'(X^{r+1} - X^r) + \Psi(X^r - X^{r-1})\Phi'(X^r - X^{r+1})\right] \le 2e\sqrt{e} \le 23.
$$

Consequently,

$$
\|\nabla g_{\mathcal{P}}(\boldsymbol{x})\|^2 \le r\frac{1}{d_0}d_0 46^2 + \frac{1}{d_0}d_0 23^2 \le 46^2(r+1).
$$

■

B.2. Proof of Lemma 7

Lemma 14 (Smoothness and boundness of $f_{\mathcal{P}}$) *The function $f_{\mathcal{P}}$ satisfies the following,*

1. *We have $f_{\mathcal{P}}(\mathbf{0}) - \inf_{\boldsymbol{x}} f_{\mathcal{P}}(\boldsymbol{x}) < 12r$.*
2. *For every $p \ge 1$, the p-th order derivatives of $f_{\mathcal{P}}$ are l_p-Lipschitz continuous, where $l_p \le \exp(cp\log p + c)$ for a numerical constant $c < \infty$.*

Proof For the first part, we have $f_{\mathcal{P}}(\mathbf{0}) = g_{\mathcal{P}}(\mathbf{0}) < 0$ and

$$
\inf_{\boldsymbol{x}\in\mathbb{R}^d} f_{\mathcal{P}}(\boldsymbol{x}) \ge \inf_{\boldsymbol{x}\in\mathbb{R}^d} g_{\mathcal{P}}(\rho(\boldsymbol{x})) \ge \inf_{\|\boldsymbol{x}\|\le R} g_{\mathcal{P}}(\boldsymbol{x}) \ge \inf_{\boldsymbol{x}\in\mathbb{R}^d} g_{\mathcal{P}}(\boldsymbol{x}) \ge -12r.
$$

For the second part, we first consider $\widetilde{g} : \mathbb{R}^d \to \mathbb{R}$ defined as

$$
\widetilde{g}(\boldsymbol{x}) = -\Psi(1)\Phi(\boldsymbol{x}_1) - \sum_{i=1}^{r}(-1)^i\left[\Psi(-\boldsymbol{x}_i)\Phi(\boldsymbol{x}_{i+1}) - \Psi(\boldsymbol{x}_i)\Phi(-\boldsymbol{x}_{i+1})\right].
$$

Then $g_{\mathcal{P}}(\boldsymbol{x}) = \widetilde{g}(PM\boldsymbol{x})$, where $M_p \in \mathbb{R}^{d\times d}$ is a permutation matrix respecting $\mathcal{P}$, such that for any $\boldsymbol{x}$ we have $M\boldsymbol{x} = [\boldsymbol{x}_{P_1}, \ldots, \boldsymbol{x}_{P_{r+1}}]^\top$ and $P \in \mathbb{R}^{d\times d}$ is a matrix for $X^i - X^{i-1}$ such that

$$
P_{i,j} = \begin{cases} 1/\sqrt{d_0} & i \in [r+1] \text{ and } (i-1)d_0 + 1 \le j \le id_0, \\ -1/\sqrt{d_0} & 2 \le i \le r+1 \text{ and } (i-2)d_0 + 1 \le j \le (i-1)d_0, \\ 0 & \text{otherwise.} \end{cases}
$$

We first bound the norm $(p+1)$-th order derivatives of $\widetilde{g}$. To do so, we first drop the coordinates from $r+2$ to d for $\widetilde{g}$, then fix a point $\boldsymbol{x} \in \mathbb{R}^{r+1}$ and a unit vector $\boldsymbol{v} \in \mathbb{R}^{r+1}$. We define the function $h_{\boldsymbol{x},\boldsymbol{v}} : \mathbb{R} \to \mathbb{R}$ as $h_{\boldsymbol{x},\boldsymbol{v}}(\theta) = \widetilde{g}(\boldsymbol{x} + \theta\boldsymbol{v})$. The function $\theta \mapsto h_{\boldsymbol{x},\boldsymbol{v}}(\theta)$ is infinitely differentiable for every $\boldsymbol{x}$ and $\boldsymbol{v}$. Therefore, $\widetilde{g}$ has l_p-Lipschitz p-th order derivatives if and only if $|h^{(p+1)}_{\boldsymbol{x},\boldsymbol{v}}(0)| \le l_p$ for every $\boldsymbol{x}$, $\boldsymbol{v}$. Using the shorthand notation $\partial_{i_1}\cdots\partial_{i_k}$ for $\frac{\partial^k}{\partial\boldsymbol{x}_{i_1}\cdots\partial\boldsymbol{x}_{i_k}}$, we have

$$
h^{(p+1)}_{\boldsymbol{x},\boldsymbol{v}}(0) = \sum_{i_1,\ldots,i_{p+1}=1}^{r+1} \partial_{i_1}\cdots\partial_{i_{p+1}}\widetilde{g}\,(\boldsymbol{x})\,\boldsymbol{v}_{i_1}\cdots\boldsymbol{v}_{i_{p+1}}.
$$

Observing that $\partial_{i_1} \cdots \partial_{i_{p+1}} \widetilde{g}$ is non-zero if and only if $|i_j - i_k| \leq 1$ for every $j, k \in [p+1]$. Consequently, we can rearrange the above summation as

$$h_{\boldsymbol{x},\boldsymbol{v}}^{(p+1)}(0) = \sum_{\delta_1,\delta_2,\ldots,\delta_p \in \{0,1\}^p \cup \{0,-1\}^p} \sum_{i=1}^{r+1} \partial_{i+\delta_1} \cdots \partial_{i+\delta_p} \partial_i \widetilde{g}\left(\boldsymbol{x}\right) \boldsymbol{v}_{i+\delta_1} \cdots \boldsymbol{v}_{i+\delta_p} \boldsymbol{v}_i, \tag{1}$$

where we let $\boldsymbol{v}_0 = \boldsymbol{v}_{r+2} = 0$. Recall that for all $x \leq 1/2$ and all $k \in [N]$, $\Psi^{(k)}(x) = 0$, $0 \leq \Psi, \Phi < \sqrt{2\pi e}$, and for all $k \in [N]$ we have

$$\sup_x |\Psi^{(k)}(x)| \leq \exp\left(\frac{5k}{2}\log(4k)\right) \text{ and } \sup_x |\Phi^{(k)}(x)| \leq \exp\left(\frac{3k}{2}\log\frac{3k}{2}\right).$$

Thus we have

$$\begin{aligned}\sup_{\boldsymbol{x}\in\mathbb{R}^{r+1}} \max_{i\in[r+1]} \max_{\delta\in\{0,1\}^p\cup\{0,-1\}^p} \left|\partial_{i+\delta_1}\cdots\partial_{i+\delta_p}\partial_i\widetilde{g}\left(\boldsymbol{x}\right)\right| &\leq \max_{k\in[p+1]}\left\{2\sup_{x\in\mathbb{R}}\left|\Psi^{(k)}(x)\right|\sup_{x'\in\mathbb{R}}\left|\Phi^{(p+1-k)}(x')\right|\right\} \\ &\leq 2\sqrt{2\pi e}\cdot e^{2.5(p+1)\log(4(p+1))} \leq \exp\left(2.5p\log p + 4p + 9\right).\end{aligned} \tag{2}$$

We further claim that $\left|\sum_{i=1}^{r} \boldsymbol{v}_{i+\delta_1}\cdots\boldsymbol{v}_{i+\delta_p}\boldsymbol{v}_i\right| \leq 1$ for every $\delta \in \{0,1\}^p \cup \{0,-1\}^p$. When $\delta = \boldsymbol{0}$, we have $\left|\sum_{i=1}^{r+1} \boldsymbol{v}_{i+\delta_1}\cdots\boldsymbol{v}_{i+\delta_p}\boldsymbol{v}_i\right| \leq \left|\sum_{i=1}^{r+1} \boldsymbol{v}_i^p\right| \leq \left|\sum_{i=1}^{r+1} \boldsymbol{v}_i^2\right| = 1$. Otherwise, let $1 \leq \sum_{j=1}^p |\delta_j| = n \leq p$, by the Cauchy-Swartz inequality, we have, for $s \in \{\pm 1\}$,

$$\left|\sum_{i=1}^{r} \boldsymbol{v}_{i+\delta_1}\cdots\boldsymbol{v}_{i+\delta_p}\boldsymbol{v}_i\right| = \left|\sum_{i=1}^{r+1} \boldsymbol{v}_i^{p+1-n}\boldsymbol{v}_{i+s}^n\right| \leq \sqrt{\sum_{i=1}^{r+1}\boldsymbol{v}_i^{2(p+1-n)}}\sqrt{\sum_{i=1}^{r+1}\boldsymbol{v}_{i+s}^{2n}} \leq \sum_{i=1}^{r+1} v_i^2 = 1.$$

Combining Eq. (1), and Eq. (2), we have

$$\begin{aligned} h_{\boldsymbol{x},\boldsymbol{v}}^{(p+1)}(0) &= \sum_{\delta_1,\delta_2,\ldots,\delta_p \in \{0,1\}^p \cup \{0,-1\}^p} \sum_{i=1}^{r+1} \partial_{i+\delta_1}\cdots\partial_{i+\delta_p}\partial_i\widetilde{g}\left(\boldsymbol{x}\right)\boldsymbol{v}_{i+\delta_1}\cdots\boldsymbol{v}_{i+\delta_p}\boldsymbol{v}_i \\ &\leq \sum_{\delta_1,\delta_2,\ldots,\delta_p \in \{0,1\}^p \cup \{0,-1\}^p} \exp\left(2.5p\log p + 4p + 9\right)\sum_{i=1}^{r+1}\boldsymbol{v}_{i+\delta_1}\cdots\boldsymbol{v}_{i+\delta_p}\boldsymbol{v}_i \\ &\leq \sum_{\delta_1,\delta_2,\ldots,\delta_p \in \{0,1\}^p \cup \{0,-1\}^p} \exp\left(2.5p\log p + 4p + 9\right) \\ &\leq (2^{p+1}-1)\exp\left(2.5p\log p + 4p + 9\right) \leq e^{2.5p\log p + 5p + 10}. \end{aligned}$$

We further bound the operator norm of $PM \in \mathbb{R}^{d\times d}$ as

$$\|PM\|_{\mathrm{op}} \leq \|P\|_{\mathrm{op}}\|M\|_{\mathrm{op}} \leq \sup_{\boldsymbol{v}:\|\boldsymbol{v}\|=1}\|A\boldsymbol{v}\| \leq \sum_{i=1}^{2d_0}\frac{1}{\sqrt{d_0}}\frac{1}{\sqrt{2d_0}} = \sqrt{2} \leq 2.$$

Thus $\left\|\nabla^{(p+1)}g_{\mathcal{P}}\right\|_{\mathrm{op}} \leq \left\|\nabla^{(p+1)}\widetilde{g}\right\|_{\mathrm{op}}\|PM\|_{\mathrm{op}} \leq e^{2.5p\log p + 5p + 10}2^{p+1} \leq e^{2.5p\log p + 6p + 11}$.

Finally, we bound $f_{\mathcal{P}} = g_{\mathcal{P}} \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2$. Since $\frac{1}{5}\left\|\cdot\right\|^2$ is $2/5$-Lipschitz first derivative and 0-Lipschitz higher order derivatives, we only consider $\widetilde{f} = g_{\mathcal{P}} \circ \rho$.

To apply Faà di Bruno formula to calculate high derivatives of $\widetilde{f}$, we first define $\mathcal{P}_k$ to be the set of all partitions of $[k] = \{1, \ldots, k\}$, *i.e.,* $(S_1, \ldots, S_L) \in \mathcal{P}_k$ if and only if the S_i are disjoint and $\cup_l S_l = [k]$. Then we have

$$\nabla^k i_1, \ldots, i_k [\widetilde{f}\,] = \sum_{(S_1,\ldots,S_L)\in\mathcal{P}_k} \sum_{j_1,\ldots,j_L=1}^{r+1} \left(\prod_{l=1}^{L} \nabla^{|S_l|}_{i_{S_l}} [\rho_{j_l}]\right) \nabla^L_{j_1,\ldots,j_L} [g_{\mathcal{P}}] \circ \rho,$$

where we have used the shorthand $\nabla^{|S|}_{i_S}$ to denote the partial derivatives with respect to each of x_{i_j} for $j \in S$. Let $\boldsymbol{v} \in \mathbb{R}^d$ be an unit arbitrary direction vector with $\|\boldsymbol{v}\| = 1$. Then for any $j \in [d]$ and $k \in [N]$ and fixed $\boldsymbol{x}$, we define $\widetilde{\boldsymbol{v}}^k_j = \left\langle \nabla^k [\rho_j] (\boldsymbol{x}), \boldsymbol{v}^{\otimes k} \right\rangle$. Then we have

$$\begin{aligned}
\left\langle \nabla^k [\widetilde{f}\,] (\boldsymbol{x}), \boldsymbol{v}^{\otimes k} \right\rangle &= \sum_{(S_1,\ldots,S_L)\in\mathcal{P}_k} \sum_{i_1,\ldots,i_k=1}^{d} \boldsymbol{v}_{i_1} \ldots \boldsymbol{v}_{i_k} \sum_{j_1,\ldots,j_L=1}^{r+1} \left(\prod_{l=1}^{L} \nabla^{|S_l|}_{i_{S_l}} [\rho_{j_l}] (\boldsymbol{x})\right) \nabla^L_{j_1,\ldots,j_L} [g_{\mathcal{P}}] \circ \rho(\boldsymbol{x}) \\
&= \sum_{(S_1,\ldots,S_L)\in\mathcal{P}_k} \sum_{j_1,\ldots,j_L=1}^{r+1} \widetilde{\boldsymbol{v}}^{|S_1|}_{j_1} \ldots \widetilde{\boldsymbol{v}}^{|S_L|}_{j_L} \nabla^L_{j_1,\ldots,j_L} [g_{\mathcal{P}}] \circ \rho(\boldsymbol{x}) \\
&= \sum_{(S_1,\ldots,S_L)\in\mathcal{P}_k} \left\langle \nabla^L [g_{\mathcal{P}}] \circ \rho(\boldsymbol{x}), \widetilde{\boldsymbol{v}}^{|S_1|} \ldots \widetilde{\boldsymbol{v}}^{|S_L|} \right\rangle.
\end{aligned}$$

We then recall the fact about $\widetilde{\boldsymbol{v}}^k(\boldsymbol{x})$ shown in (Carmon et al., 2020, Section B.4) as follows.

Fact 15 *There exists a numerical constant $c < \infty$ such that for all $k \in [N]$,*

$$\sup_{\boldsymbol{x}} \left\| \widetilde{\boldsymbol{v}}^k(\boldsymbol{x}) \right\| \leq \exp(ck \log k + c) R^{1-k}.$$

Then, combining the fact $\|\nabla g_{\mathcal{P}}(\boldsymbol{x})\| \leq 46\sqrt{r+1}$ and $\left\|\nabla^{(p+1)} g_{\mathcal{P}}\right\|_{\text{op}} \leq e^{2.5p\log p + 6p + 11}$, we have

$$\begin{aligned}
&\left| \left\langle \nabla^{(p+1)} [\widetilde{f}\,] (\boldsymbol{x}), \boldsymbol{v}^{\otimes(p+1)} \right\rangle \right| \\
\leq& \sum_{(S_1,\ldots,S_L)\in\mathcal{P}_{p+1}} \left\| \nabla^L [g_{\mathcal{P}}] \circ \rho(\boldsymbol{x}) \right\|_{\text{op}} \prod_{l=1}^{L} \left\| \widetilde{\boldsymbol{v}}^{|S_l|} \right\| \\
\leq& \sum_{(S_1,\ldots,S_L)\in\mathcal{P}_{p+1}} 46\sqrt{r+1} e^{2.5p\log p + 6p + 11} \prod_{l=1}^{L} \exp(c|S_l| \log|S_l| + c) R^{1-|S_l|} \\
\leq& e^{(p+1)\log(p+1)} 46\sqrt{r+1} e^{2.5p\log p + 6p + 11} e^{c(p+1)\log(p+1)+c} (230\sqrt{r+1})^{-p} \\
\leq& e^{c' p \log p + c'}.
\end{aligned}$$

for a numerical constant $c' < \infty$ and the penultimate inequality follows from the fact that $R = 230\sqrt{r+1}$, $q(x) = (x+1)\log(x+1)$ satisfies $q(x) + q(y) \leq q(x+y)$ for every $x, y > 0$ and there are at most $\exp(k \log k)$ partitions in $\mathcal{P}_k$. ■

B.3. Proof of Lemma 8

Lemma 16 (Characterization of output) *If there exits a constant $\alpha = \omega(1)$ such that $8R\sqrt{\frac{\alpha \log d}{d_0}} \le \frac{1}{2}$, then for any randomized algorithm A, any $\tau \le r$, and any initial point $\boldsymbol{x}^0$, $X(\mathsf{A}[f_{\mathcal{P}}, \boldsymbol{x}^0, \tau]) = (X^1(\rho(\boldsymbol{x})), \ldots, X^{r+1}(\rho(\boldsymbol{x})))$ takes a form as*

$$(x_1, \ldots, x_{2\tau}, x_{2\tau+1}, x_{2\tau+1}, \ldots, x_{2\tau+1}),$$

up to addictive error $1/4$ for every coordinate with probability $1 - d^{-\omega(1)}$ over $\mathcal{P}$.

Proof We fixed τ and prove the following by induction for $l \in [\tau]$: With high probability, the computation path of the (deterministic) algorithm A and the queries it issues in the l-th round are determined by $P_1, \ldots, P_{2l-2}$.

As a first step, we assume the algorithm is deterministic by fixing its random bits and choose the partition of $\mathcal{P}$ uniformly at random.

To prove the inductive claim, let $\mathcal{E}_l$ denote the event that for any query $\boldsymbol{x}$ issued by A in iteration l, the answer is in the form $g^l_{\mathcal{P}}(\rho(\boldsymbol{x})) + \frac{1}{5}\|\boldsymbol{x}\|^2$ where $g^l_{\mathcal{P}} : \mathbb{R}^d \to \mathbb{R}$ defined as:

$$g^l_{\mathcal{P}}(\boldsymbol{x}) = -\Psi(1)\Phi(X^1) - \sum_{i=1}^{2l-1}(-1)^i \left[\Psi(X^{i-1} - X^i)\Phi(X^{i+1} - X^i) - \Psi(X^i - X^{i-1})\Phi(X^i - X^{i+1})\right],$$

i.e., $\mathcal{E}_l$ represents the events that $\forall \boldsymbol{x} \in Q^l$, $f_{\mathcal{P}}(\boldsymbol{x}) = g_{\mathcal{P}}(\rho(\boldsymbol{x})) + \frac{1}{5}\|\boldsymbol{x}\|^2 = g^l_{\mathcal{P}}(\rho(\boldsymbol{x})) + \frac{1}{5}\|\boldsymbol{x}\|^2$. This implies that the answers depend only on $P_1, \cdots, P_{2l}$.

Since the queries in round l depend only on $P_1, \ldots, P_{2l-2}$, if $\mathcal{E}_l$ occurs, the entire computation path in round l is determined by $P_1, \ldots, P_{2l}$. By induction, we conclude that if all of $\mathcal{E}_1, \ldots, \mathcal{E}_l$ occur, the computation path in round l is determined by $P_1, \ldots, P_{2l}$.

Now we analyze the conditional probability $\mathbb{P}[\mathcal{E}_l \mid \mathcal{E}_1, \ldots, \mathcal{E}_{l-1}]$. Given all of $\mathcal{E}_1, \ldots, \mathcal{E}_{l-1}$ occur so far, we can claim that Q^l is determined by $P_1, \ldots, P_{2l-2}$. Conditioned on $P_1, \ldots, P_{2l-2}$, the partition of $[d] \setminus \bigcup_{i\in[2l-2]} P_i$ is uniformly random. We consider $\{0,1\}$-random variable Y_j, $j \in [d] \setminus \bigcup_{i\in[2l-2]} P_i$. For any query $\boldsymbol{x} \in \mathbb{R}^d$, we represent $X^i(\rho(\boldsymbol{x}))$ as a linear function of Y_is as $X^i(\rho(\boldsymbol{x})) = \frac{1}{\sqrt{d_0}} \sum_{j\in[d]\setminus\bigcup_{i\in[2l-2]} P_i} Y_j \rho_j(\boldsymbol{x})$ such that $Y_i = 1$ if $Y_i \in P_i$ and $Y_i = 0$ otherwise. By the concentration of linear functions over the Boolean slice (Theorem 12), and set $t = 8R\sqrt{\frac{\alpha \log d}{d_0}}$ with $\alpha = \omega(1)$, we have

$$\mathbb{P}_{\mathcal{P}}\left[|X^i(\rho(\boldsymbol{x})) - \mathbb{E}[X^i(\rho(\boldsymbol{x}))]| \ge \frac{t}{2}\right] \le 2\exp\left(-\frac{t^2}{16\sum_i \rho_i^2(\boldsymbol{x})/d_0}\right) = 2d^{-\omega(1)}.$$

Similarly, $\mathbb{P}_{\mathcal{P}}\left[|X^{i+1}(\rho(\boldsymbol{x})) - \mathbb{E}[X^{i+1}(\rho(\boldsymbol{x}))]| \ge \frac{t}{2}\right] \le 2d^{-\omega(1)}$. Combining the fact that $\mathbb{E}[X^i(\rho(\boldsymbol{x}))] = \mathbb{E}[X^{i+1}(\rho(\boldsymbol{x}))]$, when $t \le \frac{1}{2}$ we have with probability at least $1 - d^{-\omega(1)}$, for any fixed $i \ge 2l$

$$\left|X^i(\rho(\boldsymbol{x})) - X^{i+1}(\rho(\boldsymbol{x}))\right| \le t \le \frac{1}{2}.$$

Combining the fact that for all $x \le 1/2$ and $k \in \mathbb{N}$, $\Psi^{(k)}(x) = 0$, we have $g_{\mathcal{P}}(\boldsymbol{x}') = g^l_{\mathcal{P}}(\boldsymbol{x}')$ with a probability at least $1 - rd^{-\omega(1)}$.

By union bound over all queries $\boldsymbol{x} \in Q^l$, conditioned on that $\mathcal{E}_1, \dots, \mathcal{E}_{l-1}$ occur, with probability at least $1 - r\mathsf{poly}(d)d^{-\omega(1)}$, $\mathcal{E}_l$ occurs. Therefore by induction,

$$\begin{aligned}\mathbb{P}(\mathcal{E}_l) &= \mathbb{P}(\mathcal{E}_l|\mathcal{E}_1, \dots, \mathcal{E}_{l-1})\mathbb{P}(\mathcal{E}_{l-1}|\mathcal{E}_1, \dots, \mathcal{E}_{l-2})\dots\mathbb{P}(\mathcal{E}_2|\mathcal{E}_1)\mathbb{P}(\mathcal{E}_1)\\ &\geq 1 - r^2\mathsf{poly}(d)d^{-\omega(1)} = 1 - d^{-\omega(1)}.\end{aligned}$$

This implies that with high probability, the computation path in round l is determined by $P_1, \dots, P_{2l}$. Consequently, for all $l \in [\tau]$ a solution returned after $l-1$ rounds is determined by $P_1, \dots, P_{2l-2}$ with high probability. By the same concentration argument, for any $t \geq \tau+1$, we have $|X^t(\rho(\boldsymbol{x})) - \mathbb{E}[X^t(\rho(\boldsymbol{x}))]| < \frac{1}{4}$ with probability $1 - d^{-\omega(1)}$ over $\mathcal{P}$.

Finally, we note that by allowing the algorithm to use random bits, the results are a convex combination of the bounds above, so the same high-probability bounds are satisfied. ■

B.4. Proof of Lemma 9

Lemma 17 (Small weighted partition implies large gradient norm) *For any $\boldsymbol{x} \in \mathbb{R}^d$, if $|X^{r+1}(\rho(\boldsymbol{x})) - X^r(\rho(\boldsymbol{x}))| < 1$, then $\|\nabla f_{\mathcal{P}}(\boldsymbol{x})\| \geq 0.08$.*

Proof We let $\boldsymbol{y} = \rho(\boldsymbol{x})$ and take $j \leq r+1$ to be the smallest j for which $|X^j(\boldsymbol{y}) - X^{j-1}(\boldsymbol{y})| < 1$, so that $|X^{j-1}(\boldsymbol{y}) - X^{j-2}(\boldsymbol{y})| \geq 1$, where we let $X^0 \equiv 0$ and $X^{-1} \equiv 1$. We first show a slight stronger argument for the gradient of $g_{\mathcal{P}}$. Specifically, we will show there exits a direction vector $\boldsymbol{u} \in \mathbb{R}^d$ with $\|\boldsymbol{u}\| = 1$ satisfies

$$|\langle \boldsymbol{u}, \boldsymbol{y}\rangle| < 1 \quad \text{and} \quad |\langle \boldsymbol{u}, \nabla g_{\mathcal{P}}(\boldsymbol{y})\rangle| > \frac{1}{\sqrt{2}}.$$

To show it, we consider the $j = 1$, $1 < j < r+1$ and $j = r+1$ separately.

Case (i): $j = 1$**.** For $l \in P_1$, we have

$$\begin{aligned}&\sqrt{d_0}\partial_l g_{\mathcal{P}}(\boldsymbol{y})\\ =& -\Psi(1)\Phi'(X^1) - \Psi'(-X^1)\Phi(X^2 - X^1) - \Psi(-X^1)\Phi'(X^2 - X^1)\\ &-\Psi'(X^1)\Phi(X^1 - X^2) - \Psi(X^1)\Phi'(X^1 - X^2)\\ &-\Psi'(X^1 - X^2)\Phi(X^3 - X^2) - \Psi'(-X^1 + X^2)\Phi(X^2 - X^3)\\ \leq& -\Psi(1)\Phi'(X^1).\end{aligned}$$

Now we choose the direction vector $\boldsymbol{u}$ as $u_l = \begin{cases}\frac{1}{\sqrt{d_0}} & l \in P_1,\\ 0 & \text{otherwise,}\end{cases}$ and combining the fact that for all $x \geq 1$ and $|y| < 1$, $\Psi(x)\Phi'(y) > 1$, we have $|\langle \boldsymbol{u}, \boldsymbol{y}\rangle| = |X^1(\boldsymbol{y})| < 1$ and

$$|\langle \boldsymbol{u}, \nabla g_{\mathcal{P}}(\boldsymbol{y})\rangle| = \left\|\frac{1}{\sqrt{d_0}}\sum_{l\in P_1}\partial_l g_{\mathcal{P}}(\boldsymbol{y})\right\| \geq 1.$$

Case (ii): $1 < j < r+1$. For $l \in P_j$, we have

$$\begin{aligned}
&\sqrt{d_0}\partial_l g_{\mathcal{P}}(\boldsymbol{y}) \\
=& -(-1)^{j+1}\left[\Psi'(X^j - X^{j+1})\Phi(X^{j+2} - X^{j+1}) + \Psi'(X^{j+1} - X^j)\Phi(X^{j+1} - X^{j+2})\right] \\
&-(-1)^j\left[-\Psi'(X^{j-1} - X^j)\Phi(X^{j+1} - X^j) - \Psi(X^{j-1} - X^j)\Phi'(X^{j+1} - X^j)\right. \\
&\left.-\Psi'(X^j - X^{j-1})\Phi(X^j - X^{j+1}) - \Psi(X^j - X^{j-1})\Phi'(X^j - X^{j+1})\right] \\
&-(-1)^{j-1}\left[\Psi(X^{j-2} - X^{j-1})\Phi'(X^j - X^{j-1}) + \Psi(X^{j-1} - X^{j-2})\Phi'(X^{j-1} - X^j)\right] \\
=& (-1)^j\left[\Psi'(X^j - X^{j+1})\Phi(X^{j+2} - X^{j+1}) + \Psi'(X^{j+1} - X^j)\Phi(X^{j+1} - X^{j+2})\right. \\
&+\Psi'(X^{j-1} - X^j)\Phi(X^{j+1} - X^j) + \Psi(X^{j-1} - X^j)\Phi'(X^{j+1} - X^j) \\
&+\Psi'(X^j - X^{j-1})\Phi(X^j - X^{j+1}) + \Psi(X^j - X^{j-1})\Phi'(X^j - X^{j+1}) \\
&\left.+\Psi(X^{j-2} - X^{j-1})\Phi'(X^j - X^{j-1}) + \Psi(X^{j-1} - X^{j-2})\Phi'(X^{j-1} - X^j)\right].
\end{aligned}$$

In this case, combining the fact that $\Psi(x) = 0,\ \forall x < 1/2$ and the fact that for all $x \geq 1$ and $|y| < 1$, $\Psi(x)\Phi'(y) > 1$, we have $\operatorname{sign}(\partial_l g_{\mathcal{P}}(\boldsymbol{y})) = (-1)^j$ and

$$\begin{aligned}
\sqrt{d_0}|\partial_l g_{\mathcal{P}}(\boldsymbol{y})| &\geq |\Psi(X^{j-2} - X^{j-1})\Phi'(X^j - X^{j-1}) + \Psi(X^{j-1} - X^{j-2})\Phi'(X^{j-1} - X^j)| \\
&= |\Psi(|X^{j-2} - X^{j-1}|)\Phi'((X^j - X^{j-1}) \cdot \operatorname{sign}(X^{j-2} - X^{j-1}))| \\
&\geq 1.
\end{aligned}$$

Furthermore, for $l \in P_{j-1}$, we have $\operatorname{sign}(\partial_l g_{\mathcal{P}}(\boldsymbol{y})) = (-1)^{j-1}$. Thus if we choose the direction vector $\boldsymbol{u}$ as $\boldsymbol{u}_l = \begin{cases} \frac{1}{\sqrt{2d_0}} & l \in P_j, \\ -\frac{1}{\sqrt{2d_0}} & l \in P_{j-1}, \\ 0 & \text{otherwise}, \end{cases}$ we have $|\langle \boldsymbol{u}, \boldsymbol{y}\rangle| = \frac{1}{\sqrt{2}}|X^j(\boldsymbol{y}) - X^{j-1}(\boldsymbol{y})| < 1$ and

$$|\langle \boldsymbol{u}, \nabla g_{\mathcal{P}}(\boldsymbol{y})\rangle| \geq \left\| \frac{1}{\sqrt{2d_0}} \sum_{l \in P_j} \partial_l g_{\mathcal{P}}(\boldsymbol{y}) \right\| \geq \frac{1}{\sqrt{2}}.$$

Case (iii): $j = r+1$. For $l \in P_{r+1}$, we have

$$\sqrt{d_0}\partial_l g_{\mathcal{P}}(\boldsymbol{y}) = -(-1)^r\left[\Psi(X^{r-1} - X^r)\Phi'(X^{r+1} - X^r) + \Psi(X^r - X^{r-1})\Phi'(X^r - X^{r+1})\right].$$

Similarly, we have $\operatorname{sign}(\partial_l g_{\mathcal{P}}(\boldsymbol{y})) = (-1)^{r+1}$ and

$$\sqrt{d_0}|\partial_l g_{\mathcal{P}}(\boldsymbol{y})| = |\Psi(X^{r-1} - X^r)\Phi'(X^{r+1} - X^r) + \Psi(X^r - X^{r-1})\Phi'(X^r - X^{r+1})| \geq 1.$$

Furthermore, for $l \in P_r$, we have $\operatorname{sign}(\partial_l g_{\mathcal{P}}(\boldsymbol{y})) = (-1)^r$. Thus if we choose the direction vector $\boldsymbol{u}$ as $\boldsymbol{u}_l = \begin{cases} \frac{1}{\sqrt{2d_0}} & l \in P_{r+1}, \\ -\frac{1}{\sqrt{2d_0}} & l \in P_r, \\ 0 & \text{otherwise}, \end{cases}$ we have $|\langle \boldsymbol{u}, \boldsymbol{y}\rangle| = \frac{1}{\sqrt{2}}|X^{r+1}(\boldsymbol{y}) - X^r(\boldsymbol{y})| < 1$ and

$$|\langle \boldsymbol{u}, \nabla g_{\mathcal{P}}(\boldsymbol{y})\rangle| \geq \left\| \frac{1}{\sqrt{2d_0}} \sum_{l \in P_{r+1}} \partial_l g_{\mathcal{P}}(\boldsymbol{y}) \right\| \geq \frac{1}{\sqrt{2}}.$$

Next, we present the following lemma to address the unbounded domain, as discussed in Section 5.2 of (Carmon et al., 2020), and provide its proof here for the reader's convenience.

Lemma 18 *Assume functions $g : \mathbb{R}^d \to \mathbb{R}$ with $\|\nabla g(\boldsymbol{y})\| \leq 46\sqrt{r+1}$ for any $\boldsymbol{y} \in \mathbb{R}^d$ and $\rho : \mathbb{R}^d \to \mathbb{R}^d$ defined as $\rho(\boldsymbol{x}) = \frac{\boldsymbol{x}}{\sqrt{1+\|\boldsymbol{x}\|^2/R^2}}$ and $R = 230\sqrt{r+1}$. For any point $\boldsymbol{x} \in \mathbb{R}^d$ such that $\rho(\boldsymbol{x}) = \boldsymbol{y}$ if there exits a direction vector $\boldsymbol{u} \in \mathbb{R}^d$ with $\|\boldsymbol{u}\| = 1$ satisfies*

$$|\langle \boldsymbol{u}, \boldsymbol{y}\rangle| < 1 \quad \textit{and} \quad |\langle \boldsymbol{u}, \nabla g(\boldsymbol{y})\rangle| \geq \frac{1}{\sqrt{2}},$$

then

$$\left\|\nabla\big(g \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2\big)(\boldsymbol{x})\right\| \geq 0.08.$$

Proof By $\frac{\partial \rho}{\partial \boldsymbol{x}}(\boldsymbol{x}) = \frac{I - \rho(\boldsymbol{x})\rho(\boldsymbol{x})^\top/R^2}{\sqrt{1+\|\boldsymbol{x}\|^2/R^2}}$, we can calculate $\langle \boldsymbol{u}, \nabla(g \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2)(\boldsymbol{x})\rangle$ as

$$\begin{aligned}
&\left\langle \boldsymbol{u}, \nabla\big(g \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2\big)(\boldsymbol{x})\right\rangle \\
&= \left\langle \boldsymbol{u}, \frac{\partial \rho}{\partial \boldsymbol{x}}(\boldsymbol{x})\nabla g(\rho(\boldsymbol{x}))\right\rangle + \frac{2}{5}\langle \boldsymbol{u}, \boldsymbol{x}\rangle \\
&= \frac{\langle \boldsymbol{u}, \nabla g(\rho(\boldsymbol{x}))\rangle - \langle \boldsymbol{u}, \rho(\boldsymbol{x})\rangle\langle \rho(\boldsymbol{x}), \nabla g(\rho(\boldsymbol{x}))\rangle/R^2}{\sqrt{1+\|\boldsymbol{x}\|^2/R^2}} + \frac{2}{5}\langle \boldsymbol{u}, \rho(\boldsymbol{x})\rangle\sqrt{1+\|\boldsymbol{x}\|^2/R^2}.
\end{aligned}$$

Case (i): $\|\boldsymbol{x}\| \leq R/2$. In this case, since $\|\rho(\boldsymbol{x})\| \leq \|\boldsymbol{x}\| \leq R/2$, we have

$$\begin{aligned}
&\left\|\nabla\big(g \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2\big)(\boldsymbol{x})\right\| \\
&\geq \left|\left\langle \boldsymbol{u}, \nabla\big(g \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2\big)(\boldsymbol{x})\right\rangle\right| \\
&\geq \frac{2}{\sqrt{5}}\left|\left\langle \boldsymbol{u}, \nabla g(\rho(\boldsymbol{x}))\right\rangle\right| - |\langle \boldsymbol{u}, \rho(\boldsymbol{x})\rangle|\left(\frac{\|\nabla g(\rho(\boldsymbol{x}))\|}{2R} + \frac{1}{\sqrt{5}}\right) \\
&\geq \frac{2}{\sqrt{5}}\left|\left\langle \boldsymbol{u}, \nabla g(\boldsymbol{y})\right\rangle\right| - |\langle \boldsymbol{u}, \boldsymbol{y}\rangle|\left(\frac{R/5}{2R} + \frac{1}{\sqrt{5}}\right) \\
&\geq \frac{\sqrt{2}}{\sqrt{5}} - \frac{1}{10} - \frac{1}{\sqrt{5}} \geq 0.08.
\end{aligned}$$

Case (ii): $\|\boldsymbol{x}\| > R/2$. In this case, we have $\left\|\frac{\partial \rho}{\partial \boldsymbol{x}}(\boldsymbol{x})\right\|_{\mathrm{op}} \leq \frac{1}{\sqrt{1+\|\boldsymbol{x}\|^2/R^2}} \leq \frac{2}{\sqrt{5}}$ and that $\|\nabla g(\rho(\boldsymbol{x}))\| \leq R/5$. Thus we have

$$\left\|\nabla\big(g \circ \rho + \frac{1}{5}\left\|\cdot\right\|^2\big)(\boldsymbol{x})\right\| \geq \frac{2}{5}\left\|\boldsymbol{x}\right\| - \left\|\frac{\partial \rho}{\partial \boldsymbol{x}}(\boldsymbol{x})\right\|_{\mathrm{op}}\|\nabla g(\rho(\boldsymbol{x}))\| \geq \frac{R}{5} - \frac{2}{\sqrt{5}}\frac{R}{5} > 0.02R \geq \sqrt{r+1} \geq 1.$$

■

■

Appendix C. Missing details for constant dimensional lower bound

We first discuss how to reduce the problem of finding ϵ-stationary points to finding local minimum of a monotone path function defined over the grid, following (Bubeck and Mikulincer, 2020).

Specifically, we consider grid graph $G_{n,d} = (V_{n,d}, E_{n,d})$ as $V_{n,d} = \{0, \dots, n\}^d$ and $E_{n,d} = \{(\boldsymbol{u}, \boldsymbol{v}) \in V_{n,d} \times V_{n,d} : \|\boldsymbol{u} - \boldsymbol{v}\|_1 \leq 1\}$. A sequence of vertices $(\boldsymbol{v}^0, \dots, \boldsymbol{v}^n)$ is called a *monotone path* in $G_{n,d}$ if $\boldsymbol{v}^0 = \mathbf{0}$ and for every $1 \leq i \leq n$, $\boldsymbol{v}^i - \boldsymbol{v}^{i-1}$ equals $\boldsymbol{e}_j$ for some $j \in [d]$. In other words, the path starts at the origin and progresses by incrementing exactly one coordinate at each step. If $(\boldsymbol{v}^0, \dots, \boldsymbol{v}^n)$ is a monotone path, we associate to it a *monotone path function* $P : V_{n,d} \to \mathbb{R}$ defined as

$$P(\boldsymbol{v}) = \begin{cases} -\|\boldsymbol{v}\|_1 & \text{if } \boldsymbol{v} \in \{\boldsymbol{v}^0, \dots, \boldsymbol{v}^n\}, \\ \|\boldsymbol{v}\|_1 & \text{otherwise.} \end{cases}$$

For convenience, we may sometimes refer to the path function P and the path $(\boldsymbol{v}_0, \dots, \boldsymbol{v}_n)$ interchangeably. For $i = 0, \dots, n$, we write $P_{i,d}$ for the set $P^{-1}(-i)$. We denote the set of all monotone path functions on $G_{n,d}$ by $\mathrm{F}_{n,d}$. It is clear that if $P \in \mathrm{F}_{n,d}$ then $P_{n,d}$ is the only local minimum of P and hence the global minimum. For any $\boldsymbol{v} \in V_{n,d}$, we define

$$\mathrm{square}(\boldsymbol{v}) = \bigotimes_{s=1}^{d} \left[\frac{\boldsymbol{v}_s}{n+1}, \frac{\boldsymbol{v}_s + 1}{n+1}\right] \subset [0,1]^d.$$

The following lemma allows to construct a smooth function that preserves the structure of any monotone path function.

Lemma 19 ((Vavasis, 1993, Section 3)) *Fix $\varepsilon > 0$ and let $n(\varepsilon)$ depends on ε. For any $P \in \mathrm{F}_{n(\varepsilon),d}$, there exists a function $\widehat{P} : [0,1]^d \to \mathbb{R}$ with the following properties:*

1. *$\widehat{P}$ is smooth.*

2. *$\widehat{P} = f_P + \ell$, where ℓ is a linear function, which does not depend on P, and $\mathrm{supp}(f_P) \subset \bigcup_{i=0}^{n} \mathrm{square}\,(P_i)$, where $\mathrm{supp}(\phi) = \{\boldsymbol{x} : \phi(\boldsymbol{x}) \neq 0\}$.*

3. *If $\boldsymbol{x} \in [0,1]^d$ is an ε-stationary point of $\widehat{P}$ then $\boldsymbol{x} \in \mathrm{square}\left(P_{n(\varepsilon),d}\right)$.*

4. *if $P' \in \mathrm{F}_{n(\varepsilon),d}$ is another function and for some $i = 0, ..., n$, $(P'_{i-1,d}, P'_{i,d}, P'_{i+1,d}) = (P_{i-1,d}, P_{i,d}, P_{i+1,d})$. Then $\widehat{P}'|_{\mathrm{square}\left(P_{i,d}\right)} = \widehat{P}|_{\mathrm{square}\left(P_{i,d}\right)}$.*

5. *Assume the length of path is k then $n(\varepsilon)$ satisfies $-\varepsilon \cdot k/n(\varepsilon) + 1/n(\varepsilon)^2 \geq 0$.*

In particular, because $k \leq dn(\varepsilon)$, then $n(\varepsilon) \leq 1/\sqrt{d\varepsilon}$. We now restate the result in Section 6 of (Bubeck and Mikulincer, 2020) that finding the minimum of P is as hard as finding an ε-stationary point of $\widehat{P}$. To this end, we give formal definition of problem of finding a local minimum of a function defined on a graph.

Finding local minimum problem. Given a graph $G = (V, E)$ and oracle access to a function $f : V \to \mathbb{R}$, the goal is to find a vertex v that is a local minimum, i.e., $f(v) \le f(u)$ for all $(u, v) \in E$. To compare the difficulty of different learning problems, we leverage the *round-preserving* reduction introduced in (Brânzei and Li, 2022).

Definition 20 (Round-preserving reduction) *A reduction from oracle-based problem P_1 to oracle-based problem P_2 is round-preserving if for any instance of problem P_1 with oracle $\mathcal{O}_1$, the instance of problem P_2 with oracle $\mathcal{O}_2$ given by the reduction satisfies that*

1. *A solution of the P_1 instance can be obtained from any solution of the P_2 instance without any more queries on $\mathcal{O}_1$.*
2. *Each query to $\mathcal{O}_2$ can be answered by a constant number of queries to $\mathcal{O}_1$ in one round.*

The following lemma showed that finding the minimum of P is as hard as finding an ε-stationary point of $\widehat{P}$.

Lemma 21 (Section 6 in (Bubeck and Mikulincer, 2020)) *There is a round-preserving reduction such that any instance of finding ε-stationary point of $\widehat{P}$ is reduced to the instance of finding local minimum of a monotone path function P. Specifically, for any algorithm, the complexity of finding ε-stationary point of $\widehat{P}$ is no smaller than $1/(2d+1)$ times the complexity of finding local minimum of $P \in F_{n(\varepsilon),d}$.*

Lower bound for monotone path functions. By Lemma 21, the remaining task is to find the number of required queries in each number for finding local minimum for monotone path functions, which is obtained by Brânzei and Li (2022) for constant iteration case. Furthermore, we also find their construction also work for algorithm with $\Theta(\log(1/\varepsilon))$ iterations. We summarize this results in the following lemma.

Lemma 22 (Adaptive complexity of finding local minimum (Brânzei and Li, 2022)) *Assume For any (possible randomized) algorithm running within k-round, and issuing $n^{\frac{d^{k+1}-d^k}{d^k-1}}/(20d \cdot k)$ queries per round fails to find local minimum for monotone path function with length at most nd with probability at least $7/40$.*

Combining Lemma 21 and Lemma 22 yields the desired result in Theorem 11. We note $n(\varepsilon) \le 1/\sqrt{d\varepsilon}$ here.

Appendix D. Constant iteration gradient flow trapping

In this section, we will prove Theorem 10. We first restate several useful definitions and facts in Appendix D.1, then we propose our algorithm for unconstrained case in Appendix D.2. Finally we analyze the algorithm in Appendix D.3 and extend the result to cube constrained case in Appendix D.4

Theorem 23 *For $\varepsilon > 0$, $d, k = \Theta(1)$, there is a deterministic algorithm finding ε-stationary points, running within k-round, with*

- $C(d, k, L, \Delta) \cdot \varepsilon^{-\frac{d-1}{\left(\frac{2d}{d+1}\right)^k - 1} - d - 1}$ *queries per iteration when f is unconstrained,*
- $C(d, k, L) \cdot \varepsilon^{-\frac{d-1}{2\left(\left(\frac{2d}{d+1}\right)^k - 1\right)} - \frac{d-1}{2}}$ *queries per iteration when f is constrained on $[0, 1]^d$.*

D.1. Useful tools

Following Bubeck and Mikulincer (2020); Hollender and Zampetakis (2023), we begin with some definitions and technical lemmas.

Definition 24 (ε-unreachable) *For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ and $\varepsilon > 0$ we say that $\boldsymbol{y}$ is ε-unreachable from $\boldsymbol{x}$ if the following holds:*

$$f(\boldsymbol{y}) > f(\boldsymbol{x}) - \varepsilon \left\|\boldsymbol{x} - \boldsymbol{y}\right\|_2 .$$

Definition 25 (Full-dimensional hyperrectangle) *A k-dimensional hyperrectangle R in $\mathbb{R}^d$ is a set of the form $R = [a_1, b_1] \times \cdots \times [a_d, b_d]$, where $a_i \le b_i$ for all $i \in [d]$, and $|\{i \in [d] : a_i < b_i\}| = k$. When $k = d$, we also say that R is full-dimensional.*

Lemma 26 (Lemma 3 in Hollender and Zampetakis (2023)) *Let R be a full-dimensional hyperrectangle in $\mathbb{R}^d$ and $\boldsymbol{x}$ a point in R. For any $\varepsilon > 0$, if all $\boldsymbol{y} \in \partial R$ are ε-unreachable from $\boldsymbol{x}$, then R contains an ε-stationary point of f.*

Corollary 27 (Corollary 2 in Hollender and Zampetakis (2023)) *Let $R = [a_1, b_1] \times \cdots \times [a_d, b_d]$ be a full-dimensional hyperrectangle in $\mathbb{R}^d$ and $\boldsymbol{x}$ a point in R. For any $\varepsilon > 0$, if all $\boldsymbol{y} \in \partial R$ are $(\varepsilon/2)$-unreachable from $\boldsymbol{x}$, and $\max_i (b_i - a_i) \le \frac{\varepsilon}{2\sqrt{d}L}$, then $\boldsymbol{x}$ is an ε-stationary point of f.*

Definition 28 (Nice δ-net) *Let $\delta > 0$ and let R be a k-dimensional hyperrectangle in $\mathbb{R}^d$. A set of points $S \subseteq R$ is a nice δ-net of R if for any face F of R it holds that $S \cap F$ is a δ-net of F.*

The construction of nice δ-net with reasonable size can be summarized in the following lemma.

Lemma 29 (Lemma 9 in Hollender and Zampetakis (2023)) *Let $R = [a_1, b_1] \times \cdots \times [a_d, b_d]$ be a k-dimensional hyperrectangle in $\mathbb{R}^d$. Then, for any $\delta > 0$, there exists a nice δ-net S of R with $|S| = \prod_{i=1}^{d} (\lceil \sqrt{k}(b_i - a_i)/2\delta \rceil + 1)$. In particular, if R is $(d-1)$-dimensional, then we have $|S| \le (\sqrt{d}r/2\delta)^{d-1}$ for any r satisfying $r \ge \max_i (b_i - a_i)$ and $r \ge 8\sqrt{d}\delta$.*

The following lemma shows the unreachablility of δ-net of E implies unreachablility of S.

Lemma 30 (Lemma 10 in Hollender and Zampetakis (2023)) *Let E be a $(d-1)$-dimensional hyperrectangle of $\mathbb{R}^d$ and let S be a nice δ-net of E. Let $\varepsilon > 0$ and let $\boldsymbol{x} \in \mathbb{R}^d$ with $\operatorname{dist}(\boldsymbol{x}, E) > 0$. Then, if all $\boldsymbol{z} \in S$ are ε-unreachable from $\boldsymbol{x}$, it follows that all $\boldsymbol{y} \in E$ are ε'-unreachable from $\boldsymbol{x}$, where*

$$\varepsilon' = \varepsilon + \frac{\delta^2}{2\operatorname{dist}(\boldsymbol{x}, E)} \left(L + \frac{2\varepsilon}{\operatorname{dist}(\boldsymbol{x}, E)} \right).$$

D.2. Gradient flow fully parallel trapping

Our constant iteration algorithm for finding stationary points is summarized in Algorithm 1. In Line 2, the algorithm initializes the region with a unreachable boundary. In Lines 3–10, the gradient is trapped using multiple $(d-1)$-dimensional grids, followed by an update to the current solution. In Lines 11–18, the hyperrectangle is updated by reducing the length of its longest side to $1/\ell_t$. In Line 20, the accuracy constant is slightly increased to ensure that the boundary of the updated hyperrectangle remains unreachable to the current solution.

Algorithm 1: The Gradient Flow Grid Trapping (GFGT) Algorithm

```
Input: accuracy $\varepsilon > 0$, smooth parameter $L > 0$, dimension $d \geq 2$, initialization point
       $\boldsymbol{x}^0 \in \mathbb{R}^d$, query access to $f : \mathbb{R}^d \to [0, +\infty)$ with $L$-Lipschitz $\nabla f$.
1  Set $t = 0$ and $\varepsilon_0 = \varepsilon/4$,
2  Initialize hyperrectangle $R_0 := [a_1^1, b_1^1] \times \cdots \times [a_d^1, b_d^1]$ as
       $$R_0 = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x} - \boldsymbol{x}^0\|_\infty \leq 2f(\boldsymbol{x}^0)/\varepsilon_0\}.$$
   for $t + 1 \in [k]$ do
3      Set $r_j^t = b_j^t - a_j^t$ for $j \in [d]$ and $r^t = \min_j r_j^t$,
4      Set the number of trap barriers on each coordinate direction $\ell_t = \left(\frac{4f(\boldsymbol{x}^0)\sqrt{d}L}{3^k \cdot \varepsilon_0 \cdot \varepsilon}\right)^{\frac{\frac{d-1}{d+1}\left(\frac{2d}{d+1}\right)^t}{\left(\frac{2d}{d+1}\right)^k - 1}}$,
5      Set the gap size of trap barrier as $\delta_t = \sqrt{\frac{\varepsilon r^t (3/4)^{\lfloor td \rfloor}}{40 \cdot 3^k \cdot \ell_t d \sqrt{d} L}}$,
6      Query $f$ on the nice $\delta_t$-net $S(E)$ of any $E \in \mathcal{E}^t$ where
           $$\mathcal{E}^t = \{[a_1^t, b_1^t] \times \cdots \times [a_j^t + m r_j^t/\ell_t] \times \cdots \times [a_d^t, b_d^t] : j \in [d], m \in [\ell_t - 1]\},$$
         Let $S^\star = \left\{\boldsymbol{z} \in \bigcup_{E \in \mathcal{E}^t} S(E) : f(\boldsymbol{z}) \leq f(\boldsymbol{x}^t) - \varepsilon_t \|\boldsymbol{x}^t - \boldsymbol{z}\|_2\right\}$,
7      if $S^\star = \emptyset$ then
8          $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$,
9      else
10         $\boldsymbol{x}^{t+1} = \arg\min_{\boldsymbol{z} \in S^\star} f(\boldsymbol{z})$,
11     for $j \in [d]$ do
12         if $a_j^t \leq [\boldsymbol{x}^{t+1}]_j \leq a_j^t + r_j^t/\ell_t$ then
13             $a_j^{t+1} = a_j^t$ and $b_j^{t+1} = a_j^t + 2r_j^t/\ell_t$,
14         else if $b_j^t - r_j^t/\ell_t \leq \boldsymbol{x}_j^{t+1} \leq b_j^t$ then
15             $a_j^{t+1} = b_j^t - 2r_j^t/\ell_t$ and $b_j^{t+1} = b_j^t$,
16         else
17             Assume $a_j^t + m_1 r_j^t/\ell_t \leq \boldsymbol{x}_j^{t+1} < a_j^t + m_2 r_j^t/\ell_t$ with $m_1 \geq 1$ and $m_2 \leq \ell_t - 1$,
18             update $a_j^{t+1} = a_j^t + (m_1 - 1)r_j^t/\ell_t$ and $b_j^{t+1} = a_j^t + (m_2 + 1)r_j^t/\ell_t$,
19     end
20     Update $\varepsilon_{t+1} = \varepsilon_t + \frac{\varepsilon(3/4)^{\lfloor td \rfloor}}{16d}$.
21 end
22 return $\boldsymbol{x}_k$.
```

D.3. Analysis of Algorithm 1: Proof of the first part of Theorem 10

Lemma 31 (Complexity) *For any $d \geq 2$, the number of queries to f in each iteration is bound by*

$$C(d, k, L) f(\boldsymbol{x}^0)^{(d-1)/2} \varepsilon^{-\frac{\frac{d-1}{2}}{\left(\frac{2d}{d+1}\right)^k - 1} - d - 1},$$

where $C(d, k, L)$ is a constant depends on d, k, L.

Proof In iteration t, there is $d \cdot \ell_t$ hyperrectangles with dimension $d-1$ and side length bounded by

$$\left(\prod_{i=0}^{t-1} \frac{3}{\ell_t}\right) \cdot \frac{2f(\boldsymbol{x}^0)}{\varepsilon}.$$

Combining Lemma 29, we have

$$\begin{aligned}
\#\text{queries in iteration } t &\leq \left(\sqrt{d} \cdot 3^t r^t \cdot \frac{1}{2\sqrt{\frac{\varepsilon r^t (3/4)^{\lfloor td \rfloor}}{40 \cdot 3^k \cdot \ell_t d \sqrt{d} L}}}\right)^{d-1} \cdot d \cdot \ell_t \\
&\leq d\left(\sqrt{d} \cdot 3^t \cdot \frac{\sqrt{40 \cdot 3^k \cdot d\sqrt{d}L}}{2\sqrt{(3/4)^{\lfloor td \rfloor}}}\right)^{d-1} \cdot \ell_t^{(d+1)/2} \cdot (r^t)^{(d-1)/2} \cdot \varepsilon^{-(d-1)/2} \\
&:= C(d,k,L) \cdot \ell_t^{(d+1)/2} \cdot (r^t)^{(d-1)/2} \cdot \varepsilon^{-(d-1)/2} \\
&\leq C(d,k,L) \cdot \ell_t^{(d+1)/2} \cdot \left(\left(\prod_{i=0}^{t-1} \frac{3}{\ell_t}\right) \cdot \frac{2f(\boldsymbol{x}^0)}{\varepsilon_0}\right)^{(d-1)/2} \cdot \varepsilon^{-(d-1)/2} \\
&\leq C(d,k,L) \cdot \left(\frac{2f(\boldsymbol{x}^0)}{\varepsilon_0}\right)^{(d-1)/2} \cdot \varepsilon^{-\frac{d-1}{\left(\frac{2d}{d+1}\right)^k - 1} - \frac{d-1}{2}}
\end{aligned}$$

■

Lemma 32 (Correctness) *The output of the algorithm $\boldsymbol{x}^k$ is an ε-stationary point of f.*

Proof We assume $f(\boldsymbol{x}^0) > 0$ otherwise, $\boldsymbol{x}^0$ is a global minimum and thus a stationary point.

Let R_t denote the hyperrectangle at the beginning of iteration $t = 0, \ldots, k-1$. We consider the following invariant at every iteration: for all $\boldsymbol{y} \in \partial R_t$ are ε_t-unreachable from $\boldsymbol{x}^t$.

If such invariant holds, we observe that

$$\max_i (b_i^k - a_i^k) \leq \left(\prod_{i=0}^{k-1} \frac{3}{\ell_t}\right) \cdot \frac{2f(\boldsymbol{x}^0)}{\varepsilon_0} \leq \frac{\varepsilon}{2\sqrt{d}L},$$

and ε_k can be bounded as

$$\varepsilon_k = \varepsilon_0 + \sum_{t=0}^{T-1} \frac{\varepsilon (3/4)^{\lfloor td \rfloor}}{16d} \leq \varepsilon_0 + \sum_{t=0}^{\infty} \frac{\varepsilon (3/4)^{\lfloor td \rfloor}}{16d} \leq \frac{\varepsilon}{2}.$$

By Corollary 27, $\boldsymbol{x}_k$ is ε-stationary point of f.

Now we prove the invariant by induction. For $t = 0$, for any $\boldsymbol{y} \in R_0$, by definition of R_0 we have $\left\|\boldsymbol{x}^0 - \boldsymbol{y}\right\|_2 \geq \left\|\boldsymbol{x}^0 - \boldsymbol{y}\right\|_\infty = 2f(\boldsymbol{x}^0)/\varepsilon_0$ thus

$$f(\boldsymbol{y}) \geq 0 \geq 2f(\boldsymbol{x}^0) - \varepsilon_0 \left\|\boldsymbol{x}^0 - \boldsymbol{y}\right\|_2 > f(\boldsymbol{x}^0) - \varepsilon_0 \left\|\boldsymbol{x}^0 - \boldsymbol{y}\right\|_2.$$

Now we show the invariant holds for iteration $t+1$ conditioned on that it holds for iteration t.

Case 1: $S^\star = \emptyset$. In this case $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t$. For $\boldsymbol{y} \in \partial R_{t+1} \cap \partial R_t$ are ε_t-unreachable from $\boldsymbol{x}^t$. Thus by $\varepsilon_{t+1} \geq \varepsilon_t$, it is also ε_{t+1}-unreachable from $\boldsymbol{x}^{t+1}$. For $\boldsymbol{y} \in \partial R_{t+1} \setminus \partial R_t \subseteq \bigcup_{E \in \mathcal{E}^t \cap \partial R_{t+1}} S(E)$, we show as follows. By construction in Lines 12–19, we have

$$\mathrm{dist}(\boldsymbol{x}^t, E) \geq \min_j r_j^t / \ell_t \geq \left(\prod_{i=0}^{k-1} \frac{1}{\ell_t} \right) \cdot \frac{2f(\boldsymbol{x}^0)}{\varepsilon_0} = \frac{1}{3^k} \frac{\varepsilon}{2\sqrt{d}L}$$

Combining $S^\star = \emptyset$ and Lemma 30, we can claim that for all $\boldsymbol{y} \in \bigcup_{E \in \mathcal{E}^t \cap \partial R_{t+1}} S(E)$ are ε'-unreachable from $\boldsymbol{x}^{t+1}$, where

$$\begin{aligned}
\varepsilon' &= \varepsilon_t + \frac{\delta^2}{2\mathrm{dist}(\boldsymbol{x}, E)} \left(L + \frac{2\varepsilon}{\mathrm{dist}(\boldsymbol{x}, E)} \right) \\
&\leq \varepsilon_t + \frac{\delta^2}{2\mathrm{dist}(\boldsymbol{x}, E)} \left(L + 3^k 4\sqrt{d}L \right) \\
&\leq \varepsilon_t + \frac{\delta^2}{2\mathrm{dist}(\boldsymbol{x}, E)} 3^k 5\sqrt{d}L \\
&\leq \varepsilon_t + \frac{\delta^2 \ell_t}{2r^t} 3^k 5\sqrt{d}L \\
&\leq \varepsilon_t + \frac{\varepsilon(3/4)^{\lfloor td \rfloor}}{16d} = \varepsilon_{t+1}.
\end{aligned}$$

Case 2: $S^\star \neq \emptyset$. In this case, $\boldsymbol{x}^{t+1} = \arg\min_{\boldsymbol{z} \in S^\star} f(\boldsymbol{z})$. For $\boldsymbol{y} \in \partial R_{t+1} \cap \partial R_t$, we have

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}^t) - \varepsilon_t \left\| \boldsymbol{x}^t - \boldsymbol{y} \right\|_2 \geq f(\boldsymbol{x}^{t+1}) + \varepsilon_t \left\| \boldsymbol{x}^t - \boldsymbol{x}^{t+1} \right\|_2 - \varepsilon_t \left\| \boldsymbol{x}^t - \boldsymbol{y} \right\|_2 \geq f(\boldsymbol{x}^{t+1}) - \varepsilon_t \left\| \boldsymbol{x}^{t+1} - \boldsymbol{y} \right\|_2,$$

i.e. , $\boldsymbol{y}$ is ε_t-unreachable from $\boldsymbol{x}^{t+1}$. Since $\varepsilon_{t+1} \geq \varepsilon_t$, we have $\boldsymbol{y}$ is ε_{t+1}-unreachable from $\boldsymbol{x}^{t+1}$. For $\boldsymbol{y} \in \partial R_{t+1} \setminus \partial R_t$, we first show for every $\boldsymbol{y} \in S(E)$ where $E \in \mathcal{E}^t \cap \partial R_{t+1}$, it is ε_t-unreachable for $\boldsymbol{x}^{t+1}$. We note $\partial R_{t+1} \setminus \partial R_t \subseteq \mathcal{E}^t$. On the one hand, for $\boldsymbol{y} \in S(E) \setminus S^\star$, it is ε_t-unreachable from $\boldsymbol{x}^t$. Thus by same argument before, it is ε_t-unreachable from $\boldsymbol{x}^{t+1}$. On the other hand, for $\boldsymbol{y} \in S(E) \cap S^\star$, since $\boldsymbol{x}^{t+1} = \arg\min_{\boldsymbol{z} \in S^\star} f(\boldsymbol{z})$ and $\left\| \boldsymbol{x}^{t+1} - \boldsymbol{y} \right\|_2 > 0$, we have

$$f(\boldsymbol{y}) > f(\boldsymbol{x}^{t+1}) - \varepsilon_t \left\| \boldsymbol{x}^{t+1} - \boldsymbol{y} \right\|_2,$$

i.e. , $\boldsymbol{y}$ is ε_t-unreachable from $\boldsymbol{x}^{t+1}$. Finally, we use Lemma 30 as before and claim that for all $\boldsymbol{y} \in \bigcup_{E \in \mathcal{E}^t \cap \partial R_{t+1}} S(E)$ are ε_{t+1}-unreachable from $\boldsymbol{x}^{t+1}$. ■

D.4. Constrained setting: proof of second part of Theorem 10

In this section, we adapt Algorithm 1 to the problem of finding stationary points constrained on cube $[0, 1]^d$. The goal is to find an ε-stationary point, also known as an ε-KKT point, *i.e.*, a point $\boldsymbol{x} \in [0, 1]^d$ such that $\|g(\boldsymbol{x})\|_2 \leq \varepsilon$, where g is the projected gradient of f on $[0, 1]^d$ defined as

$$g_i(\boldsymbol{x}) = \begin{cases} \min\{0, [\nabla f(\boldsymbol{x})]_i\} & \text{if } x_i = 0 \\ [\nabla f(\boldsymbol{x})]_i & \text{if } x_i \in (0, 1) \\ \max\{0, [\nabla f(\boldsymbol{x})]_i\} & \text{if } x_i = 1. \end{cases}$$

We modify the algorithm as follows. We also note similar argument was proposed in (Hollender and Zampetakis, 2023, Section 4.3).

- **Initialization:** We initialize $\boldsymbol{x}^0$ to be an arbitrary point in $[0,1]^d$ instead of
$$R_0 = \{\boldsymbol{x} \in \mathbb{R}^d : \left\|\boldsymbol{x} - \boldsymbol{x}^0\right\|_\infty \le 2f(\boldsymbol{x}^0)/\varepsilon_0\}.$$
- **The number of barriers.** We set $r^0 = 1$ instead of $r^0 = 2f(\boldsymbol{x}^0)/\varepsilon_0$, and
$$\ell_t = \left(\frac{2\sqrt{d}L}{3^k \cdot \varepsilon}\right)^{\frac{\frac{d-1}{d+1}\left(\frac{2d}{d+1}\right)^t}{\left(\frac{2d}{d+1}\right)^k - 1}}.$$
With this setting, we have
$$\max_i (b_i^k - a_i^k) \le \left(\prod_{i=0}^{k-1} \frac{3}{\ell_t}\right) \le \frac{\varepsilon}{2\sqrt{d}L},$$
- **Extraction of a solution:** Instead of outputting $\boldsymbol{x}_d$, we check the $2^d = \Theta(1)$ corners of R_k until we find one with $\|g(\boldsymbol{y})\|_2 \le \varepsilon$. We can use $O(d)$ queries to f to compute a sufficiently good approximation of ∇f and thus g at any point.

The rest of the algorithm is unchanged. The analysis is also almost same and we highlight the main difference as follows.

- **Correctness:** The invariant is modified as: for all $\boldsymbol{y} \in \partial R_t \setminus \partial[0,1]^d$ are ε_t-unreachable from $\boldsymbol{x}^t$. The proof of this new invariant is identical to the old one.

  This new invariant, combined with a modified version of the proof of Lemma 26 (where we analyze the piecewise differentiable *projected* gradient flow defined by $\gamma'(t) = -g(\gamma(t))$), establishes the following: there exists a point $\boldsymbol{x} \in R_T$ such that $\|g(\boldsymbol{x})\|_2 \le \varepsilon/2$. A straightforward argument then shows that any corner $\boldsymbol{y}$ of the face of R_k containing $\boldsymbol{x}$ must satisfy $\|g(\boldsymbol{y})\|_2 \le \varepsilon$. Hence, the algorithm successfully returns an ε-stationary point.

- **Running time:** The number of queries used by algorithm is identical to the unconstrained version, namely,
$$\begin{aligned}
\#\text{queries in iteration } t &\le \left(\sqrt{d}\cdot 3^t r^t \cdot \frac{1}{2\sqrt{\frac{\varepsilon r^t (3/4)^{\lfloor td \rfloor}}{40\cdot 3^k \cdot \ell_t d\sqrt{d}L}}}\right)^{d-1} \cdot d \cdot \ell_t \\
&\le d\left(\sqrt{d}\cdot 3^t \cdot \frac{\sqrt{40\cdot 3^k \cdot d\sqrt{d}L}}{2\sqrt{(3/4)^{\lfloor td \rfloor}}}\right)^{d-1} \cdot \ell_t^{(d+1)/2} \cdot \left(r^t\right)^{(d-1)/2} \cdot \varepsilon^{-(d-1)/2} \\
&:= C(d,k,L) \cdot \ell_t^{(d+1)/2} \cdot \left(r^t\right)^{(d-1)/2} \cdot \varepsilon^{-(d-1)/2} \\
&\le C(d,k,L) \cdot \ell_t^{(d+1)/2} \cdot \left(\left(\prod_{i=0}^{t-1} \frac{3}{\ell_t}\right)\right)^{(d-1)/2} \cdot \varepsilon^{-(d-1)/2} \\
&\le C(d,k,L) \cdot \varepsilon^{-\frac{d-1}{\left(\frac{2d}{d+1}\right)^k - 1} - \frac{d-1}{2}}.
\end{aligned}$$