UNIVERSITY OF CALIFORNIA
Los Angeles

Enhancing Local Derivative-Free Optimization with Curvature Information and Inspection Strategies

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mathematics

by

Bumsu Kim

2023

© Copyright by
Bumsu Kim
2023

ABSTRACT OF THE DISSERTATION

Enhancing Local Derivative-Free Optimization with
Curvature Information and Inspection Strategies

by

Bumsu Kim
Doctor of Philosophy in Mathematics
University of California, Los Angeles, 2023
Professor Stanley J. Osher, Chair

(Abstract omitted for brevity)

The dissertation of Bumsu Kim is approved.

Quanquan Gu

Guido F. Montufar

Deanna Needell

Wotao Yin

Stanley J. Osher, Committee Chair

University of California, Los Angeles

2023

To my parents,

for whom my respect and love remain throughout my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

(Acknowledgments omitted for brevity.)

VITA

2015 B.S. in Mathematics, Korea Advanced Institute of Science and Technology.

2017 M.S. in Mathematics, Korea Advanced Institute of Science and Technology.

2017–Present Teaching and Research Assistant, Department of Mathematics, UCLA.

PUBLICATIONS

Bumsu Kim, HanQin Cai, Daniel McKenzie, and Wotao Yin. Curvature-Aware Derivative-Free Optimization. *arXiv preprint arXiv:2109.13391 (2021)*.

Bumsu Kim, and Wotao Yin. Bridging the Gap Between Local and Global Derivative-Free Optimization. *In Preparation*.

CHAPTER 1

Introduction

1.1 Introduction

Derivative-Free Optimization

CHAPTER 2

Curvature-Aware Derivative-Free Optimization

2.1 Introduction

We consider minimizing a function $f : \mathbb{R}^d \to \mathbb{R}$, with only access to function evaluations $f(x)$, and no access to gradients or directional derivatives. This setting is commonly referred to as derivative-free optimization (DFO). DFO has a rich history and has recently gained popularity in various areas such as reinforcement learning [SHC17, MGR18, CPP20], hyperparameter tuning [BB12], and adversarial attacks on neural network classifiers [CZS17, CMY22b]. In all of these applications, evaluating $f(x)$ is either expensive, time-consuming, or inconvenient, and therefore, it is desirable for DFO algorithms to minimize the number of function evaluations required.

Classical methods for DFO include the Nelder-Mead simplex method [NM65], direct search methods [KLT03], and model-based methods [CSV09]. However, these methods tend to scale poorly with the problem dimension d, although recent works [CR22, CMO22, CO22] have made progress in this direction. Due to the demands of large-scale machine learning applications, *zeroth-order* (ZO) methods for DFO have gained increasing attention [LCK20]. ZO methods mimic first-order methods like gradient descent but approximate all derivative information using function queries. At each iteration, the algorithm selects a direction u_k and takes a step $x_{k+1} = x_k + \alpha_k u_k$. While the selection of u_k has been well studied (see [BCC21] and references therein), this chapter focuses on the selection of α_k, allowing for u_k to be either randomly selected or an approximation to the negative gradient (i.e., $u_k \approx -\nabla f(x_k)$).

Intelligently choosing α_k can lead to convergence in fewer iterations, but this gain may be offset by the number of queries it takes. If we compute $u_k \approx -\nabla f(x_k)$, techniques such as backtracking line search from first-order optimization can be employed [BCS21]. However, obtaining a sufficiently accurate approximation to $-\nabla f(x_k)$ requires $\Omega(d)$ queries per iteration [BCC21], which is impractical for large d. On the other hand, when we take u_k as a random vector, with high probability u_k is almost orthogonal to $-\nabla f(x_k)$. Hence, α_k in [GL13, NS17, BGR20] is very small to guarantee descent at every iteration (possibly in expectation). Our approach differs from these methods.

We propose using finite difference approximations to the first and second derivatives of the univariate function $\alpha \mapsto f(x_k + \alpha u_k)$ to compute a candidate α_+ for α_k. Specifically, we set

$$\alpha_+ = \frac{d_r}{\hat{L} h_r},$$

where

$$d_r := \frac{f(x_k + r u_k) - f(x_k - r u_k)}{2r}, \tag{2.1}$$

$$h_r := \frac{f(x_k + r u_k) - 2f(x_k) + f(x_k - r u_k)}{r^2}, \tag{2.2}$$

and $\hat{L}$ is a user-specified parameter. Computing α_+ requires only three queries per iteration. This simple modification to the well-known Random Search algorithm [GL13, NS17] (which takes $\alpha_k = d_r/\hat{L}$ or similar) can be viewed as an inexact one-dimensional Newton's method at each iteration. When encountering low curvature directions, h_r is small and α_+ is large, so this α_+ may occasionally fail to guarantee descent. To remedy this, we combine our step-size rule with a simple safeguarding scheme based on the recently introduced Stochastic Three Point method [BGR20], which guarantees $f(x_{k+1}) \leq f(x_k)$ at every iterate. Importantly, we show that α_+ is a good candidate, i.e., $f(x_{k+1})$ is significantly smaller than $f(x_k)$ a positive proportion of the time. From this, we can quantify the expected total number

of function queries required to reach a target solution accuracy. Because our method is a natural extension of Random Search that incorporates second derivative information, we dub it Curvature-Aware Random Search, or CARS.

In addition to CARS, we propose an extension called CARS-CR (CARS with Cubic Regularization) and CARS-NQ (CARS with Numerical Quadrature). CARS-CR modifies the stochastic subspace cubic Newton method [HDN20] into a zeroth-order method, and it is essentially CARS with an adaptive parameter $\hat{L}$ and achieves $\mathcal{O}(k^{-1})$ convergence for convex functions. CARS-NQ, on the other hand, incorporates Gauss-Hermite quadrature to estimate the derivatives of the smoothed function. It allows larger sampling radius and is particularly effective for non-convex functions of the form $f = f_{\text{cvx}} + f_{\text{osc}}$ where f_{cvx} is strongly convex and f_{osc} is rapidly oscillating.

Our numerical experiments show that both CARS and its variants outperform state-of-the-art algorithms on benchmarks across various problem dimensions, demonstrating efficiency and robustness. Furthermore, our results on adversarial attacks show that CARS can be adapted to different sample distributions of u_k. We demonstrate that CARS performs well with a tailored distribution for a particular problem, an adversarial attack on a pre-trained neural network.

Organization. This chapter is laid out as follows. In the rest of this section, we fix the notation and discuss prior art. In Section 2.2, we introduce the main algorithm, namely Curvature-Aware Random Search (CARS), along with its convergence analysis. Section 2.3 extends CARS with Cubic Regularization (CARS-CR) for general convex functions. In Section 2.6, we provide mathematical proofs to support our technical claims. Section 2.7 contains extensive numerical experiments that empirically verify our technical claims. Section 2.8 concludes the chapter.

2.1.1 Assumptions and Notation

In developing and analyzing CARS, we assume that f is a convex and twice continuously differentiable function. We use $g(x) = \nabla f(x)$ and $H(x) = \nabla^2 f(x)$ briefly in the theoretical analysis of Section 2.2.1. For a fixed initial point x_0, we define the level-set $\mathcal{Q} = \{x \in \mathbb{R}^d : f(x) \leq f(x_0)\}$, $\|\cdot\|$ as the Euclidean norm, and $f_\star := \min_{x\in\mathbb{R}^d} f(x)$. We say x_k is an ε-optimal solution if $f(x_k) - f_\star \leq \varepsilon$. We use $\mathcal{D}$ to denote a probability distribution on $\mathbb{R}^d$. For any measurable set $S \subseteq \mathbb{R}^d$ with finite measure, $\text{Unif}(S)$ denotes the uniform distribution over S. The unit sphere is written as $\mathbb{S}^{d-1} := \{u : \|u\| = 1\} \subseteq \mathbb{R}^d$, and $e_1, \cdots, e_d$ represent the canonical basis vectors in $\mathbb{R}^d$. For two matrices A and B, we write $A \preceq B$ if $B - A$ is positive semi-definite.

Definition 1. We say f is L-smooth, $L > 0$, if $H(x) \preceq LI_d$ for all $x \in \mathcal{Q}$.

Definition 2. We say f is μ-strongly convex, $\mu > 0$, if $\mu I_d \preceq H(x)$ for all $x \in \mathcal{Q}$.

Under strong convexity, $H(z)$ is positive definite for all $z \in \mathcal{Q}$; hence the following inner product and induced norm are well-defined for all $z \in \mathcal{Q}$:

$$\langle x, y\rangle_{H(z)} := \langle H(z)x, y\rangle \qquad \text{and} \qquad \|x\|^2_{H(z)} := \langle x, x\rangle_{H(z)}.$$

Strong convexity also implies the following [GKL19, Proposition 2].

Lemma 2.1.1 ($\hat{L}$-Relative Smoothness and $\hat{\mu}$-Relative Convexity)**.** *If f is μ-strongly convex, then f is $\hat{\mu}$-relatively convex and $\hat{L}$-relatively smooth for some $\hat{L} \geq \hat{\mu} > 0$, i.e. for all $x, y \in \mathcal{Q}$*

$$\frac{\hat{\mu}}{2}\|x - y\|^2_{H(y)} \leq f(x) - f(y) - \langle g(y), x - y\rangle \leq \frac{\hat{L}}{2}\|x - y\|^2_{H(y)}.$$

We also make the following regularity assumption on H.

Assumption 1. H is a-Hölder continuous for some $a > 0$, *i.e.*

$$|u^\top (H(x) - H(y))\, u| \leq L_a \|x - y\|^a \tag{2.3}$$

for any unit vector $u \in \mathbb{S}^{d-1}$ and $x, y \in \mathcal{Q}$.

Hölder continuity reduces to Lipschitz continuity when $a = 1$. Assumption 1 can be used to refine the relative smoothness and relative convexity constants in a smaller region.

2.1.2 Prior Art

For a comprehensive introduction to DFO we refer the reader to [CSV09] or the more recent survey article [LMW19]. As mentioned above, our interest is in ZO approaches to DFO [LCK20], as these have low per-iteration query complexity (with respect to the dimension of the problem) and have been successfully used in modern machine learning applications, such as adversarial attacks on neural networks [CZS17, LKC18, CSC19, CMY22b, CLM21] and reinforcement learning [SHC17, CRS18, FGK18]. Of particular relevance to this work is ZO algorithms based on *line search*:

$$\begin{aligned}
&\text{Sample } u_k \text{ from } \mathcal{D}, \\
&\alpha_k \approx \alpha_\star = \arg\min_{\alpha \in \mathbb{R}} f(x_k + \alpha u_k), && \text{(2.4a)} \\
&x_{k+1} = x_k + \alpha_k u_k, && \text{(2.4b)}
\end{aligned}$$

which may be thought of as zeroth-order analogues of coordinate descent [Nes12]. All of the complexity results discussed below assume *noise-free* access to $f(x)$. The noisy case is more complicated, see [JNR12]. The first papers to use this scheme were [Kar74, Kar75], where convergence is discussed under the assumptions that u_k is a descent direction[1] for

[1] $u_k^\top \nabla f(x_k) < 0$.

all k and (2.4a) is solved sufficiently accurately. Assuming (2.4a) is solved *exactly*, [MR64] proves this scheme finds an ε-optimal solution in $\mathcal{O}(\log(1/\varepsilon))$ iterations when f is a quadratic function (see also [SC76] for a discussion of these results in English). In [Kru83], $\mathcal{O}(\log(1/\varepsilon))$ iteration complexity was proved assuming access to an approximate line search oracle that solves (2.4a) sufficiently accurately, for any strongly convex f, as long as u_k are cyclically sampled coordinate vectors. Similar ideas can be found in [GLL88, GS07, GR15]. More recently, [SMG13] studied (2.4) under the name *Random Pursuit* which assumes access to an approximate line search oracle satisfying either additive ($\alpha_\star - \delta \leq \tilde{\alpha} \leq \alpha_\star + \delta$) or multiplicative ($(1-\delta)\alpha_\star \leq \tilde{\alpha} \leq \alpha_\star$ and $\operatorname{sign}(\tilde{\alpha}) = \operatorname{sign}(\alpha_\star)$) error bounds. They show Random Pursuit finds an ε-optimal solution in $\mathcal{O}(\log(1/\varepsilon))$ (resp. $\mathcal{O}(1/\varepsilon)$) iterations if f is strongly convex (resp. convex). The use of $\mathcal{O}(\cdot)$ above suppresses the dependence of the query complexity on the dimension d. In all results stated, the query complexity scales at least linearly with d. This is unavoidable in DFO for generic f; see [WDB18, BG21, CMY22b, CMY22a, CLM21, CR21] for recent progress in overcoming this.

We highlight a shortcoming of the aforementioned works: Although they provide essentially optimal bounds on the *iteration* complexity, they do not bound the *query* complexity. Indeed, the true query complexity will depend on the inner workings of the solver employed to solve (2.4a). For example, [SMG13] reports each call to the line search oracle requires an average of 4 function queries when $d \leq 128$ which increases to 7 when $d = 1024$. In contrast, CARS requires *only three queries* per iteration, independent of d. The recently introduced Stochastic Three Point (STP) method [BGR20, BBS20] also uses only three queries per iteration. However, STP is not scale invariant, and in practice we find its performance compares poorly against CARS (see Section 2.7).

We are partially motivated by ZOO-Newton [CZS17], which is essentially CARS with $\mathcal{D} = \operatorname{Unif}(\{e_1, \cdots, e_d\})$. In [CZS17], it is demonstrated empirically that ZOO-Newton performs well but no theoretical guarantees are provided. Our convergence guarantees for CARS imply convergence of ZOO-Newton as a special case. Many other works consider

Algorithm	Strg. Convex	Convex	Queries/Iter	Line Search Oracle
[Kar75]	$\mathcal{O}(\log(1/\varepsilon))$	$\mathcal{O}(1/\varepsilon)$	—	Yes
[Kru83]	$\mathcal{O}(\log(1/\varepsilon))$	—	—	Yes
NDFLS [GR15]	—	—	$< \infty$	No
Random Pursuit [SMG13]	$\mathcal{O}(\log(1/\varepsilon))$	$\mathcal{O}(1/\varepsilon)$	4–7 (empirical)	Yes
ZOO-Newton [CZS17]	—	—	3	No
Stochastic 3 Points [BGR20]	$\mathcal{O}(\log(1/\varepsilon))$	$\mathcal{O}(1/\varepsilon)$	3	No
CARS (proposed)	$\mathcal{O}(\log(1/\varepsilon))$	$\mathcal{O}(1/\varepsilon)^\dagger$	3 or $4^\dagger$	No

Table 2.1: Comparison of various line-search based ZO algorithms, all of which use random search directions. We refer to algorithms without an agreed-upon name by the paper in which it first appeared. If a quantity (*e.g.* queries per iteration, convergence rate) is not explicitly computed we denote this with "—". Notes: †: refers to CARS-CR variant.

adapting Newton's method to the derivative-free setting. However, obtaining an estimate of the $d \times d$ Hessian $\nabla^2 f(x_k)$ for general (*i.e.* unstructured) $f(x)$ is difficult. Thus, one needs to either use $\Omega(d^2)$ queries [Fab71] in order to obtain an accurate estimate of $\nabla^2 f(x_k)$—far too many for most applications—or use a high-variance approximation to $\nabla^2 f(x_k)$ [Spa00, YHF18, GK20, ZWS19, Zhu20]. CARS sidesteps this dichotomy, as it applies Newton's method to a one-dimensional function. Thus the "Hessian" to be estimated is 1×1.

Connection to Evolution Strategies. Evolution strategies (ES) are a class of derivative free optimization strategies inspired by biological evolution. Surprisingly, recent works [SHC17] have shown that many ES algorithms are in fact equivalent to Random Search algorithm [NS17] with Monte Carlo estimation of smoothed gradient. In this section we show how CARS can also be viewed as an ES. First, recall that the defining feature of an ES algorithm is that it maintains and iteratively updates a *population distribution*, *i.e.* a distribution over the search space $\mathbb{R}^d$, p_θ. The goal of any ES is to find

$$\theta^\star = \arg\min\{F(\theta) = \mathbb{E}_{x\sim p_\theta}[f(x)]\}.$$

Now consider a Gaussian population distribution: $p_\theta = \mathcal{N}(\psi, r^2 A^2)$ where $\theta = (\psi, A)$.

and $r > 0$ is a small, fixed, constant. We assume A is symmetric and positive definite. $F(\theta)$ can be written as

$$F(\psi, A) = \mathbb{E}_{v\sim\mathcal{N}(0,I_d)}[f(\psi + rAu)] = \int f(\psi + rAu)\phi(u)\ du,$$

where $\phi(u) = (2\pi)^{-d/2}\exp(-\|u\|^2/2)$ denotes the density of the d-dimensional standard normal distribution. Thus, we can regard F as a smoothed version of f. Integration by parts reveals:

$$\nabla_\psi F(\psi, A) = r^{-1}\mathbb{E}\left[f(\psi + rAu)A^{-1}u\right]. \tag{2.5}$$

Noticing that $\mathbb{E}[f(\psi)A^{-1}u] = 0$ and $\mathbb{E}[f(\psi - rAu)A^{-1}u] = -\mathbb{E}[f(\psi + rAu)A^{-1}u]$, we can also write (2.5) as

$$\begin{aligned}\nabla_\psi F(\psi, A) &= \mathbb{E}\left[r^{-1}(f(\psi + rAu) - f(\psi))A^{-1}u\right] \\ &= \mathbb{E}\left[(2r)^{-1}(f(\psi + rAu) - f(\psi - rAu))A^{-1}u\right] \\ &= \mathbb{E}\left[d_r(\psi; Au)A^{-1}u\right]\end{aligned}$$

Fixing A to be a constant matrix, and using a single sample to estimate $\nabla_\psi F(\psi)$ we obtain the NRS gradient estimator [NS17]. Setting $A = I_d$ and using a population of size greater than one for the Monte Carlo approximation of the expectation, it becomes the evolution strategy introduced in [SHC17]. Allowing A to vary and using additional samples to approximate $A \approx H^{-1}$ yields the gradient estimator employed in [YHF18, SY20]. Note that they use the estimated Newton vector: $A^2\nabla_\psi F(\psi, A)$ as their descent direction.

We now show the connection between this evolution strategy and CARS. Again by integration by parts[2]:

[2]This can also by deduced from Stein's formula [Ste72, Ste81]

$$\nabla^2_\psi F(\psi, A) = r^{-2} A^{-1} \mathbb{E}[f(\psi + rAu)(uu^\top - I_d)] A^{-1}. \tag{2.6}$$

For simplicity define $M := r^{-2}\mathbb{E}[f(\psi + rAu)(uu^\top - I_d)]$. Then, fixing A, the Newton vector $\vec{n}(\psi)$ for F at ψ is

$$\vec{n}(\psi) = -(\nabla^2_\psi F)^{-1}(\nabla_\psi F) = AM^{-1}\mathbb{E}[d_r(\psi; Au)u]. \tag{2.7}$$

Note that $\mathbb{E}[uu^\top - I_d] = 0$, so we can subtract $2f(\psi)(uu^\top - I_d)$ from M inside the expectation. Using symmetry:

$$M = \mathbb{E}[h_r(\psi; Au)(uu^\top - I_d)/2].$$

Finally, if we use a single sample u for each expectation in (2.7),

$$\vec{n}(\psi) \approx -2\frac{d_r(\psi; Au)}{h_r(\psi; Au)} A(uu^\top - I_d)^{-1}u \overset{(a)}{=} -2\frac{d_r(\psi; Au)}{h_r(\psi; Au)} \frac{1}{\|u\|_2^2 - 1} Au,$$

where (a) follows as u is an eigenvector of $uu^\top - I_d$ with eigenvalue $\|u\|_2^2 - 1$. For $A = I_d$, we recover CARS, up to a constant factor coming from the difference between the Gaussian distribution and the uniform distribution on the unit sphere. Thus, CARS can be thought of as an ES maintaining an isotropic population distribution (as I_d is the covariance) but taking a *Newton step* each iteration to update the mean. This interpretation suggests the following modification to CARS: one can also update A using the stochastic gradient:

$$\nabla_A F(\psi, A) = A^{-1}\mathbb{E}[f(\psi + rAu)(uu^\top - I_d)],$$

Another natural extension of CARS in this direction is to use a population of size greater than one to estimate the Newton vector in (2.7). This yields

$$\vec{n}(\psi) \approx -2A \left(\sum_{i=1}^m h_r(\psi; Au_i)(u_i u_i^\top - I_d) \right)^{-1} \left(\sum_{i=1}^m d_r(\psi; Au_i)u_i \right), \tag{2.8}$$

where the inverse of the sum of m matrices in (2.8) can be efficiently computed through, for instance, the Woodbury matrix formula. We leave the exploration of thes ideas to future work.

2.1.3 Main Contributions

We propose a simple and lightweight zeroth-order algorithm: CARS. To derive convergence rates for CARS we use a novel convergence analysis that hinges on the insight that CARS need only significantly decrease the objective function on a positive proportion of iterations. Our results allow for a Hölder continuous Hessian—a weaker assumption than the Lipschitz continuity typically considered in such settings. We also propose a cubic-regularized variant, CARS-CR. The analysis of CARS-CR extends that of the Stochastic Subspace Newton method [HDN20] to the zeroth-order setting. The key ingredient is a careful handling of the errors introduced by replacing directional derivatives with their finite difference counterparts. We further propose an additional variant, CARS-NQ, in combination with numerical quadrature. This variant facilitates obtaining more precise approximation, thus permitting a larger smoothing parameter. Our theoretical results are corroborated by rigorous benchmarking on two datasets: Moré-Garbow-Hillstrom [MGH81] and CUTEst [GOT15]. The benchmark results demonstrate that CARS outperforms existing line-search based ZOO algorithms. Our result is accompanied by an open-source implementation of CARS (and its variant), available online at `https://github.com/bumsu-kim/CARS`.

2.2 Curvature-Aware Random Search

Given u_k sampled from $\mathcal{D}$, consider the one-dimensional Taylor expansion:

$$T_2(\alpha; x_k, u_k) := f(x_k) + \alpha u_k^\top g_k + \frac{\alpha^2}{2} u_k^\top H_k u_k \approx f(x_k + \alpha u_k). \tag{2.9}$$

CARS selects $\alpha_k \approx \arg\min_\alpha T_2(\alpha; x_k, u_k)$. The exact minimizer $u_k^\top g_k / u_k^\top H_k u_k$ depends on unavailable quantities. CARS uses $\alpha_k = d_{r_k}/h_{r_k}$, where d_r and h_r are finite difference approximations:

$$d_{r_k}(x_k; u_k) := \frac{f(x_k + r_k u_k) - f(x_k - r_k u_k)}{2r_k} = u_k^\top g_k + \mathcal{O}(r_k^2 \|u_k\|^2), \tag{2.10}$$

$$h_{r_k}(x_k; u_k) := \frac{f(x_k + r_k u_k) - 2f(x_k) + f(x_k - r_k u_k)}{r^2} = u_k^\top H_k u_k + \mathcal{O}(r_k^2 \|u_k\|^2). \tag{2.11}$$

(We write d_{r_k} and h_{r_k}, in place of $d_{r_k}(x_k; u_k)$ and $h_{r_k}(x_k; u_k)$ when x_k and u_k are clear from context.) Thus each iteration of CARS is a zeroth-order analogue of a single iteration of Newton's method applied to f restricted to the line spanned by u_k. As is well-known [NP06], pure Newton's method may not converge. So, following [GKL19] we add a fixed step-size $1/\hat{L}$ and define:

$$x_{\text{CARS},k} = x_k - \frac{d_{r_k}}{\hat{L} h_{r_k}} u_k. \tag{2.12}$$

We allow the distribution $\mathcal{D}$ to be iteration dependent, *i.e.* u_k can be sampled from $\mathcal{D}_k$. In computing $d_{r_k}(x_k)$ and $h_{r_k}(x_k)$, CARS queries f at the symmetric points $x_k + r_k u_k$ and $x_k - r_k u_k$. We extend STP [BGR20] into a *safeguarding mechanism* for CARS and choose the next iterate

$$x_{k+1} = \arg\min\{f(x_{\text{CARS},k}), f(x_k), f(x_k - r_k u_k), f(x_k + r_k u_k)\},$$

which ensures monotonicity: $f(x_0) \geq f(x_1) \geq f(x_2) \geq \cdots$. CARS requires two input parameters, $\hat{L}$ and C. Ideally, $\hat{L}$ should be the relative smoothness parameter (see Lemma 2.1.1), although CARS-CR (see Section 2.3) introduces a mechanism for selecting $\hat{L}$ adaptively. The selection of C is the subject of the next section.

Algorithm 1 **C**urvature-**A**ware **R**andom **S**earch (CARS)

Input: x_0: initial point; $\hat{L}$: relative smoothness parameter, C: scale-free sampling radius limit.
Get the oracle $f(x_0)$.
for $k = 0$ to K **do**
    Sample u_k from $\mathcal{D}_k$.
    Set $r_k \leq C/\|u_k\|$.
    Evaluate and store $f(x_k \pm r_k u_k)$.
    Compute d_{r_k} and h_{r_k} using (2.10) and (2.11).
    Compute $x_{\text{CARS},k} = x_k - \frac{d_{r_k}}{\hat{L} h_{r_k}} u_k$.
    $x_{k+1} = \arg\min\{f(x_{\text{CARS},k}), f(x_k), f(x_k - r_k u_k), f(x_k + r_k u_k)\}$.
end for
Output: x_K: estimated optimum point.

2.2.1 Convergence Guarantees

Before proceeding we list two necessary assumptions on $\mathcal{D}_k$. To describe the assumptions, introduce:

$$\eta(g, H; \mathcal{D}) = \mathbb{E}_{u \sim \mathcal{D}} \left[\frac{(u^\top g)^2}{(u^\top H u)(g^\top H^{-1} g)} \right]. \tag{2.13}$$

By Cauchy-Schwarz $\eta(g, H; \mathcal{D}) \leq 1$ for all g, $\mathcal{D}$, and positive definite H. We use η to measure the quality of the sampling distribution $\mathcal{D}$ with respect to the Newton vector $H^{-1}g$, and it is exactly 1 when all $u \sim \mathcal{D}$ are parallel to $H^{-1}g$. Our analysis assumes $\eta(g, H; \mathcal{D})$ is bounded away from zero, and this property holds for common choices of $\mathcal{D}$ as shown in Lemma 2.2.3. Since replacing $(r_k, \mathcal{D}_k)$ by $(\beta^{-1} r_k, \beta \mathcal{D}_k)$, for any $\beta > 0$, will not affect CARS, we use the *scale-free sampling radius*, $r_k \|u_k\|$, and define the following constants depending on the Hölder continuity of H:

$$C_{1,a} = \left(\frac{(a+1)(a+2)}{2^{1/2+a} L_a} \right)^{1/(1+a)} \quad \text{and} \quad C_{2,a} = \left(\frac{(a+1)(a+2)}{4(\sqrt{2}+1) L_a} \right)^{1/a}.$$

Our analysis requires us to define the following sampling radius limit, C, which also depends on the target accuracy ε and a free parameter $\gamma \in (0, 1]$:

$$C := \min\{C_{1,a}(\gamma\sqrt{2\mu\varepsilon})^{1/(1+a)}, C_{2,a}\mu^{1/a}\}. \tag{2.14}$$

CARS uses C to choose the sampling radius $r_k\|u_k\|$ *after* sampling u_k (see Line 6 of Algorithm 1). For instance, when H is Lipschitz continuous, this rule gives $r_k\|u_k\| = \mathcal{O}(\varepsilon^{1/4})$. Note that C is scale-invariant, *i.e.* replacing (f, ε) by $(\lambda f, \lambda\varepsilon)$ for any $\lambda > 0$ does not change C.

Theorem 2.2.1 (Expected descent of CARS)**.** *Suppose f is μ-strongly convex and its Hessian, H, is a-Hölder continuous. Suppose further that $\eta(g_k, H_k; \mathcal{D}_k) \geq \eta_0 > 0$. Let $\gamma \in (0, 1]$ and ε be the target accuracy. Take the scale-free limit of sampling radius C in* (2.14)*. Let $x_{\mathrm{CARS},k}$ be as in* (2.12) *and let $\mathcal{A}_k$ denote the event:*

$$\gamma\|u_k\|\sqrt{2\mu\varepsilon} \leq |u_k^\top g_k|. \tag{2.15}$$

Then,

$$\mathbb{E}\left[f(x_{\mathrm{CARS},k}) - f_\star \mid \mathcal{A}_k\right] \leq \left(1 - \eta_0\frac{\hat{\mu}}{2\hat{L}}\right)(f(x_k) - f_\star). \tag{2.16}$$

In words, by limiting the sampling radius to C, and conditioning on u_k being "good enough" (*i.e.* $\mathcal{A}_k$ occurs,) we obtain linear descent in expectation. The proof of Theorem 2.2.1 can be found in Section 2.6. Although $\mathcal{A}_k$ does not occur with probability 1, we show $\mathcal{A}_k$ occurs for a positive fraction of CARS iterations. When $\mathcal{A}_k$ does not occur, the safeguarding mechanism (Line 10 of Algorithm 1) still ensures monotonicity: $f(x_{k+1}) \leq f(x_k)$. This reveals the key idea behind CARS: *it exploits good search directions u_k when they arise yet is robust against poor search directions.* Carefully quantifying this intuition, we have:

Corollary 2.2.2 (Convergence of CARS)**.** *Take the assumptions of Theorem 2.2.1. Suppose*

further that there exists $\gamma \in (0, 1]$ *such that*

$$p_\gamma := \inf_{k \geq 0} \mathbb{P}_{u_k \sim \mathcal{D}_k} \left[|u_k^\top g_k| \geq \gamma \|u_k\| \|g_k\| \right] > 0 \tag{2.17}$$

for all $k \geq 0$*, and use* γ *to define* C *in* (2.14)*. Then, Algorithm 1 converges linearly. More specifically, for any*

$$K \geq \frac{2\hat{L}}{\eta_0 p_\gamma \hat{\mu}} \log \left(\frac{f(x_0) - f_\star}{\varepsilon} \right),$$

we have $\mathbb{E}[f(x_K)] - f_\star \leq \varepsilon$*.*

The additional assumption on $\mathcal{D}_k$ *i.e.* the existence of γ, is very mild, and is discussed in Sec. 2.2.2.

2.2.2 Further Results on the Sampling Distribution

The speed of convergence of CARS depends crucially on the lower bounds η_0 and p_γ (see (2.13) and (2.17)). The following Lemma computes η_0 for several commonly used distributions.

Lemma 2.2.3. *1. (Isotropic distributions) When*

$$\mathcal{D} = \mathrm{Unif}(\mathbb{S}^{d-1}), \quad \mathrm{Unif}(\{e_1, \cdots, e_d\}), \quad \mathcal{N}(0, I_d), \quad \text{or} \quad \mathrm{Unif}(\{\pm 1\}^d),$$

we have $\eta(g, H; \mathcal{D}) \geq \mu/(dL)$*. The distributions in the above equation are uniform on sphere, coordinate directions, Gaussian, and Rademacher, respectively.*

2. (Approximate gradient direction) If $\mathcal{D}$ *satisfies*

$$\mathbb{E}_{u \sim \mathcal{D}} \left[\left(\frac{|u^\top g|}{\|u\| \|g\|} \right)^2 \right] \geq \beta > 0 \tag{2.18}$$

for some $\beta > 0$*, then* $\eta(g, H; \mathcal{D}) \geq \beta\mu/L$*.*

3. *(Newton direction) When u is parallel to $H^{-1}g$ with probability 1, we have $\eta(g, H; \mathcal{D}) = 1$.*

Proof. Since $u^\top Hu \le L\|u\|^2$ and $g^\top H^{-1} g \le \mu^{-1}\|g\|^2$,

$$\eta(g, H; \mathcal{D}) \ge \frac{\mu}{L}\, \mathbb{E}_{u\sim\mathcal{D}}\left[\left(\frac{|u^\top g|}{\|u\|\|g\|}\right)^2\right]. \tag{2.19}$$

1. When $\mathcal{D} = \mathcal{N}(0, I_d)$ or $\mathrm{Unif}(\mathbb{S}^{d-1})$, we can replace g by the standard basis vector e_1 by symmetry, and it immediately follows that $\eta(g, H; \mathcal{D}) \ge \mu/(dL)$. When $\mathcal{D} = \mathrm{Unif}(\{e_1, \cdots, e_d\})$,

$$\mathbb{E}_{u\sim\mathcal{D}}\left[\left(\frac{|u^\top g|}{\|u\|\|g\|}\right)^2\right] = \frac{1}{d}\sum_{i=1}^{d} |g_i|^2/\|g\|^2 = \frac{1}{d}$$

   and when $\mathcal{D} = \mathrm{Unif}(\{\pm 1\}^d)$,

$$\mathbb{E}_{u\sim\mathcal{D}}\left[\left(\frac{|u^\top g|}{\|u\|\|g\|}\right)^2\right] = \frac{1}{2^d}\sum_{u\in\{\pm1\}^d} \frac{\sum_{i=1}^d |g_i|^2 + \sum_{i\ne j} u_i u_j g_i g_j}{d\|g\|^2} = \frac{1}{d}.$$

   Hence, again from (2.19), we have the same lower bound $\mu/(dL)$.

2. When (2.18) holds, (2.19) provides the lower bound $\eta(g, H; \mathcal{D}) \ge \beta\mu/L$. In particular, when u is parallel to g (*i.e. gradient direction*) with probability p, then $\eta \ge p\mu/L$.

3. When u is the Newton direction, *i.e.* u is parallel to $H^{-1}g$ with probability 1, $u^\top g = u^\top Hu = g^\top H^{-1} g$, and so $\eta(g, H; \mathcal{D}) = 1$.

This finishes the proof. □

Lemma 2.2.3 suggests that assuming $\eta(g_k, H_k; \mathcal{D}_k) \ge \eta_0 > 0$ for all $k \ge 0$ is reasonable in practice. Note Case 3 yields the best possible η, as $\eta \le 1$ by Cauchy-Schwarz. The next Lemma suggests that assuming $p_\gamma > 0$ is also reasonable in practice.

Lemma 2.2.4 (Estimation and Lower Bounds of p_γ for Various Distributions)**.**

1. *(Uniform on sphere and Gaussian) When $\mathcal{D} = \mathcal{N}(0, I_d)$ or $\mathrm{Unif}(\mathbb{S}^{d-1})$ we have*

$$\mathbb{P}_{u\sim\mathcal{D}}\left[|u^\top g| \geq \gamma\|u\|\|g\|\right] = I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2}\right). \tag{2.20}$$

   In particular, for $d \geq 2$, $\mathbb{P}_{u\sim\mathcal{D}}\left[|u^\top g| \geq \|u\|\|g\|/\sqrt{d}\right] \geq 0.315603$.

2. *(Random coordinate direction) When $\mathcal{D} = \mathrm{Unif}(\{e_1, \cdots, e_d\})$ we have*

$$\mathbb{P}_{u\sim\mathcal{D}}\left[|u^\top g| \geq \|u\|\|g\|/\sqrt{d}\right] \geq 1/d.$$

Proof. 1. First note that we can assume $\|u\| = 1$ in (2.20), and thus we only need to consider the case $\mathcal{D} = \mathrm{Unif}(\mathbb{S}^{d-1})$. In this case, $\mathcal{D}$ is invariant under rotation so we can take $g = e_1$ and

$$\mathbb{P}_{u\sim\mathcal{D}}\left[|u^\top g| \geq \gamma\|u\|\|g\|\right] = \mathbb{P}[|u_1| \geq \gamma] = I_{1-\gamma^2}\left(\frac{d-1}{2}, \frac{1}{2}\right)$$

   where I is the regularized incomplete Beta function as in [CMY22a, Theorem 2.3]. In particular, when $\gamma = 1/\sqrt{d}$, the function $d \mapsto I_{1-1/d}\left(\frac{d-1}{2}, \frac{1}{2}\right)$ is decreasing for $d \geq 2$ and bounded below by 0. Thus $p_\gamma \geq \lim_{d\to\infty} I_{1-1/d}\left(\frac{d-1}{2}, \frac{1}{2}\right) = 0.315603\cdots$.

2. When $\mathcal{D} = \mathrm{Unif}\{e_1, \cdots, e_d\}$,

$$\mathbb{P}_{u\sim\mathcal{D}}\left[|u^\top g| \geq \gamma\|u\|\|g\|\right] = \frac{1}{d}\sum_{i=1}^{d} 1_{|g_i|\geq\gamma\|g\|}.$$

   Recall that $\|g\|^2 = \sum_i |g_i|^2$. Hence, we have $\max_i |g_i| \geq \|g\|/\sqrt{d}$, which implies $\mathbb{P}_{u\sim\mathcal{D}}\left[|u^\top g| \geq \|u\|\|g\|/\sqrt{d}\right] \geq 1/d$. Note that this bound is tight; the equality holds when, for example, $g = (1, 0, \cdots, 0)$.

This finishes the proof. □

When γ is small enough and $\mathcal{D}_k$ approximates the gradient or Newton direction close enough, both $\eta_{\mathcal{D}_k}$ and p_γ do not depend on d, leading to dimension independent convergence rates. So, CARS can be combined with other derivative-free techniques that estimate the gradient (or Newton direction)—at the cost of two additional function queries per iteration CARS will choose an approximately optimal step-size in this computed direction. Our analysis easily extends to such combined methods, and we sketch how to do so for the widely used [NS17, SHC17, CRS18, FGK18] variance-reduced Nesterov-Spokoiny gradient estimate:

$$\tilde{g}_k := \frac{1}{m}\sum_{i=1}^{m} d_r(x_k; u_k)u_k \approx g_k. \tag{2.21}$$

For simplicity, we assume access to exact directional derivatives (as in [NS17]).

Corollary 2.2.5. *Let f be μ-strongly convex and H be a-Hölder continuous. Suppose, at each step, u_k is generated by first sampling $v_1, \cdots, v_m$ from Gaussian distribution $\mathcal{N}(0, I_d)$ and defining:*

$$u_k = \frac{1}{m}\sum_{j=1}^{m}(g_k^\top v_j)v_j.$$

Then CARS (Algorithm 1) finds x_K with $\mathbb{E}[f(x_K)] - f_\star \leq \varepsilon$ if

$$K \geq \frac{2\hat{L}L(m+d+1)}{\hat{\mu}\mu m p_\gamma}\log\left(\frac{f(x_0)-f_\star}{\varepsilon}\right).$$

Proof. From Lemma 2.2.3 (part 2) we obtain:

$$\eta_{\mathcal{D}} \geq \frac{\mu m}{L(m+d+1)}.$$

Combining this with Corollary 2.2.2 yields the claim. □

2.3 CARS with Cubic Regularization for General Convex Functions

Here, we adopt cubic regularization [NP06, HDN20], a technique to achieve global convergence of a second-order method for convex functions, in CARS and prove convergence. We drop strong convexity and assume only L-smoothness. We assume Lipschitz continuity of the Hessian (*i.e.* $a = 1$ in Assumption 1) and let $M = L_1$ be the Lipschitz constant. Instead of using the second-order Taylor expansion (2.9), we now use

$$P(\alpha; d, h) := d\alpha + \frac{1}{2}h\alpha^2 + \frac{M}{6}|\alpha|^3, \tag{2.22}$$

with the exact derivatives $P(\,\cdot\,; d_0, h_0)$ and the finite difference approximations $P(\,\cdot\,; \pm d_{r_k}, h_{r_k})$. The method of Stochastic Subspace Cubic Newton (SSCN) [HDN20] takes exact derivatives and uses the following inequality [HDN20, Lemma 2.3]

$$f(x_k + \alpha u_k) \leq f(x_k) + P(\alpha; d_0(x_k; u_k), h_0(x_k; u_k)) \tag{2.23}$$

to derive the algorithm $x_{k+1} = x_k + \hat{\alpha}_k u_k$, where $\hat{\alpha}_k = \arg\min_\alpha P(\alpha; d_0, h_0)$. We propose using $\alpha_k^{\pm} = \arg\min_\alpha P(\alpha; \pm d_{r_k}, h_{r_k})$ in place of $\hat{\alpha}_k$. By solving $P'(\alpha; \pm d_{r_k}, h_{r_k}) = 0$ we obtain

$$\alpha_k^{\pm} = -\frac{\pm 2 d_{r_k}}{h_{r_k} + \sqrt{h_{r_k}^2 + 2M|d_{r_k}|}}.$$

This step-size is equal to $-\frac{\pm d_{r_k}}{h_{r_k}\hat{L}_k}$ with

$$\hat{L}_k = \frac{1}{2} + \sqrt{\frac{1}{4} + \frac{M|d_{r_k}|}{2h_{r_k}^2}}, \tag{2.24}$$

so it is CARS with this varying relative smoothness constant. We formalize this as Algorithm 2.

To analyze CARS-CR (Algorithm 2), we make a boundedness assumption.

Definition 3. Recall that $\mathcal{Q} = \{x \in \mathbb{R}^d : f(x) \leq f(x_0)\}$. We say f has an $\mathcal{R}$-*bounded level*

Algorithm 2 CARS with **C**ubic **R**egularization (CARS-CR)

1: **Input:** ε: target accuracy; x_0: initial point; r_0: initial sampling radius; M: Lipschitz constant of Hessian.
2: Get the oracle $f(x_0)$.
3: **for** $k = 0$ to K **do**
4: Sample u_k from $\mathcal{D}_k$.
5: Set $r_k \leq \rho\sqrt{\varepsilon}/\sqrt{k+2}$ where $\rho = R/\sqrt{2B}$ as defined in Theorem 2.3.3.
6: Evaluate and store $f(x_k \pm r_k u_k)$.
7: Compute d_{r_k} and h_{r_k} using (2.10) and (2.11).
8: Compute $\hat{L}_k$ using (2.24).
9: Compute $x_{\mathrm{CR}\pm,k} = x_k \pm \frac{d_{r_k}}{\hat{L}_k h_{r_k}} u_k$.
10: $x_{k+1} = \arg\min\{f(x_{\mathrm{CR}+,k}), f(x_{\mathrm{CR}-,k}), f(x_k), f(x_k - r_k u_k), f(x_k + r_k u_k)\}$.
11: **end for**
12: **Output:** x_K: estimated optimum point.

set if the diameter of $\mathcal{Q}$ is $\mathcal{R} < \infty$.

Without loss of generality, we may assume the distribution is normalized (*i.e.* $\|u\| = 1$ w.p. 1.) This is because we only need to bound the scale-free sampling radius $r_k\|u_k\|$, as before. To ensure that the finite difference error is insignificant, the sampling radius needs to be small enough. However, for a more concise analysis, it is helpful to have an upper bound that can be chosen arbitrarily. Let $R > 0$ be an upper bound of r_k for all $k \geq 0$. Note that any r_k selected by CARS-CR automatically satisfies $r_k \leq R$ (see line 5 of Algorithm 2). Using this notation, we get:

Lemma 2.3.1 (Finite difference error bound for the minimum of P)**.** *Let* $P(\,\cdot\,) = P(\,\cdot\,; d_0, h_0)$. *Then for any* $0 \leq r_k \leq R$,

$$\min(|P(\hat{\alpha}_k) - P(\alpha_k^+)|,\, |P(\hat{\alpha}_k) - P(\alpha_k^-)|) \leq \frac{2B}{R^2} r_k^2, \tag{2.25}$$

where $B = \max(LR^2, MR^3, f(x_0) - f_\star)$.

If the sampling distribution is isotropic in expectation, *i.e.* it satisfies $\mathbb{E}\left[u_k u_k^\top\right] = \frac{1}{d} I_d$,, we get the following descent lemma:

Theorem 2.3.2 (Expected descent of CARS-CR)**.** *Suppose f is convex, L-smooth, and has M-Lipschitz Hessian. If $\mathcal{D}_k$ is isotropic in expectation, then with Algorithm 2, we have*

$$\mathbb{E}\left[f(x_{k+1}) \mid x_k\right] \leq \left(1 - \frac{1}{d}\right) f(x_k) + \frac{1}{d} f(x_k + z) + \frac{L}{2d}\|z\|^2 + \frac{M}{6d}\|z\|^3 + \frac{2B}{R^2} r_k^2 \tag{2.26}$$

for any $z \in \mathbb{R}^d$.

Finally, with decreasing r_k as given in Algorithm 2, we obtain the $\mathcal{O}(k^{-1})$ convergence rate for CARS-CR.

Theorem 2.3.3 (Convergence of CARS-CR)**.** *Under the assumptions of Theorem 2.3.2, further assume f has an $\mathcal{R}$-bounded level set. Set $r_k \leq \frac{\rho\sqrt{\varepsilon}}{\sqrt{k+2}}$ where $\rho = \frac{R}{\sqrt{2B}}$. Then, with Algorithm 2, we have*

$$\begin{aligned} \mathbb{E}[f(x_K)] - f_\star \leq \frac{s^s (f(x_0) - f_\star)(1 + \log(K+2))}{(K/d)^{s+1}} + \frac{e^{s/K}(s+1)^2 L\mathcal{R}^2}{2s(K/d)} \\ + \frac{e^{(s-1)/K}(s+1)^3 M\mathcal{R}^3}{6(s-1)(K/d)^2} + \frac{e^{2(s+1)/K}}{s+1}\varepsilon \end{aligned} \tag{2.27}$$

for any $s > 1$. That is, for any $0 < p < 1$ there exists $C_p > 0$ such that $\mathbb{E}[f(x_K)] - f_\star \leq \varepsilon$ if

$$K \geq C_p d \max\left\{\frac{L\mathcal{R}^2}{\varepsilon}, \sqrt{\frac{M\mathcal{R}^3}{\varepsilon}}, \left(\frac{f(x_0) - f_\star}{\varepsilon}\right)^p\right\}. \tag{2.28}$$

2.4 Incorporating Numerical Quadrature

Recall that the *Gaussian smoothing* of f is defined as

$$\mathcal{G}_r[f](x) = \mathbb{E}_{u \sim \mathcal{N}(0, I_d)}[f(x + ru)].$$

As discussed in [NS17] and elsewhere

$$\nabla \mathcal{G}_r[f](x) = r^{-1}\mathbb{E}[f(x + ru)u] = \mathbb{E}[d_r(x; u)u],$$

thus $d_r(x;u)u$ is an unbiased estimator of $\nabla\mathcal{G}_r[f]$. This fact is key in proving the convergence of NS random search [NS17]. Consider a single step of Newton's method to a *smoothed, one-dimensional* function, by slicing f in a fixed direction $\hat{u}_k \in S^{d-1}$. That is, consider $u_k = t_k\hat{u}_k$ with $t_k \sim \mathcal{N}(0,1)$, and define the one-dimensional function $\tilde{f}(t;x_k,\hat{u}_k) = f(x_k + t\hat{u}_k)$. Then its Gaussian smoothing is given by

$$\mathcal{G}_r[\tilde{f}(\cdot;x_k,\hat{u}_k)](s) = \mathbb{E}_{t\sim\mathcal{N}(0,1)}[\tilde{f}(s+rt;x_k,\hat{u}_k)].$$

For simplicity, we omit x_k and $\hat{u}_k$ when they are clear from context. Note that d_r is an unbiased estimator of $\mathcal{G}_r[\tilde{f}]'(0)$, and h_r is a (biased) estimator of $\mathcal{G}_r[\tilde{f}]''(0)$. Thus, the CARS step $x_{\text{CARS}} = x_k - d_r/\hat{L}h_r$ actually mirrors a single step of Newton's method, starting at $s = 0$ for $\mathcal{G}_r[\tilde{f}](s)$.

In this section we propose a variant of CARS which uses better estimators of $\mathcal{G}_r[\tilde{f}]'(0)$ and $\mathcal{G}_r[\tilde{f}]''(0)$. By integration by parts[3]:

$$\mathcal{G}_r[\tilde{f}]'(s) = r^{-1}\mathbb{E}_{t\sim\mathcal{N}(0,1)}\left[t\tilde{f}(s+rt)\right], \tag{2.29}$$

$$\mathcal{G}_r[\tilde{f}]''(s) = r^{-2}\mathbb{E}_{t\sim\mathcal{N}(0,1)}\left[(t^2-1)\tilde{f}(s+rt)\right]. \tag{2.30}$$

As these integrals are *one-dimensional* they may be accurately approximated using as few as three queries with *Gauss-Hermite* (GH) quadrature. The following is a simple consequence of [AS64].

Lemma 2.4.1. *Let $\{(t_i,w_i)\}_{i=1}^q$ be the GH quadrature points and weights. Define*

$$d_{r,q}^{\text{NQ}} = \frac{1}{r\sqrt{\pi}}\sum_{i=1}^{q} w_i\sqrt{2}t_i\tilde{f}(\sqrt{2}rt_i), \tag{2.31}$$

$$h_{r,q}^{\text{NQ}} = \frac{1}{r^2\sqrt{\pi}}\sum_{i=1}^{q} w_i(2t_i^2-1)\tilde{f}(\sqrt{2}rt_i). \tag{2.32}$$

[3]Again, this can also by deduced from Stein's formula [Ste72, Ste81]

Then $d_{r,q}^{\mathrm{NQ}} - \mathcal{G}_r[\tilde{f}]'(0) = \mathcal{O}(r^{2q-2})$ *and* $h_{r,q}^{\mathrm{NQ}} - \mathcal{G}_r[\tilde{f}]''(0) = \mathcal{O}(r^{2q-4})$.

Here the constant for the $\mathcal{O}$-notation depends on $\tilde{f}$ and q. See [AS64] for numerical expressions for t_i and w_i. Notice that when q is odd, we only need $q-1$ new function queries, since $t=0$ is one of the quadrature points. Using $d_{r,q}^{\mathrm{NQ}}, h_{r,q}^{\mathrm{NQ}}$ allows one to obtain extremely accurate approximations to $\mathcal{G}_r[\tilde{f}]'(s)$ and $\mathcal{G}_r[\tilde{f}]''(s)$ while using a much larger r. Although large r is not necessarily desirable when f is strongly convex, it can be very useful when f is non-convex of the form

$$f(x) = f_{\mathrm{cvx}}(x) + f_{\mathrm{osc}}(x), \tag{2.33}$$

where f_{cvx} is strongly convex while f_{osc} is rapidly oscillating, *i.e.* $f_{\mathrm{osc}}(x) = \psi(x)\cos(\lambda\phi(x))$ where ψ and ϕ are smooth and $\|\psi\|_\infty < \infty$.

Theorem 2.4.2. *Suppose* f *is non-convex of the form* (2.33). *For any fixed* $\hat{u}$ *consider* $\tilde{f}_{\mathrm{cvx}}(t; x, \hat{u}) = f_{\mathrm{cvx}}(x + t\hat{u})$ *and* $\mathcal{G}_r[\tilde{f}_{\mathrm{cvx}}](s) = \mathbb{E}_{t\sim\mathcal{N}(0,1)}[\tilde{f}_{\mathrm{cvx}}(s + rt; x, \hat{u})]$. *If* $\tilde{\phi}(t) = \phi(x + t\hat{u})$ *satisfies* $|\tilde{\phi}'(t)| \geq c$ *and* $\tilde{\phi}'$ *is monotone, then*

$$|d_{r,q}^{\mathrm{NQ}} - \mathcal{G}_r[\tilde{f}_{\mathrm{cvx}}]'(0)| = \mathcal{O}(r^{2q-2} + \frac{1}{r}), \tag{2.34}$$

$$|h_{r,q}^{\mathrm{NQ}} - \mathcal{G}_r[\tilde{f}_{\mathrm{cvx}}]''(0)| = \mathcal{O}(r^{2q-4} + \frac{1}{r^2}). \tag{2.35}$$

Proof. Note that, thanks to Lemma 2.4.1, we only need to show $\mathcal{G}_r[\tilde{f}_{\mathrm{osc}}]'(0) = \mathcal{O}(r^{-1})$ and $\mathcal{G}_r[\tilde{f}_{\mathrm{osc}}]''(0) = \mathcal{O}(r^{-1})$. Denote $\tilde{\psi}(t) = \psi(x + t\hat{u})$. Then $\tilde{f}_{\mathrm{osc}}(t) = \tilde{\psi}(t)\cos(\lambda\tilde{\phi}(t))$. Then

$$\begin{aligned}\left|\mathcal{G}_r[\tilde{f}_{\mathrm{osc}}]'(0)\right| &= (2\pi)^{-1/2} r^{-1} \left|\int_{-\infty}^{\infty} t e^{-t^2/2} \tilde{\psi}(rt)\cos(\lambda\tilde{\phi}(rt))dt\right| \\ &\leq (2\pi)^{-1/2} r^{-1} \left|\int_{-\infty}^{\infty} t e^{-t^2/2} \tilde{\psi}(rt) e^{i\lambda\tilde{\phi}(rt)}dt\right|.\end{aligned}$$

Because $\|\tilde{\psi}(t)\|_\infty < \infty$, we can bound the tail part of the integral arbitrarily small by a

smooth bump function $a_R(t)$ that vanishes for $|t| > R$, and taking a sufficiently large $R > 0$:

$$\left|\mathcal{G}_r[\tilde{f}_{\text{osc}}]'(0)\right| \leq (2\pi)^{-1/2} r^{-1} \left|\int_{-\infty}^{\infty} a_R(t) t e^{-t^2/2r^2} \tilde{\psi}(rt) e^{i\lambda\tilde{\phi}(rt)} dt\right| + \mathcal{O}(r^{-1})$$

Then we apply Lemma 2.6 of [Tao07] with $\phi(t) = \tilde{\phi}(rt)$ and the first term is also bounded by $\mathcal{O}(r^{-1})$. Similarly, for the second derivative,

$$\left|\mathcal{G}_r[\tilde{f}_{\text{osc}}]''(0)\right| = (2\pi)^{-1/2} r^{-2} \left|\int_{-\infty}^{\infty} (t^2 - 1) e^{-t^2/2} \tilde{\psi}(rt) \cos(\lambda\tilde{\phi}(rt)) dt\right|$$

and we get $\mathcal{G}_r[\tilde{f}_{\text{osc}}]''(0) = \mathcal{O}(r^{-2})$. □

Thus a judicious choice of r "smoothes out" the oscillatory part, while still accurately estimating the first and second derivatives of the strongly convex part. Each iterate of CARS with Numerical Quadrature (CARS-NQ, see Algorithm 3) effectively applies (one step of) Newton's method to f_{cvx}, while ignoring f_{osc}. This suggests, but does not prove, CARS-NQ is robust towards local minima induced by f_{osc} and will converge towards the global minimum of f (assuming it is close to the global minimum of f_{cvx}). This intuition is supported by our empirical results, which are presented in Section 2.7.

Algorithm 3 CARS with **N**umerical **Q**uadrature (CARS-NQ)

Input: x_0: initial point; r: sampling radius; q: number of quadrature points.
Get the oracle $f(x_0)$.
for $k = 0$ to K **do**
    Sample u_k from $\mathcal{D}$.
    Compute $x^i := x_k + r\sqrt{2}t_i u_k$ for $i = 1, \cdots, q$.
    Compute $d_{r,q}^{\text{NQ}}$ and $h_{r,q}^{\text{NQ}}$ using (2.31) and (2.32).
    Compute $\hat{L}_{u_k}$ using (2.36).
    Compute $x_{\text{CARS}} = x_k - \frac{d_{r,q}^{\text{NQ}}}{\hat{L}_{u_k} h_{r,q}^{\text{NQ}}}$.
    $x_{k+1} = \arg\min\{f(x^1), \cdots, f(x^q), f(x_k), f(x_{\text{CARS}})\}$.
end for
Output: x_K: estimated optimum point.

[ZBZ21, TZ20] also suggest using GH quadrature in DFO. However the underlying

principle of their proposed algorithms, DGS and AdaDGS, respectively, is quite different from CARS-NQ. They apply GH quadrature in *each coordinate direction*. The resulting estimates are then stacked into a vector, which they refer to as the Directional Gaussian Smoothed (DGS) gradient. The DGS gradient is not easily interpretable as the gradient of a function, hence analyzing the convergence of AdaDGS and DGS is tricky. Nevertheless, in practice both AdaDGS and DGS converge in relatively few iterations, although we note their per-iteration query complexity is $\Omega(d)$, making them unsuitable for high-dimensional problems.

Bounding the relative smoothness. Both CARS and CARS-NQ require the knowledge of the relative smoothness constant, $\hat{L}$, so as to set the step-size appropriately. Using NQ a proper value of $\hat{L}$ may be suggested.

Proposition 2.4.3. *Let $f : \mathbb{R} \to \mathbb{R}$ satisfy $f''(0) > 0$. Assume f is three times continuously differentiable and $M = \sup_{x\in\mathbb{R}} |f'''(x)| < \infty$. Then f satisfies*

$$f(t) - f(0) - tf'(0) \leq \frac{\tilde{L}}{2} t^2 f''(0)$$

at $t = -f'(0)/(\tilde{L}f''(0))$, where

$$\tilde{L} = \frac{1}{2} + \sqrt{\frac{1}{4} + \frac{M|f'(0)|}{3f''(0)}}.$$

Proof. From Taylor's theorem,

$$f(t) - f(0) - tf'(0) = \frac{t^2}{2} f''(0) + \frac{t^3}{6} f'''(\zeta_t).$$

for some $\zeta_t \in (0, t)$. Therefore, dividing both sides by $t^2 f''(x)/2$, we only need to show that

$$1 + \frac{t f'''(\zeta_t)}{3 f''(0)} \leq \tilde{L}.$$

Since $t = -f'(0)/\tilde{L} f''(0)$, this is equivalent to

$$\tilde{L}^2 - \tilde{L} \geq -\frac{f'(0) f'''(\zeta_t)}{3 f''(0)}$$

However, from the definition of $\tilde{L}$,

$$\left(\tilde{L} - \frac{1}{2}\right)^2 \geq \frac{1}{4} + \frac{M|f'(0)|}{3 f''(0)}$$

and it follows that

$$\tilde{L}^2 - \tilde{L} \geq \frac{M|f'(0)|}{3 f''(0)} \geq -\frac{f'(0) f'''(\zeta_t)}{3 f''(0)}.$$

□

Although Proposition 2.4.3 does not upper bound the relative smoothness parameter over the whole domain, it provides the desired inequality at the desired point. As in (2.29) and (2.30), also the higher order derivatives of $\mathcal{G}_r[\tilde{f}]$ can be easily estimated using NQ. Hence, we suggest the following approximation to the relative smoothness parameter for each direction u:

$$\hat{L}_u \approx \frac{1}{2} + \sqrt{\frac{1}{4} + \frac{|d_{r,q}^{\mathrm{NQ}}|\, m_{r,q}^{\mathrm{NQ}}}{(h_{r,q}^{\mathrm{NQ}})^2}}, \tag{2.36}$$

where $m_{r,q}^{\mathrm{NQ}}$ denote the NQ estimator of $\mathcal{G}_r[\tilde{f}]'''(0)$.

2.5 More on Curvature: Randomized Matrix Inversion and SHIPS

Recall that the Newton vector is the solution of the linear system $Ax = b$, where $A = H(x_k)$ and $b = -g(x_k)$. As we have discussed in Sections 2.2.1 and 2.4, for any $z \in \mathbb{R}^d$, the scalar measurements $z^\top Az$ and $b^\top z$ can be easily estimated via finite difference ((2.1) and (2.2), respectively) or numerical integration (Lemma 2.4.1).

Previously, the reconstruction of the covariance matrix A from quadratic measurements of the form $u^\top Au$ has been studied, for instance, in [CCG15]. However, the proposed method necessitates a low-rank assumption on the covariance matrix and involves solving an expensive sub-problem. Contrarily, we introduce a novel approach that is easy to compute, and provides an unbiased estimator for the inverse of a matrix, up to a scalar factor. Remarkably, this is achieved through quadratic measurements without any need for additional assumptions on the matrix, apart from symmetry and positive definiteness.

In this section we present a novel randomized matrix inversion method that estimates A^{-1} and $A^{-1}b$ using *only those scalar measurements*. For the rest of this section, we assume A is a general symmetric positive definite (SPD) matrix.

Theorem 2.5.1 (Randomized Matrix Inversion)**.** *Let $A \in \mathbb{R}^{d\times d}$ be SPD, and $p = d/2 + 1$. If $u \sim \mathcal{N}(0, I_d)$ is a Gaussian random vector, and $w = u/\|u\| \sim \mathrm{Unif}(S^{d-1})$, then*

$$\mathbb{E}\left[\frac{\|u\|^d}{(u^\top Au)^p}uu^\top\right] = \mathbb{E}\left[\frac{ww^\top}{(w^\top Aw)^p}\right] = \frac{A^{-1}}{d\sqrt{\det(A)}}. \tag{2.37}$$

In fact, the determinant of the right-hand-side of (2.37), $\det(d^{-1}A^{-1}/\sqrt{\det(A)})$, computes to $d^{-d}(\det(A))^{-p}$. Therefore, provided that the estimation of the expectation is accurate enough, we can estimate $\det(A)$ and also recover A^{-1} without additional scalar factor. However, we don't need to perform this because we only need the *direction* of the Newton vector. Recall that (2.12) is invariant under scalar multiplication on u.

And the following Corollary provides the direction of the Newton vector.

Corollary 2.5.2 (Solution of the linear system through scalar measurements)**.** *Let A, p, u and w be as defined in Theorem 2.5.1. Then*

$$\mathbb{E}\left[\frac{\|u\|^d u^\top b}{(u^\top A u)^p} u\right] = \mathbb{E}\left[\frac{w^\top b}{(w^\top A w)^p} w\right] = \frac{A^{-1}b}{d\sqrt{\det(A)}}. \tag{2.38}$$

When the direction u that we sample is parallel to the Newton vector, $\vec{n}$, CARS recovers the full Newton's method, assuming that the finite difference approximation for directional derivatives are exact. Combining this fact with the above stochastic Hessian inversion, which can provide u whose direction is close to $\vec{n}$, we propose the following algorithm, coined Stochastic Hessian Inversion for Projected Search (SHIPS).

2.5.1 Discussion on SHIPS

One immediate advantage of SHIPS is the algorithm's potential for parallelization. The oracles at each iteration can be computed in parallel effortlessly. Furthermore, unlike other Newton-type methods, it does not need to store the whole Hessian (or its inverse) in memory.

The gradient estimation g_k in line 6 of Algorithm 4 can be achieved via several methodologies, including linear interpolation or smoothing method. For example, when the number of samples $m \leq d$, an estimation through linear interpolation can be given by solving $2rU_k g_k = F_k$, where U_k is an $m \times d$ matrix with the i-th row being $u_{k.i}$, and F_k is a d-vector whose i-th entry is $f(x_k + ru_{k,i}) - f(x_k - ru_{k,i})$. When a solution exists for this linear system, $u_{k,i}^\top g_k$ is just the directional finite difference, $(F_k)_i/(2r)$. In such a case, there is no need to solve the linear system; the line 6 can be bypaseed, and directly proceeding to line 7.

Alternatively, g_k can be estimated via a smoothing method incorporating Monte Carlo estimation. We have [BCC21]

$$\nabla F(x) = \frac{d}{r}\,\mathbb{E}_{\mathrm{Unif}(S^{d-1})}[f(x+\mu u)u],$$

Algorithm 4 **S**tochastic **H**essian **I**nversion for **P**rojected **S**earch (SHIPS)

Input: x_0: initial point; μ: sampling radius; M: the number of samples per iteration; $\hat{L}$: relative smoothness parameter
Get the oracle $f(x_0)$.
for $k = 0$ to K **do**
    Sample *i.i.d.* directions $u_{k,i} \sim \mathrm{Unif}(S^{d-1})$, $i = 1, \cdots, m$.
    Get oracles $f(x_k \pm r u_{k,i})$ for $i = 1, \cdots, m$ ($2m$ queries).
    Estimate g_k using the $2m$ measurements above.
    Compute the sample mean of (2.38) to estimate the direction of the Newton vector

$$\hat{u}_k = \frac{1}{m} \sum_{i=1}^{m} \frac{u_{k,i}^\top g_k}{(h_r(x_k; u_{k,i}))^p} u_{k,i}.$$

    Set $u_k = \hat{u}_k / \|\hat{u}_k\|$.
    Perform additional queries along u_k to conduct CARS(-CR) (finite difference, 3(or 4) more queries)

$$x_{k,\mathrm{CARS}} = x_k - \frac{d_r(x_k; u_k)}{\hat{L} h_r(x_k; u_k)} u_k,$$

    or CARS-NQ (GH quadrature, $2q + 1$ more queries)

$$x_{k,\mathrm{CARS}} = x_k - \frac{\mathcal{G}_r[\tilde{f}]'(0)}{\hat{L}_{u_k} \mathcal{G}_r[\tilde{f}]''(0)} u_k.$$

    $x_{k+1} = \arg\min\{f(x_k), f(x_k \pm r u_{k_i}), f(x_{k,\mathrm{CARS}})\}$.
end for
Output: x_K: estimated optimum point.

where $F(x) = \mathbb{E}_{v \sim \mathrm{Unif}(B^d)}[f(x + rv)]$ is smoothed version of f on a ball of radius r. We refer the interested reader to [BCC21] for a comparison between the interpolation method and the smoothing method.

2.5.2 Enhancing the Quality of Estimation via Adaptive Sampling

Let B be a SPD matrix and consider $v \sim \mathcal{N}(0, B^{-2})$ by taking $v = B^{-1}u$, where u is sampled from the standard normal distribution. Then Theorem 2.5.1 can be rewritten as an expectation over v.

Algorithm 5 **R**andomized **I**nversion with **A**daptive **S**ampling (RIAS)

Input: M: the number of samples per iteration;
Get $M_0 \approx A^{-1}$ using (2.37).
for $k = 0$ to K **do**
    Find $B_k^{-1} = M_0^{1/2}$.
    Get $M_k \approx A^{-1}$ using (2.39) with $B = B_k$.
end for
Output: M_K: estimated inverse of A.

Corollary 2.5.3 (Randomized Matrix Inversion with $\mathcal{N}(0, B^{-2})$)**.** *Let A, p, u be as defined in Theorem 2.5.1. If B is SPD and $v = B^{-1}u$, then*

$$\mathbb{E}_{v\sim\mathcal{N}(0,B^{-2})}\left[\frac{\|Bv\|^d}{(v^\top Av)^p}vv^\top\right] = \frac{\det(B)}{d\sqrt{\det(A)}}A^{-1}. \tag{2.39}$$

Proof. Let $v = B^{-1}u$. Then $v \sim \mathcal{N}(0, B^{-2})$ and

$$\begin{aligned}\mathbb{E}_{v\sim\mathcal{N}(0,B^{-2})}\left[\frac{\|Bv\|^d}{(v^\top Av)^p}vv^\top\right] &= \mathbb{E}_{u\sim\mathcal{N}(0,I_n)}\left[\frac{\|u\|^d}{(u^\top B^{-\top}AB^{-1}u)^p}B^{-1}uu^\top B^{-1}\right]\\ &= \frac{1}{d\sqrt{\det(B^{-\top}AB^{-1})}}A^{-1}\\ &= \frac{\det(B)}{d\sqrt{\det(A)}}A^{-1}.\end{aligned}$$

□

An intriguing case occurs when $B = A^{1/2} = PD^{1/2}P^\top$. This results in a similar approach to evolution strategies where the covariance matrix approximates the inverse Hessian [YHF18, SY20]. We propose the adaptive sampling procedure based on this idea in Algorithm 5.

However, the significance of the improvement resulting from this adaptive sampling remains uncertain in higher dimensions. This may be due to a large variance in estimating the inverse matrix. This aspect is left for future investigation.

2.6 Proofs

Here we collect the proofs of the results of Sections 2.2.1 and 2.3, and state and prove some auxiliary lemmas needed in the proofs of the main results. We begin with a lemma quantifying the expected descent given access to exact derivatives.

2.6.1 Proofs for Results in Section 2.2.1

Lemma 2.6.1 (Expected descent of CARS with exact derivatives)**.** *Let $u_k \sim \mathcal{D}_k$ and $x_{\mathrm{ED},k}$ be the CARS step with exact derivatives*

$$x_{\mathrm{ED},k} = x_k - \frac{u_k^\top g_k}{\hat{L} u_k^\top H_k u_k} u_k. \tag{2.40}$$

Then letting $\eta_k = \eta(g_k, H_k; \mathcal{D}_k)$,

$$\mathbb{E}\left[f(x_{\mathrm{ED},k}) \mid x_k\right] - f_\star \leq \left(1 - \eta_k \frac{\hat{\mu}}{\hat{L}}\right)(f(x_k) - f_\star). \tag{2.41}$$

Remark 2.6.2. Lemma 2.6.1 is similar to [GKL19, Corollary 1] and [KBD21, Corollary 1 part (ii)]. However, Lemma 2.6.1 allows for more general sampling distributions $\mathcal{D}$.

Proof. From $\hat{\mu}$-relative strong convexity we have

$$f_\star - f(x_k) \geq \langle g_k, x_\star - x_k \rangle + \frac{\hat{\mu}}{2}\|x_\star - x_k\|_{H_k}^2 \geq -\frac{1}{2\hat{\mu}}\|g_k\|_{H_k^{-1}}^2, \tag{2.42}$$

where the second inequality follows by taking $x = x_\star - x_k$ and $c = \hat{\mu}$ in the following general inequality [GKL19, Lemma 9]:

$$\arg\min_{x \in \mathbb{R}^d} \langle g, x \rangle + \frac{c}{2}\|x\|_H^2 = -\frac{1}{c} H^{-1} g \quad \text{if } H \succ 0 \text{ and } c > 0.$$

Rearranging (2.42) yields $-\|g_k\|_{H_k^{-1}}^2 \leq 2\hat{\mu}(f_\star - f(x_k))$. Let $M_k := \frac{u_k u_k^\top}{u_k^\top H_k u_k}$. Then, from

$\hat{L}$-relative smoothness and [GKL19, Lemma 5],

$$f(x_{\mathrm{ED},k}) \leq f(x_k) - \frac{1}{2\hat{L}}\|g_k\|_{M_k}^2 = f(x_k) - \frac{1}{2\hat{L}}\frac{\langle u_k u_k^\top g_k, g_k\rangle}{u_k^\top H_k u_k} = f(x_k) - \frac{1}{2\hat{L}}\frac{(u_k^\top g_k)^2}{u_k^\top H_k u_k}. \quad (2.43)$$

Now let $\mathbb{E}_k[\cdot] := \mathbb{E}[\cdot|x_k]$ and take the conditional expectation of both sides of (2.43):

$$\begin{aligned}\mathbb{E}_k\left[f(x_{\mathrm{ED}})\right] &\leq f(x_k) - \frac{1}{2\hat{L}}\mathbb{E}_k\left[\frac{(u_k^\top g_k)^2}{u_k^\top H_k u_k}\right] \\ &= f(x_k) - \frac{\eta(g_k, H_k; \mathcal{D}_k)}{2\hat{L}}\|g_k\|_{H_k^{-1}}^2 \\ &\leq f(x_k) - \eta_k\frac{\hat{\mu}}{\hat{L}}(f(x_k) - f_\star)\end{aligned}$$

Subtracting $f_\star$ from both sides yields the desired result. □

Proof of Theorem 2.2.1. In this proof, for notational convenience let $d_0 = g_k^\top u_k$ for the first-order directional derivative, and $h_0 = u_k^\top H_k u_k$ for the second-order, and denote r_k by r. From the definition of $\hat{L}$-relative smoothness, how much we progress at each step can easily be described by a quadratic function $q(t)$:

$$f(x_k) - f(x_k + tu_k) \geq q(t) := -d_0 t - \frac{1}{2}\hat{L}h_0 t^2.$$

As in the exact derivatives case, the maximizer of q is $t_\star = -d_0/(\hat{L}h_0)$, with corresponding maximum $q(t_\star) = d_0^2/(2\hat{L}h_0) = \|g_k\|_{M_k}/(2\hat{L})$, where $M_k := \frac{u_k u_k^\top}{u_k^\top H_k u_k}$ as before. Recall that $x_{\mathrm{CARS},k} = x_k - d_r/(\hat{L}h_r)u_k$. Our goal is to show that the finite difference estimate $t_r := -d_r/(\hat{L}h_r)$ approximates $t_\star$ well enough so that $q(t_r) \geq q(t_\star)/2$. Observe that if

$$|t_r/t_\star - 1| \leq \sqrt{1-c} \iff |t_r - t_\star|^2 \leq (1-c)t_\star^2 \quad (2.44)$$

holds for some $0 < c < 1$, then by completing the square in $q(t)$:

$$q(t_r) = -\frac{\hat{L}h_0}{2}(t_r - t_\star)^2 + q(t_\star) \geq -(1-c)q(t_\star) + q(t_\star) = cq(t_\star).$$

Because we want to show $q(t_r) \geq q(t_\star)/2$, it suffices to show (2.44) holds for $c = 1/2$, *i.e.*,

$$\left|\frac{t_r}{t_\star} - 1\right| = \left|\frac{d_r/d_0}{h_r/h_0} - 1\right| \leq \sqrt{1 - \frac{1}{2}} = \frac{1}{\sqrt{2}}. \tag{2.45}$$

To prove (2.45), we further bound the left-hand side by the two separate (relative) finite difference errors. Let e_d and e_h be the absolute errors in estimating d_0 and h_0, respectively, *i.e.* $e_d = |d_0 - d_r|$ and $e_h = |h_0 - h_r|$. Then, when $e_h < h_0$, which will be shown shortly,

$$\left|\frac{d_r/d_0}{h_r/h_0} - 1\right| = \left|\frac{-\frac{d_0-d_r}{d_0} + \frac{h_0-h_r}{h_0}}{1 - \frac{h_0-h_r}{h_0}}\right| \leq \frac{e_d/|d_0| + e_h/h_0}{1 - e_h/h_0},$$

and thus, for (2.45) we only need to prove

$$\frac{e_d}{|d_0|} + \left(1 + \frac{1}{\sqrt{2}}\right)\frac{e_h}{h_0} \leq \frac{1}{\sqrt{2}}. \tag{2.46}$$

Now we bound e_d and e_h using Taylor's theorem and Assumption 1. Because we have

$$f(x_k \pm ru_k) = f(x_k) \pm rg_k^\top u_k + r^2 \int_0^1 (1-t)u_k^\top H(x_k \pm tru_k)u_k \, dt, \tag{2.47}$$

we get the following representation for the error of the first-order directional derivative:

$$\begin{aligned} d_r - d_0 &= \frac{f(x_k + ru_k) - f(x_k - ru_k)}{2r} - g_k^\top u_k \\ &= \frac{r}{2}\int_0^1 (1-t)u_k^\top \left[H(x_k + tru_k) - H(x_k - tru_k)\right] u_k \, dt. \end{aligned}$$

By Assumption 1, $\left|u_k^\top \left[H(x_k + tru_k) - H(x_k - tru_k)\right] u_k\right| \leq L_a(2tr)^a \|u_k\|^{a+2}$ and therefore,

$$e_d = |d_r - d_0| \leq 2^{a-1} L_a r^{a+1} \|u_k\|^{a+2} \int_0^1 (1-t)t^a \, dt = \left(\frac{r\|u_k\|}{C_{1,a}}\right)^{1+a} \frac{\|u_k\|}{2\sqrt{2}}. \tag{2.48}$$

Similarly, for the second-order directional derivative,

$$e_h = |h_r - h_0| \leq 2L_a r^a \|u_k\|^{a+2} \int_0^1 (1-t)t^a \, dt = \left(\frac{r\|u_k\|}{C_{2,a}}\right)^{a} \frac{\|u_k\|^2}{2\sqrt{2}+2} \tag{2.49}$$

We see that $r\|u_k\| \leq C = \min\{C_{1,a}(\gamma\sqrt{2\mu\varepsilon})^{1/(1+a)}, C_{2,a}\mu^{1/a}\}$ implies two separate bounds

$$e_d \leq \frac{\gamma\sqrt{\mu\varepsilon}\|u_k\|}{2} \overset{(a)}{\leq} \frac{|d_0|}{2\sqrt{2}} \quad \text{and} \quad e_h \leq \frac{\mu\|u_k\|^2}{2\sqrt{2}+2} \overset{(b)}{\leq} \frac{h_0}{2\sqrt{2}+2}, \tag{2.50}$$

where (a) holds assuming $\mathcal{A}_k$ occurs and (b) follows from strong convexity:

$$h_0 = u_k^\top H_k u_k \geq \mu. \tag{2.51}$$

As (2.50) implies (2.46) we have proved the theorem. □

We now are ready to prove the convergence of CARS (Algorithm 1).

Proof of Corollary 2.2.2. From strong convexity we have

$$f_\star - f(x) \geq \langle g(x), x_\star - x\rangle + \frac{\mu}{2}\|x_\star - x\|^2 \geq -\frac{1}{2\mu}\|g(x)\|^2,$$

for any $x \in \mathbb{R}^d$, where the second inequality comes from

$$\arg\min_{x\in\mathbb{R}^d} \langle g, x\rangle + \frac{c}{2}\|x\|^2 = -\frac{1}{c}g.$$

Thus $\|g(x)\|^2 \geq 2\mu(f(x) - f_\star)$. Taking expectation on both sides $\mathbb{E}[\|g(x_k)\|^2] \geq 2\mu(\mathbb{E}[f(x_k)] - f_\star)$.

If $\|g(x_k)\|^2 \leq 2\mu\varepsilon$ at the k-th step with $k \leq K$, then $f(x_K) - f_\star \leq \varepsilon$ as $f(x_k)$ is monotonically decreasing by definition (See line 9 of Algorithm 1.) Thus we need only consider the case where $\|g(x_k)\|^2 > 2\mu\varepsilon$ for all $k < K$; because if the expectation of $f(x_K)$ conditioned on this event is less than or equal to $f_\star + \varepsilon$, then the total expectation is also bounded by the same value.

The key of the proof is that $\mathcal{A}_k$ occurs with probability at least $p_\gamma > 0$. Indeed, we have $|u_k^\top g_k| \geq \gamma\|u_k\|\|g_k\|$ with probability at least p_γ, and since $\|g_k\| > \sqrt{2\mu\varepsilon}$,

$$\mathbb{P}[\mathcal{A}_k] \geq \mathbb{P}\left[|u_k^\top g_k| \geq \gamma\|u_k\|\|g_k\| \geq \gamma\|u_k\|\sqrt{2\mu\varepsilon}\right] \geq p_\gamma.$$

If $\mathcal{A}_k$ occurs then by Theorem 2.2.1, we get

$$\mathbb{E}[f(x_{k+1})|\mathcal{A}_k] - f_\star \leq \left(1 - \eta_{\mathcal{D}}\frac{\hat{\mu}}{2\hat{L}}\right)(f(x_k) - f_\star).$$

If $\mathcal{A}_k$ does not occur then, as CARS is non-increasing, $f(x_{k+1}) \leq f(x_k)$. Thus

$$\begin{aligned}
\mathbb{E}\left[f(x_{k+1}) \mid x_k\right] - f_\star &= \mathbb{E}[f(x_{k+1}) - f_\star|\mathcal{A}_k]\mathbb{P}[\mathcal{A}_k] + \mathbb{E}[f(x_{k+1}) - f_\star|\mathcal{A}_k^c]\mathbb{P}[\mathcal{A}_k^c] \\
&\leq \left(1 - \eta_{\mathcal{D}}\frac{\hat{\mu}}{2\hat{L}}\right)(f(x_k) - f_\star)\mathbb{P}[\mathcal{A}_k] + (f(x_k) - f_\star)\left(1 - \mathbb{P}[\mathcal{A}_k]\right) \\
&= \left(1 - \eta_{\mathcal{D}}\mathbb{P}[\mathcal{A}_k]\frac{\hat{\mu}}{2\hat{L}}\right)(f(x_k) - f_\star) \\
&\leq \left(1 - \eta_{\mathcal{D}}p_\gamma\frac{\hat{\mu}}{2\hat{L}}\right)(f(x_k) - f_\star) \\
\Rightarrow \mathbb{E}[f(x_{k+1})] - f_\star &\leq \left(1 - \eta_{\mathcal{D}}p_\gamma\frac{\hat{\mu}}{2\hat{L}}\right)^{k+1}(f(x_0) - f_\star),
\end{aligned}$$

whence solving for K in

$$\left(1 - \eta_{\mathcal{D}}p_\gamma\frac{\hat{\mu}}{2\hat{L}}\right)^{K}(f(x_0) - f_\star) \leq \varepsilon \tag{2.52}$$

completes the proof. □

2.6.2 Proofs for Results in Section 2.3

Recall that:

$$P(\alpha; d, h) := d\alpha + \frac{1}{2}h\alpha^2 + \frac{M}{6}|\alpha|^3$$

(we write $P(\alpha)$ in place of $P(\alpha; d, h)$ when d and h are clear from context.) Define the map $\phi : \mathbb{R} \times \mathbb{R}_{\geq 0} \to \mathbb{R}$:

$$\phi(d, h) := \arg\min_{\alpha} P(\alpha; d, h).$$

Note that not only $h_0 \geq 0$, but also $h_{r_k} \geq 0$ due to the convexity of f:

$$h_{r_k}(x_k; u_k) = \frac{2}{r_k^2}\left(\frac{f(x_k + r_k u_k) + f(x_k - r_k u_k)}{2} - f(x_k)\right) \geq 0.$$

Then $\hat{\alpha}_k = \phi(d_0, h_0)$ and $\alpha_k^{\pm} = \phi(\pm d_{r_k}, h_{r_k})$ by their definition. Along the way, we have useful identities for ϕ:

$$\phi(d, h) = \frac{\operatorname{sign}(d)}{M}\left(h - \sqrt{h^2 + 2M|d|}\right) = \frac{-2d}{h + \sqrt{h^2 + 2M|d|}}, \tag{2.53}$$

and

$$\frac{M}{2}|\alpha_{\min}|\alpha_{\min} = -d - h\alpha_{\min}. \tag{2.54}$$

Note that (2.53) shows that ϕ is well-defined. We first describe the perturbation of ϕ, and how P behaves near its minimum.

Lemma 2.6.3 (Perturbation of ϕ)**.** *Let $d, d' \in \mathbb{R}$ have the same sign and $h, h' \geq 0$. Defining $S = \sqrt{h^2 + 2M|d|}$ and $S' = \sqrt{(h')^2 + 2M|d'|}$,*

$$|\phi(d, h) - \phi(d', h')| \leq \frac{|h - h'|}{M} + \frac{2|d - d'|}{S + S'}. \tag{2.55}$$

Proof. Because d and d' have the same sign, from (2.53), we obtain that $\phi(d, h)$ and $\phi(d', h')$

have the same sign and so $|\phi(d,h) - \phi(d',h')| = \frac{1}{M}|S - S' - (h - h')|$, whence

$$\begin{aligned}
|\phi(d,h) - \phi(d',h')| &= \frac{1}{M}\left|(S - S')\,\frac{S + S'}{S + S'} - (h - h')\right| = \frac{1}{M}\left|\frac{S^2 - (S')^2}{S + S'} - (h - h')\right| \\
&= \frac{1}{M}\left|\frac{(h - h')(h + h')}{S + S'} + \frac{2M(|d| - |d'|)}{S + S'} - (h - h')\right| \\
&\leq \frac{1}{M}\left(1 - \frac{h + h'}{S + S'}\right)|h - h'| + \frac{2|d - d'|}{S + S'} \quad \leq \frac{|h - h'|}{M} + \frac{2|d - d'|}{S + S'},
\end{aligned}$$

where the last inequality comes from that $0 \leq h + h' \leq S + S'$. □

We now analyze the effect of perturbations to $\alpha_{\min}$ on $P(\alpha)$, under the assumption that the perturbed value of α has the same sign as $\alpha_{\min}$.

Lemma 2.6.4 (Perturbation of $P(\alpha)$ near minimum)**.** *Let $d \in \mathbb{R}$ and $h \geq 0$. Define $\alpha_{\min} = \phi(d,h)$, and let $\alpha' \in \mathbb{R}$ have $\operatorname{sign}(\alpha') = \operatorname{sign}(\alpha_{\min})$. Then*

$$0 \leq P(\alpha'; d, h) - P(\alpha_{\min}; d, h) \leq \frac{1}{2}(\alpha_{\min} - \alpha')^2(h + M|\alpha_{\min}| + \frac{M}{3}|\alpha_{\min} - \alpha'|). \tag{2.56}$$

Proof. Let $\sigma = \operatorname{sign}(\alpha_{\min}) = \operatorname{sign}(\alpha')$. We write $P(\alpha_{\min})$, resp. $P(\alpha)$, for $P(\alpha_{\min}; d, h)$, resp. $P(\alpha'; d, h)$. Then,

$$\begin{aligned}
&P(\alpha') - P(\alpha_{\min}) \\
&= d(\alpha' - \alpha_{\min}) + \frac{h}{2}(\alpha' - \alpha_{\min})(\alpha' + \alpha_{\min}) + \frac{\sigma M}{6}(\alpha' - \alpha_{\min})((\alpha')^2 + \alpha_{\min}^2 + \alpha_{\min}\alpha') \\
&= (\alpha' - \alpha_{\min})\left(d + \frac{h}{2}(\alpha' + \alpha_{\min}) + \frac{\sigma M}{6}((\alpha')^2 + \alpha_{\min}^2 + \alpha_{\min}\alpha')\right).
\end{aligned}$$

Using (2.54), we get

$$\begin{aligned} P(\alpha') - P(\alpha_{\min}) &= (\alpha' - \alpha_{\min}) \left(\frac{h}{2}(\alpha' - \alpha_{\min}) + \frac{\sigma M}{6}((\alpha')^2 - 2\alpha_{\min}^2 + \alpha_{\min}\alpha') \right) \\ &= \frac{1}{2}(\alpha' - \alpha_{\min})^2 \left(h + \frac{M}{3}|\alpha' + 2\alpha_{\min}| \right) \\ &\leq \frac{1}{2}(\alpha' - \alpha_{\min})^2 \left(h + M|\alpha_{\min}| + \frac{M}{3}|\alpha' - \alpha_{\min}| \right). \end{aligned}$$

Noting that $P(\alpha') - P(\alpha_{\min}) \geq 0$ as $\alpha_{\min}$ minimizes $P(\alpha)$ we obtain the desired statement. □

From (2.53) we see that if $\operatorname{sign}(d_{r_k}) = \operatorname{sign}(d_0)$ then $\operatorname{sign}(\hat{\alpha}_k) = \operatorname{sign}(\alpha_k^+)$, whence we may use the perturbation bounds of Lemmas 2.6.3 and 2.6.4. If $\operatorname{sign}(d_{r_k}) = -\operatorname{sign}(d_0)$ then $\operatorname{sign}(\hat{\alpha}_k) = \operatorname{sign}(\alpha_k^-)$ and the conclusions of Lemmas 2.6.3 and 2.6.4 still apply. We conclude that at least one of α_k^+ and α_k^- is a good approximation for $\hat{\alpha}_k$, and formalize this as Lemma 2.3.1.

Proof of Lemma 2.3.1. First, assume that $\operatorname{sign}(d_0) = \operatorname{sign}(d_{r_k})$, so $\operatorname{sign}(\hat{\alpha}_k) = \operatorname{sign}(\alpha_k^+)$ by (2.53). Thus, by Lemma 2.6.4,

$$|P(\hat{\alpha}_k) - P(\alpha_k^+)| \leq \frac{1}{2}(\alpha_k^+ - \hat{\alpha}_k)^2 \left(h_0 + M|\hat{\alpha_k}| + \frac{M}{3}|\alpha_k^+ - \hat{\alpha}_k| \right). \tag{2.57}$$

Since $h_0 = u_k^\top H_k u_k \leq L$, it only remains to find appropriate bounds for $|\alpha_k^+ - \hat{\alpha}_k|$ and $\hat{\alpha}_k$. For notational convenience, define $S_r := \sqrt{h_r^2 + 2M|d_r|}$ for $r \geq 0$. As f is convex we know that $|d_0| \leq \|g_k\| \leq \sqrt{2L(f(x_k) - f_\star)}$, see [Ber97, Prop. B.3] and so

$$\begin{aligned} M|\hat{\alpha}_k| &\overset{(2.53)}{=} |S_0 - h_0| \leq \sqrt{S_0^2 - h_0^2} = \sqrt{2M|d_0|} \leq \sqrt{2M\|g_k\|} \\ &\leq \sqrt{2M\sqrt{2L(f(x_k) - f_\star)}} = \sqrt{\frac{2}{R^3}(MR^3)\sqrt{\frac{2}{R^2}(LR^2)(f(x_k) - f_\star)}} \leq \frac{2^{3/4}B}{R^2}, \end{aligned}$$

using the definition of $B = \max(LR^2, MR^3, f(x_k) - f_\star)$. Defining the finite difference errors

$e_k^d = d_{r_k} - d_0$ and $e_k^h = h_{r_k} - h_0$, Lemma 2.6.3 implies

$$|\hat{\alpha}_k - \alpha_k^+| \leq \frac{|e_k^h|}{M} + \frac{2|e_k^d|}{S_0 + S_{r_k}}. \tag{2.58}$$

As $\|u_k\| = 1$ and H is assumed Lipschitz continuous (*i.e.* $a = 1$), from (2.49), we have $|e_k^h| \leq \frac{Mr_k}{3}$ and the first term on the right-hand side of (2.58) is bounded by $\frac{r_k}{3}$. Appealing to (2.48) we obtain $|e_k^d| \leq \frac{Mr_k^2}{6}$. We use this and the fact that $\text{sign}(d_0) = \text{sign}(d_k)$ to bound the second term on the right-hand side of (2.58):

$$\begin{aligned}\frac{2|e_k^d|}{S_0 + S_{r_k}} = \frac{2|d_k - d_0|}{S_0 + S_{r_k}} &\leq \frac{2\left(\sqrt{|d_0|} + \sqrt{|d_k|}\right)\left|\sqrt{|d_0|} - \sqrt{|d_k|}\right|}{\sqrt{2M|d_0|} + \sqrt{2M|d_k|}} \\ &= \frac{2\left|\sqrt{|d_0|} - \sqrt{|d_k|}\right|}{\sqrt{2M}} \leq \frac{2\sqrt{|d_0 - d_k|}}{\sqrt{2M}} \leq 2\sqrt{\frac{1}{2M}\frac{Mr_k^2}{6}} = \frac{r_k}{\sqrt{3}}.\end{aligned}$$

This provides a nice bound independent of L, M, and R; $|\hat{\alpha}_k - \hat{\alpha}_k| \leq (1/3 + 1/\sqrt{3})r_k < r_k$. Combining everything with (2.57), we get

$$\begin{aligned}\left|P_0(\hat{\alpha}_k) - P_0(\alpha_k^+)\right| < \frac{1}{2}r_k^2\left(L + \frac{2^{3/4}B}{R^2} + \frac{M}{3}r_k\right) &\leq \frac{1}{2}r_k^2\left(\frac{B}{R^2} + \frac{2^{3/4}B}{R^2} + \frac{B}{3R^2}\right) \\ \leq \frac{Br_k^2}{R^2}\left(\frac{2}{3} + \frac{1}{2^{1/4}}\right) &\leq \frac{2B}{R^2}r_k^2.\end{aligned}$$

If $\text{sign}(d_0) = -\text{sign}(d_{r_k})$ then $\text{sign}(\hat{\alpha}_k) = \text{sign}(\alpha_k^-)$, again by (2.53). Lemma 2.6.3 and Lemma 2.6.4 now yield

$$\begin{aligned}&|\hat{\alpha}_k - \alpha_k^-| \leq \frac{|e_k^h|}{M} + \frac{2|d_0 - (-d_{r_k})|}{S_0 + S_{r_k}} \\ &|P(\hat{\alpha}_k) - P(\alpha_k^-)| \leq \frac{1}{2}(\alpha_k^- - \hat{\alpha}_k)^2\left(h_0 + M|\hat{\alpha_k}| + \frac{M}{3}|\alpha_k^- - \hat{\alpha}_k|\right).\end{aligned} \tag{2.59}$$

The first term in (2.59) can be bounded as before. Because $|d_0 + d_{r_k}| \leq |d_0 - d_{r_k}| \leq |e_k^d|$ as d_0 and d_{r_k} have opposite signs, the second term in (2.59) is bounded by $r_k/\sqrt{3}$ as before.

Following the proof of the $\text{sign}(d_0) = \text{sign}(d_{r_k})$ case we conclude that,

$$\left|P_0(\hat{\alpha}_k) - P_0(\alpha_k^-)\right| \leq \frac{2B}{R^2} r_k^2,$$

thus proving the theorem. □

Proof of Theorem 2.3.2. First, fix $u_k \in \mathbb{R}^d$ drawn from $\mathcal{D}_k$. Then, for $\sigma = -\text{sign}(d_0(x_k; u_k))$ and any $z \in \mathbb{R}^d$,

$$\begin{aligned}
&f(x_{k+1}) - f(x_k) \\
&\leq f(x_k + \alpha_k^\sigma u) - f(x_k) \quad \leq P(\alpha_k^\sigma; d_0(x_k; u_k), h_0(x_k; u_k)) && \text{(Eq. (2.23))} \\
&\leq P(\hat{\alpha}_k; d_0, h_0) + \frac{2B}{R^2} r_k^2 && \text{(Lemma 2.3.1)} \\
&\leq P(u_k^\top z; d_0, h_0) + \frac{2B}{R^2} r_k^2 && \text{(minimality of } \hat{\alpha}_k\text{)} \\
&= (z^\top u_k)(u_k^\top g_k) + \frac{1}{2}(z^\top u_k)(u_k^\top H_k u_k)(u_k^\top z) + \frac{M}{6}|u_k^\top z|^3 + \frac{2B}{R^2} r_k^2
\end{aligned}$$

holds. Now taking the expectation and using the isotropy condition:

$$\mathbb{E}\left[f(x_{k+1}) \mid x_k\right] - f(x_k) \leq \frac{1}{d} z^\top g_k + \frac{1}{2} z^\top \mathbb{E}\left[u_k u_k^\top H_k u_k u_k^\top\right] z + \frac{M}{6} \mathbb{E}\left[|u_k^\top z|^3\right] + \frac{2B}{R^2} r_k^2.$$

Note that the expectations above satisfy $\frac{1}{2} z^\top \mathbb{E}\left[u_k u_k^\top H_k u_k u_k^\top\right] z \leq \frac{1}{2} z^\top \mathbb{E}\left[L u_k u_k^\top\right] z = \frac{L}{2d}\|z\|^2$ and $\mathbb{E}\left[|u_k^\top z|^3\right] \leq \mathbb{E}\left[|u_k^\top z|^2\right] \|z\| = \frac{1}{d}\|z\|^3$, respectively. Therefore,

$$\mathbb{E}\left[f(x_{k+1}) \mid x_k\right] - f(x_k) \leq \frac{1}{d} z^\top g_k + \frac{L}{2d}\|z\|^2 + \frac{M}{6d}\|z\|^3 + \frac{2B}{R^2} r_k^2. \tag{2.60}$$

Finally, using convexity of f, namely $f(x_k + z) - f(x_k) \geq z^\top g_k$, we obtain (2.26). □

Proof of Theorem 2.3.3. Let $\delta(x)$ denote the optimality gap $f(x) - f_\star$, and $\delta_k := \mathbb{E}[\delta(x_k)]$. Since Algorithm 2 has non-increasing δ_k, we may assume $\delta_0 > \varepsilon$. Note that δ is convex. Letting $x_\star$ be any fixed minimizer (*i.e.* $f(x_\star) = f_\star$), we note that $\delta(x_\star) = 0$. For any

$t_k \in (0,1)$, setting $z = t_k(x_\star - x_k)$ in Theorem 2.3.2 and defining $\Delta_k = \|x_\star - x_k\|$ yields

$$\begin{aligned}&\mathbb{E}\left[f(x_{k+1}) \,|\, x_k\right] - f_\star \\ &\leq (1-\frac{1}{d})f(x_k) + \frac{1}{d}f((1-t_k)x_k + t_k x_\star) - f_\star + \frac{L}{2d}t_k^2\Delta_k^2 + \frac{M}{6d}t_k^3\Delta_k^3 + \frac{2B}{R^2}r_k^2\end{aligned}$$

and

$$\delta_{k+1} \leq (1-\frac{1}{d})f(x_k) + \frac{1-t_k}{d}f(x_k) + \frac{t_k}{d}f_\star - f_\star + \frac{L}{2d}t_k^2\Delta_k^2 + \frac{M}{6d}t_k^3\Delta_k^3 + \frac{2B}{R^2}r_k^2 \tag{2.61}$$

$$\delta_{k+1} \leq (1-\frac{1}{d}+\frac{1}{d}-\frac{t_k}{d})f(x_k) - (1-\frac{t_k}{d})f_\star + \frac{L}{2d}t_k^2\Delta_k^2 + \frac{M}{6d}t_k^3\Delta_k^3 + \frac{2B}{R^2}r_k^2 \tag{2.62}$$

$$\delta_{k+1} \leq (1-\frac{t_k}{d})\delta_k + \frac{L}{2d}t_k^2\Delta_k^2 + \frac{M}{6d}t_k^3\Delta_k^3 + \frac{2B}{R^2}r_k^2, \tag{2.63}$$

where in (2.61) we use the convexity of f, in (2.62) we use $f(x_\star) = f_\star$, and in (2.63) we use the definition of δ_k. We adopt an auxiliary sequence $\{\beta_k\}$ to make (2.63) telescoping. Let $s > 1$, and define $\gamma_k = k^s$ and $\beta_k = \beta_0 + \sum_{j=1}^k \gamma_j$ with $\beta_0 = s^s d^{s+1}/(s+1)$, then $t_k = d\frac{\gamma_{k+1}}{\beta_{k+1}} \in (0,1)$, and $1 - \frac{t_k}{d} = \frac{\beta_k}{\beta_{k+1}}$. We further note that:

$$\frac{k^{s+1}}{s+1} \leq \beta_0 + \int_1^k \frac{1}{x^s}dx \leq \beta_k \leq \beta_0 + \int_2^{k+1}\frac{1}{x^s}dx = \beta_0 + \frac{(k+1)^{s+1}}{s+1} \tag{2.64}$$

Then by multiplying β_{k+1} on both sides of (2.63), we get

$$\beta_{k+1}\delta_{k+1} \leq \beta_k\delta_k + \frac{Ld}{2}\frac{\gamma_{k+1}^2}{\beta_{k+1}}\Delta_k^2 + \frac{Md^2}{6}\frac{\gamma_{k+1}^3}{\beta_{k+1}^2}\Delta_k^3 + \frac{2B}{R^2}\beta_{k+1}r_k^2,$$

and summing up from $k = 0$ to $K-1$, we have

$$\delta_K \leq \frac{\beta_0}{\beta_K}\delta_0 + \frac{Ld}{2\beta_K}\sum_{k=1}^K \frac{\gamma_k^2}{\beta_k}\Delta_{k-1}^2 + \frac{Md^2}{6\beta_K}\sum_{k=1}^K\frac{\gamma_k^3}{\beta_k^2}\Delta_{k-1}^3 + \frac{2B}{R^2\beta_K}\sum_{k=1}^K\beta_k r_{k-1}^2. \tag{2.65}$$

First, $\frac{\beta_0}{\beta_K} \leq \frac{\beta_0}{\beta_K - \beta_0} \leq \frac{s^s}{(K/d)^{s+1}}$. Because the sequence $f(x_k)$ is non-increasing, $x_k \in \mathcal{Q}$ for all

$k \geq 0$ and so $\Delta_k \leq \mathcal{R}$ (see Definition 3). Using $(1+\frac{1}{K})^s \leq e^{s/K}$,

$$\begin{aligned}\frac{1}{\beta_K}\sum_{k=1}^{K}\frac{\gamma_k^2}{\beta_k}\Delta_{k-1}^2 &\overset{(2.64)}{\leq} \frac{\mathcal{R}^2(s+1)^2}{K^{s+1}}\sum_{k=1}^{K}\frac{k^{2s}}{k^{s+1}} &&= \frac{\mathcal{R}^2(s+1)^2}{K^{s+1}}\sum_{k=1}^{K}k^{s-1}\\ &\leq \frac{\mathcal{R}^2(s+1)^2}{K^{s+1}}\frac{(K+1)^s}{s} &&\leq \frac{\mathcal{R}^2 e^{s/K}(s+1)^2}{sK}\end{aligned}$$

and

$$\begin{aligned}\frac{1}{\beta_K}\sum_{k=1}^{K}\frac{\gamma_k^3}{\beta_k^2}\Delta_{k-1}^3 &\overset{(2.64)}{\leq} \frac{\mathcal{R}^3(s+1)^3}{K^{s+1}}\sum_{k=1}^{K}\frac{k^{3s}}{k^{2s+2}} &&= \frac{\mathcal{R}^3(s+1)^3}{K^{s+1}}\sum_{k=1}^{K}k^{s-2}\\ &\leq \frac{\mathcal{R}^3(s+1)^3}{K^{s+1}}\frac{(K+1)^{s-1}}{s-1} &&\leq \frac{\mathcal{R}^3 e^{(s-1)/K}(s+1)^3}{(s-1)K^2}.\end{aligned}$$

Lastly, the error due to the finite difference is controlled by the sampling radius:

$$\begin{aligned}\frac{2B}{R^2\beta_K}\sum_{k=1}^{K}\beta_k r_{k-1}^2 &\overset{(2.64)}{\leq} \frac{2(s+1)B\varepsilon\rho^2}{R^2K^{s+1}}\sum_{k=1}^{K}\frac{(k+1)^s}{s+1} + \frac{2B\varepsilon\rho^2\beta_0}{R^2\beta_K}\sum_{k=1}^{K}\frac{1}{(k+1)}\\ &\leq \frac{\varepsilon e^{2(s+1)/K}}{s+1} + \frac{\varepsilon\beta_0\log(K+2)}{\beta_K}.\end{aligned}$$

Combining the above with $\varepsilon < \delta_0$ we get

$$\begin{aligned}\delta_K \leq{}& \frac{s^s\delta_0(1+\log(K+2))}{(K/d)^{s+1}} + \frac{e^{s/K}(s+1)^2L\mathcal{R}^2}{2s(K/d)}\\ &+ \frac{e^{(s-1)/K}(s+1)^3M\mathcal{R}^3}{6(s-1)(K/d)^2} + \frac{e^{2(s+1)/K}}{s+1}\varepsilon.\end{aligned} \tag{2.66}$$

When $K > s$, bounding the first three term in (2.66) by $\frac{s-1}{6(s+1)}$, and the last term by $\frac{s+3}{2(s+1)}$ gives the sufficient conditions on K:

$$\frac{K}{d} \geq \max\left\{\frac{3(s+1)^3e}{s(s-1)}\frac{L\mathcal{R}^2}{\varepsilon}, \frac{(s+1)^2\sqrt{e}}{(s-1)}\sqrt{\frac{M\mathcal{R}^3}{\varepsilon}}, s\left(\frac{6(s+1)}{s-1}\right)^{1/s}\left(\frac{\delta_0}{\varepsilon}\right)^{1/s}\right\}$$

and $K \geq \frac{2(s+1)}{\log(1+s/2)}$, respectively. These immediately give (2.28). □

2.6.3 Proofs for Results in Section 2.5

Proof of Theorem 2.5.1. We first show (2.37) for a diagonal matrix $A = \mathrm{diag}(\lambda_1, \cdots, \lambda_d)$, and then extend the result to general SPD matrices. Let $u = (u_1, \cdots, u_d)$ be a Gaussian vector and $q(u)$ denote the matrix in the expectation:

$$q(u) = \frac{\|u\|^d}{(u^\top A u)^p} u u^\top.$$

Then the (i,j)-th component $(q(u))_{ij}$ is $\|u\|^d u_i u_j/(u^\top A u)^p$. To see the off-diagonal entries are zero, let $\tilde{u}$ denote the vector obtained by flipping the sign of u_i, $(u_1, \cdots, -u_i, \cdots, u_d)$. Note that $\tilde{u}$ is also a Gaussian vector. Then for $i \neq j$, $\mathbb{E}[(q(u))_{ij}] = 0$ follows from $\mathbb{E}[(q(u))_{ij}] = \mathbb{E}[(q(\tilde{u}))_{ij}] = -\mathbb{E}[(q(u))_{ij}]$.

For the diagonal elements, it suffices to prove

$$\mathbb{E}[(q(u))_{ii}] = \mathbb{E}\left[\frac{\|u\|^d u_i^2}{(\sum_{j=1}^d \lambda_j u_j^2)^p}\right] = \frac{1}{d\lambda_i \sqrt{\prod_{j=1}^d \lambda_j}} = \frac{(A^{-1})_{ii}}{d\sqrt{\det(A)}}. \tag{2.67}$$

We compute the expectation directly as follows, using that u_i's are *i.i.d.* with the density $f(x) = \exp(-x^2/2)/\sqrt{2\pi}$:

$$\begin{aligned}\mathbb{E}[(q(u))_{ii}] &= \mathbb{E}\left[\frac{\|u\|^d u_i^2}{(\sum_{j=1}^d \lambda_j u_j^2)^p}\right] \\ &= \int_{\mathbb{R}^d} \frac{u_i^2}{\left(\sum_{j=1}^d \lambda_j u_j^2\right)^p} \left(\sum_{j=1}^d u_j^2\right)^{p-1} f(u_1)\cdots f(u_d)\, du,\end{aligned}$$

and with the change of variables $v_j = \sqrt{\lambda_j} u_j$ for $j = 1, \cdots, d$, it computes

$$\frac{(2\pi)^{-d/2}}{\lambda_i \sqrt{\prod_{j=1}^d \lambda_j}} \int_{\mathbb{R}^d} \frac{v_i^2}{\left(\sum_{j=1}^d v_j^2\right)^p} \left(\sum_{j=1}^d \frac{v_j^2}{\lambda_j}\right)^{p-1} \exp(-\sum_{j=1}^d \frac{v_j^2}{2\lambda_j})\, dv. \tag{2.68}$$

Thus we only need to prove the integral equals $d^{-1}(2\pi)^{d/2}$. By rearranging (interchanging v_i and v_1) and using a spherical coordinate,

$$\begin{aligned} v_1 &= r\cos(\phi_1) \\ v_j &= r\sin(\phi_1)\cdots\sin(\phi_{j-1})\cos(\phi_j), \qquad 2 \le j \le d-1 \\ v_d &= r\sin(\phi_1)\cdots\sin(\phi_{d-1}), \end{aligned}$$

we have $\left(\sum_{j=1}^{d} v_j^2\right)^p = r^{2p} = r^{d+2}$ and

$$dv_1 \cdots dv_d = r^{d-1}\prod_{j=1}^{d-2}\sin^{d-1-j}(\phi_j)\, dr\, d\phi_1 \cdots d\phi_{d-1}.$$

Therefore, the integral in (2.68) becomes

$$\int_0^{2\pi}\int_0^{\pi}\cdots\int_0^{\pi}\int_0^{\infty} r^{d-1}\cos^2(\phi_1)\alpha^d\exp(-\frac{r^2\alpha^2}{2})\times \prod_{j=1}^{d-2}\sin^{d-1-j}(\phi_j)\; dr\, d\phi_1 \cdots d\phi_{d-1},$$

where

$$\begin{aligned} \alpha^2 &:= r^{-2}\sum_{j=1}^{d}\frac{v_j^2}{\lambda_j} \\ &= \frac{\cos^2(\phi_1)}{\lambda_1} + \frac{\sin^2(\phi_1)\cos^2(\phi_2)}{\lambda_2} + \cdots + \frac{\sin^2(\phi_1)\cdots\sin^2(\phi_{d-1})}{\lambda_d}. \end{aligned}$$

Notice that the integral is the product of two quantities (A) and (B), where

$$(A) := \int_0^{\infty} r^{d-1}\alpha^d\exp(-\frac{r^2\alpha^2}{2})\, dr,$$

which computes $(A) = 2^{d/2-1}\Gamma(n/2)$, and

$$(B) := \int_0^{2\pi} \int_0^{\pi} \cdots \int_0^{\pi} \cos^2(\phi_1) \prod_{j=1}^{d-2} \sin^{d-1-j}(\phi_j)\ d\phi_1 \cdots d\phi_{d-1}.$$

For the notational simplicity, let P_m denote the definite integral $\int_0^{\pi} \sin^m(\theta)d\theta$. Then

$$(B) = 2\pi P_1 P_2 \cdots P_{d-3}(P_d - P_{d-2}),$$

from $\cos^2(\phi_1) = 1 - \sin^2(\phi_1)$. Also note that from integrating by parts,

$$P_d = \frac{d-1}{d} P_{d-2}$$

and thus $(B) = 2\pi d^{-1} P_1 P_2 \cdots P_{d-2}$. But the product $2\pi P_1 \cdots P_{d-2}$ is exactly the surface area of the unit sphere, $2\pi^{d/2}\Gamma(d/2)^{-1}$. Therefore, the integral in (2.68) is precisely $d^{-1}(2\pi)^{d/2}$ and this proves the proposition for diagonal A.

Finally, let A be a general SPD matrix. Then A admits the eigendecomposition $A = PDP^\top$, where $PP^\top = P^\top P = I_d$, and D is diagonal. Then $v = P^\top u$ still follows the standard normal distribution. Therefore,

$$\mathbb{E}[q(u)] = P^\top \mathbb{E}\left[\frac{\|v\|^d}{(v^\top D v)^p} v v^\top\right] P = \frac{P D^{-1} P^\top}{d\sqrt{\det(D)}} = \frac{A^{-1}}{d\sqrt{\det(A)}}.$$

This finishes the proof. □

2.7 Experimental Results

For a detailed description of all experimental settings and hyperparameters, see Appendix A.1. The code for all the experiments can be found online at `https://github.com/bumsu-kim/CARS`.

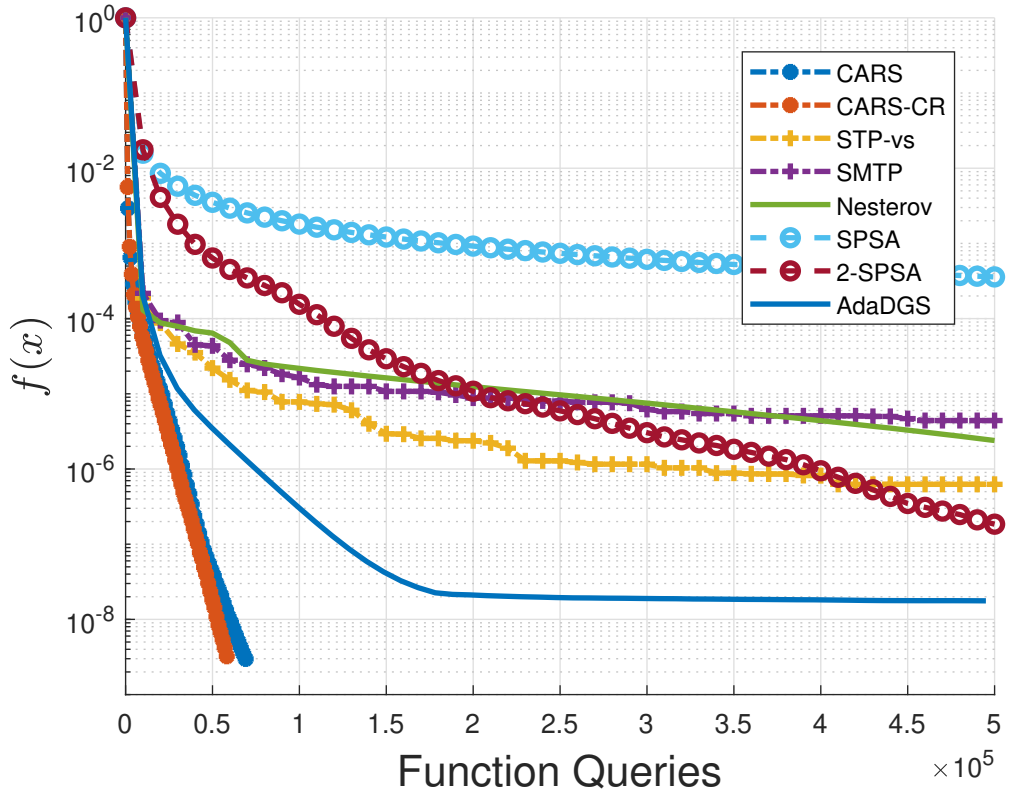

Figure 2.1: Performance of each algorithm on a convex quartic function $f(x) = 0.1\sum_{i=1}^{d} x_i^4 + \frac{1}{2}x^\top Ax + 0.01\|x\|^2$, where $A = G^\top G$ with $G_{ij} \overset{i.i.d}{\sim} \mathcal{N}(0,1)$. The problem dimension $d = 30$.

2.7.1 Convex Functions

We compared the performance of CARS and CARS-CR to STP [BGR20], SMTP [GBS19], Nesterov-Spokoiny [NS17], SPSA [Spa92], 2SPSA [Spa00], and AdaDGS [TZ20] on the following convex quartic function:

$$f(x) = \alpha \sum_{i=1}^{d} x_i^4 + \frac{1}{2}x^\top Ax + \beta\|x\|^2,$$

where $\alpha, \beta > 0$ and $A = G^\top G$ with $G_{ij} \overset{i.i.d}{\sim} \mathcal{N}(0,1)$ for $i, j = 1, 2, \cdots, d$. We show in Figure 2.1 the objective function value versus the number of function queries.

2.7.2 Benchmark Problem Sets with Non-Convex Functions

The test results in this section are presented in the form of performance profiles [DM02], which is a commonly used tool for comparing the performance of multiple algorithms over a suite of test problems. Performance profiles tend to be more informative than single-dimensional summaries (*e.g.* average number of iterations required to solve a problem). Formally, consider fixed sets of problems $\mathcal{P}$ and algorithms $\mathcal{S}$. For each $p \in \mathcal{P}$ and $s \in \mathcal{S}$ the *performance ratio* $r_{p,s}$ is defined by

$$r_{p,s} = \frac{t_{p,s}}{\min_{s' \in \mathcal{S}} t_{p,s'}},$$

where $t_{p,s}$ is the number of function queries required for s to solve p. This is the relative performance of s on p compared to the best algorithm in $\mathcal{S}$ for p. The *performance profile* of s, $\rho_s : [1, \infty) \to [0, 1]$ is defined as

$$\rho_s(\tau) = \frac{|\{p \in \mathcal{P} : r_{p,s} \leq \tau\}|}{|\mathcal{P}|}.$$

Therefore, $\rho_s(1)$ is the fraction of problems for which s performs the best, while $\rho_s(\tau)$ for large τ measures the robustness of s. For all τ, a *higher value of $\rho_s(\tau)$ is better*. We use a log-scale on the horizontal access when plotting $\rho_s(\tau)$.

Moré-Garbow-Hillstrom Problems. We tested the same set of algorithms using the well-known non-convex Moré-Garbow-Hillstrom 34 test problems [MGH81]. For each target accuracy ε, a problem is considered solved when we have $f(x_k) - f_\star \leq \varepsilon(f(x_0) - f_\star)$ within the budget of 20,000 queries. We used the recommended starting point x_0 as in [MGH81] for all the tested algorithm, and repeated each test 10 times. The results are presented in Figure 2.2.

CUTEst Problems. We further assessed the performance of CARS and CARS-CR to the same suite of algorithms on the CUTEst [GOT15] problem set, which contains various convex and non-convex problems. As before, we compared the methods using performance profiles

Algorithm	Success Rate (%)	Median Queries	Average Queries
ZOO*	93.95	11,700	11,804
PGD-NES*	88.39	2,450	4,584
ZOHA-Gauss*	91.69	1,400	2,586
ZOHA-Diag*	91.06	1,656	3,233
STP	53.64	2,193	3,141
SMTP	65.68	1,415	2,250
Nesterov	67.72	1,105	2,044
Square Attack	**98.21**	1,060	1,297
CARS (Square)	97.09	**717**	**1,169**

Table 2.2: Comparison of success rates, and median and average function queries for the successful black-box adversarial attacks on MNIST with ℓ_∞-perturbation bound 0.2. CARS, equipped with the Square Attack's distribution, shows the best performance in successful attacks, while reaching the second best success rate. The results marked with * are cited from [YHF18].

for the 146 problems with dimension less than or equal to 50. The query budget for each problem was set to be $20,000$ times the problem dimension. The target accuracies were again set to $\varepsilon(f(x_0) - f_\star)$. The results are reported in Figure 2.2.

2.7.3 Problems with Highly Oscillatory Noise

We also evaluated the performance of the algorithms, including CARS-NQ, on the same set of problems, but this time with additional highly oscillatory noise. The results are depicted in Figure 2.3. The experiment notably showcases the efficacy of an increased sampling radius for CARS-NQ.

2.7.4 Black-box Adversarial Attacks

Suppose $\mathcal{N}$ is an image classifier. The problem of generating small perturbations x that, when added to a natural image x_{nat}, fool the classifier (*i.e.* $\mathcal{N}(x_{\text{nat}} + x) \neq \mathcal{N}(x_{\text{nat}})$) is known as finding an *adversarial attack* [GSS14]. As described in [CZS17], when no access to the internal workings of the classifier is available, this problem becomes a black-box, or derivative-free,

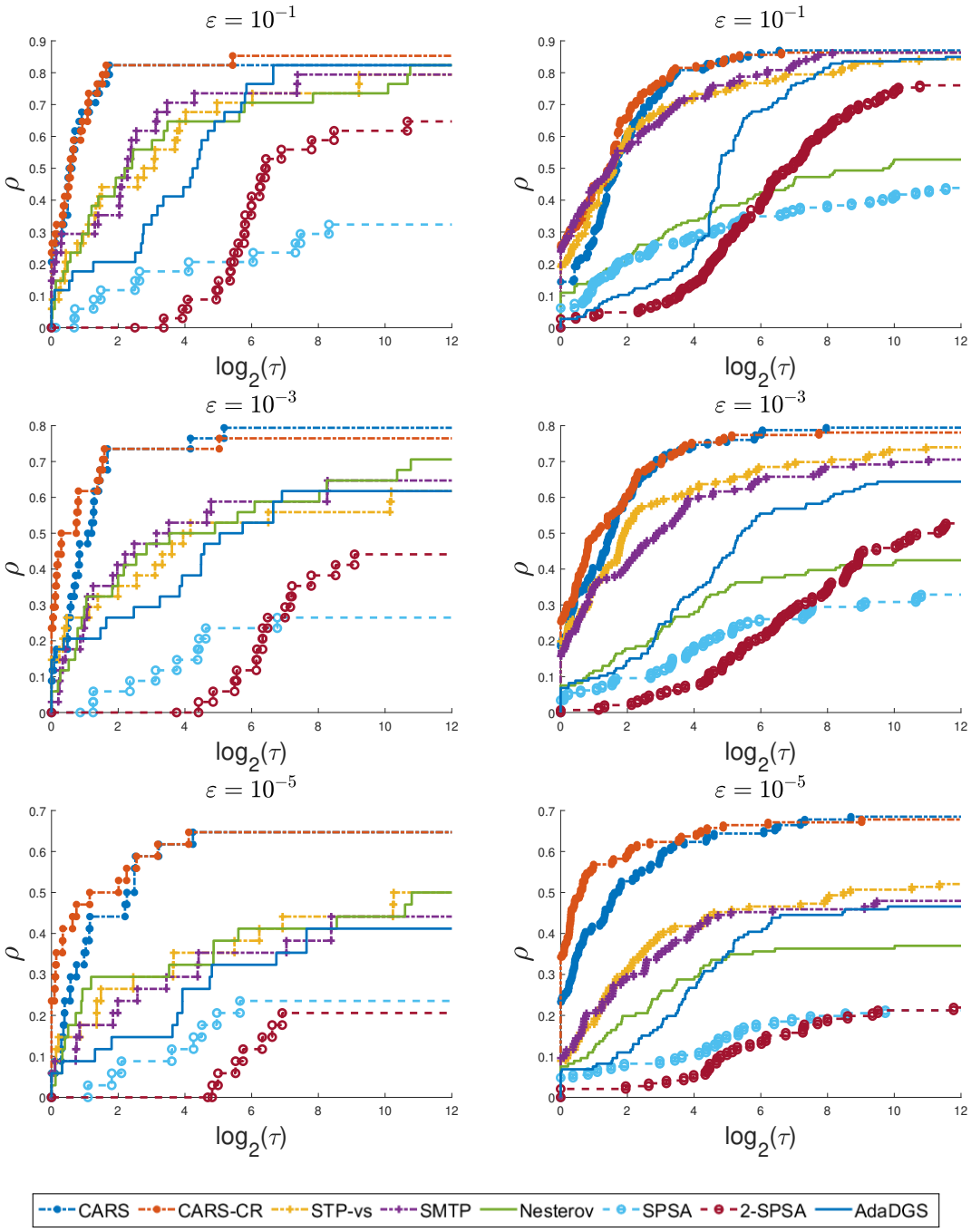

Figure 2.2: Performance profiles on Moré-Garbow-Hillstrom problems (**left**) and CUTEst problems (**right**), for various target accuracies $\varepsilon = 10^{-1}$ (**top**), 10^{-3} (**middle**), and 10^{-5} (**bottom**). Our results demonstrate that CARS and CARS-CR consistently outperform other methods in terms of both efficiency (ρ at low τ values) and robustness (ρ at high τ values.) at all levels of accuracy.

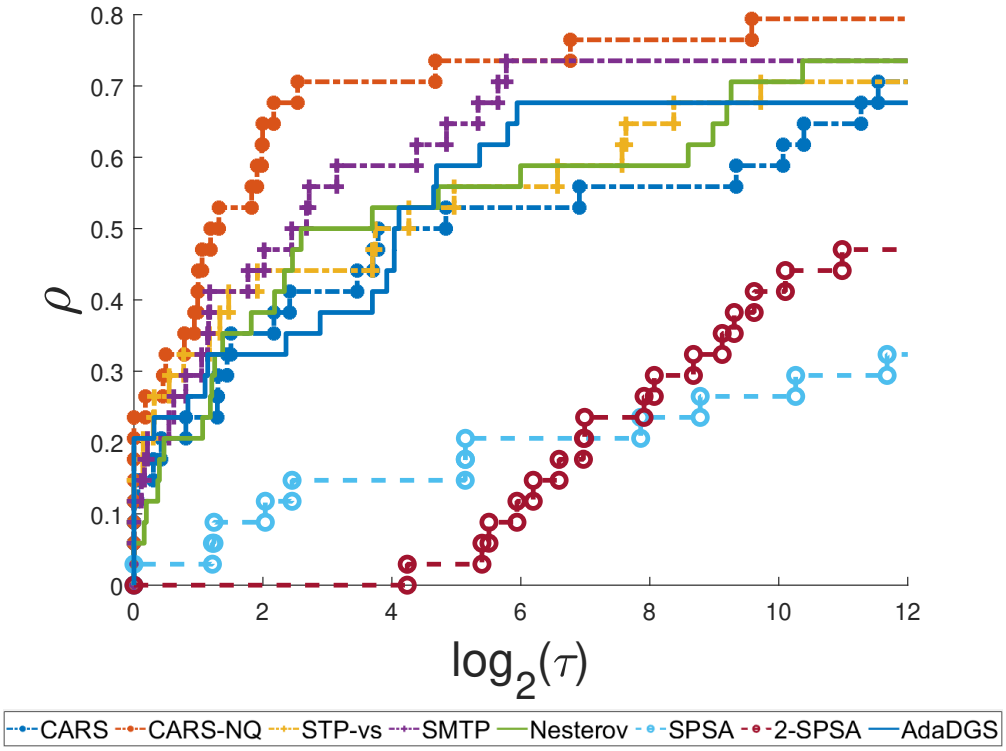

Figure 2.3: Performance of each algorithm on Moré-Garbow-Hillstrom Problems with sinusoidal noise $f_{\text{osc}}(x) = \psi \frac{1}{d} \sum_{i=1}^{d} (1 - \cos(\phi x_i))$, where $\psi = 0.05\varepsilon(f(x_0) - f_\star)$ and $\phi = 100\pi$. The target accuracy ε is set to 10^{-3}.

Figure 2.4: Adversarial examples with misclassified labels on MNIST generated with CARS. More pictures are available in Appendix A.1.

optimization problem. In order to ensure the attacked image $x_{\text{nat}} + x$ appears natural, a pixel-wise bound $\|x\|_\infty \leq \varepsilon_{\text{atk}}$ is usually enforced. CARS showed state-of-the-art performance in generating black-box adversarial attacks for $\mathcal{N}$ trained on the MNIST digit classification dataset [LCB10].

In our experiments, $\mathcal{N}$ is a two-layer CNN achieving 99% test accuracy on unperturbed images. We use $\varepsilon_{\text{atk}} = 0.2$ and consider all $10,000$ images from the test set of MNIST. We consider an attack a success if it fools $\mathcal{N}$ before a budget of $10,000$ queries is met. The success rates, median and average queries for successful attacks are shown in Table 2.2. The results from ZOO [CZS17], PGD-NES [IEA18], and ZOHA-type algorithms [YHF18] are cited from [YHF18]. As pointed out in Section 2.2.1, the choice of sampling directions for CARS is not restrictive. Hence we used a similar initialization and distribution $\mathcal{D}$ as the Square Attack [ACF20], which is known to be particularly well-suited for attacking CNN models. Visualization of attacked images is partly shown in Figure 2.4. More pictures and detailed settings can be found in Appendix A.1.

2.7.5 Benchmarking the Performance of SHIPS

In this section we present a numerical result for SHIPS, which is a combination of CARS and the randomized matrix inversion presented in Section 2.5. The main purpose of this comparison is to assess SHIPS' ability to generate effective search directions. Thus, we contrast it with Exact Gradient Descent (Exact GD), and variance-reduced version of CARS and Nesterov-Spokoiny (CARS-VarRed and NS-VarRed, respectively.)

For a fairer comparison with methods based on exact derivatives, we, in the case of DFO methods, sample d directions and use $2d$ queries to estimate the directional derivatives per iteration.

Variance reduction, achieved by utilizing multiple samples when calculating the expectation, has shown to work well for Nesterov-Spokoiny [SHC17, MGR18]. In contrast, CARS, does not have a known reduced variance version yet. For this, we apply a multi-sample extension (2.8) as introduced in Section 2.1.2.

The test function for our benchmark is given by:

$$f(x) = \frac{1}{2}x^\top Ax + \frac{1}{12}\sum_{i=1}^{d} \alpha_i x_i^4, \tag{2.69}$$

where A is a random positive definite matrix with $A = G^\top G$ with $G_{ij} \overset{i.i.d}{\sim} \mathcal{N}(0,1)$, and $\alpha_i \overset{i.i.d}{\sim} \text{Unif}(0,1)$.

For each iteration, we used d uniformly random directions on the unit sphere, and consumed $2d$ queries to estimate the respective directional derivatives. The methods labeled *Exact GD* and *Exact SHIPS* are provided with the exact gradient and Hessian for comparative purposes. As demonstrated in Figure 2.5, SHIPS deivers superior convergence speed.

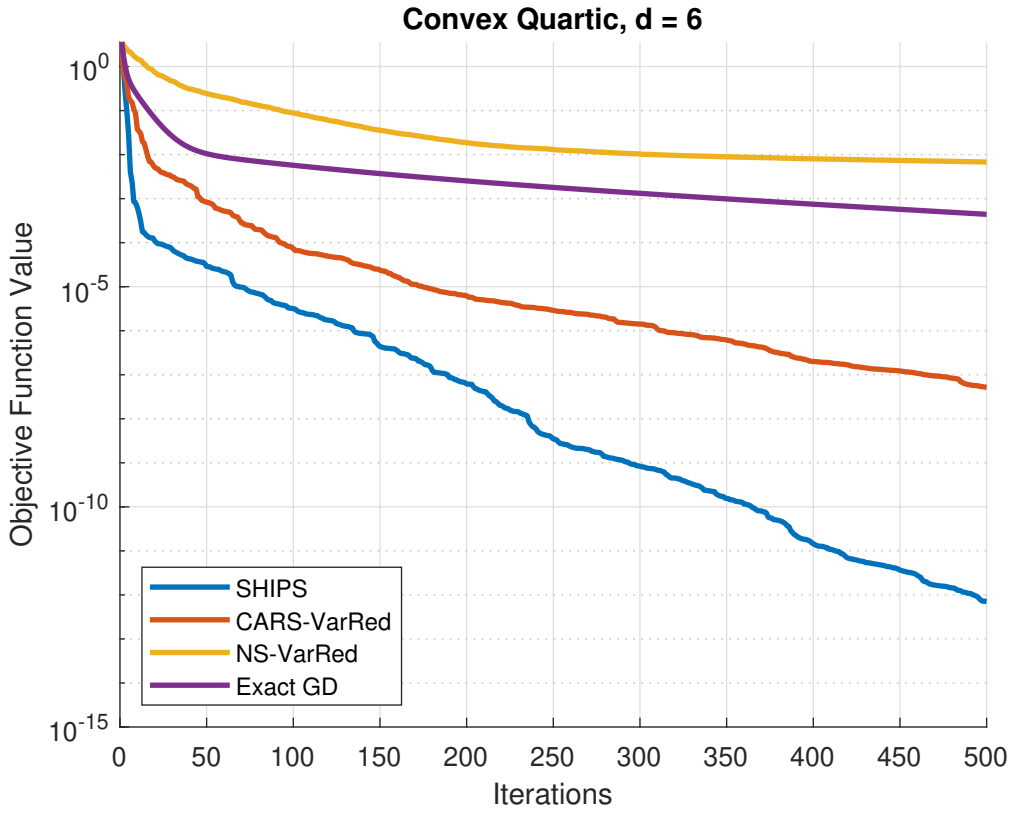

Figure 2.5: The x-axis denotes the number of iterations, not the number of function queries. Among all the methods, SHIPS shows the best convergence rate, outperforming even Exact GD, as it effectively utilizes the curvature information.

2.8 Conclusion Remarks

We proposed three query-efficient and lightweight DFO algorithms: CARS, CARS-CR and CARS-NQ. Our analysis establishes the convergence on strongly convex functions for CARS and convex functions for CARS-CR. Specifically, we develop a novel and rigorous analysis on the finite difference errors and the probability of significant descents of the objective function. CARS can incorporate various distributions, making it highly adaptable to a range of problem-specific distributions. We demonstrate the efficacy of CARS and its variants through benchmark tests, where they outperform existing methods in minimizing non-convex functions as well.

CHAPTER 3

Bridging the Gap Between Local and Global DFO Method Through Inspection Strategy

3.1 Introduction

In this chapter, we discuss the inspection strategy for DFO methods, focusing on filling the significant gap between global and local optimization strategies.

When the objective function is non-convex and has spurious local minima, it is in general difficult to find the global minimum using local optimization methods. Global optimization methods In such cases, local optimization methods are often used to find a local minimum. However, the quality of the local minimum is not guaranteed, and it is often difficult to quantify how good the solution is. Global optimization methods seek the absolute best solution throughout the entire solution space. While this ensures the optimal solution is found, these methods can be computationally intensive, particularly for high-dimensional problems. On the contrary, local optimization methods are geared towards finding a suitable solution within a localized area of the solution space. Although efficient, these methods may overlook the global optimum if it is located outside a certain neighborhood.

We delve into the dichotomy between these two optimization strategies and discuss the R-local optimization approach [CSY19], designed to bridge this gap. R-local optima lie between local and global optima, and often ensures a satisfactory solution quality by providing a definable radius for the solution's local neighborhood.

We also explore how R-local optimization interacts with DFO methods. Despite their

advantages, R-local optimization methods can be computationally expensive, particularly when the cost of sampling the objective function is not significantly cheaper than the gradient. However, in the context of DFO, the addition of inspections may not substantially impact the overall sample complexity.

Our proposed framework directs local DFO methods towards an R-local minimum using a similar number of additional queries, up to a constant factor, to those the method would typically consume *per iteration*. This algorithm that doesn't rely on a global surrogate model, which can be computationally demanding in high-dimensional scenarios.

3.1.1 The gap between global optimization and local optimization

The majority of the optimization methods are either global optimization or local optimization methods. They approach these problems in very different ways, which creates a gap between them.

Global Optimization This class of methods aims to find the absolute best solution (up to a small tolerance) in the *entire* solution space for an optimization problem. In other words, it searches for the overall minimum or maximum. The techniques used for global optimization are typically exhaustive, as they have to explore the entire problem space to ensure they have found the best possible solution up to the tolerance. This is because there may be many local optima, but only one (or a few) global optima. As a result, these methods can be computationally expensive, particularly for complex, high-dimensional problems.

Local Optimization This class of methods, on the other hand, focuses on finding a good solution in a neighborhood, no matter how small it is, of a specific point in the solution space. This does not necessarily mean that the solution will be the best overall; instead, it means that there is no better solution in some vicinity of the found one. Thus, local optimization methods can be faster and less computationally expensive than global ones, but they may

miss the global optimum if it lies outside of the selected neighborhood.

Gap between these two types of optimization The gap can be understood through the following perspectives.

Global optimization guarantees the best solution, while local optimization only guarantees the best solution within a local neighborhood. However, the size of the neighborhood is not known or controllable.

Due to the need to explore the entire solution space, global optimization usually requires much more computational effort, causing much longer running time, than local optimization.

In practice, the choice between local and global optimization often depends on the specific problem, the resources available, and the acceptable trade-offs. For low-dimensional problems where it is crucial to find the absolute best solution, global optimization becomes necessary. However, for many problems, a solution that is "good enough" is sufficient.

Local methods, despite only producing local solutions, may be preferred and sometimes are only the choice due to their efficiency. To improve the solution quality of local methods, meta-heuristics were introduced. They include using multiple starting points, simulated annealing, genetic algorithms that allow occasional moves in the worse direction, etc. However, neither local methods nor their meta-heuristic improvements could quantify how "good enough" their solutions are.

Filling the gap by R-local optimization According to [CSY19], an R-local minimizer is a type of local minimizer that specifies a radius of the local neighborhood around the minimizer.

More specifically, suppose we would like to minimize a function $f : \mathbb{R}^n \to \mathbb{R}$. We say that $x^\star \in \mathbb{R}^n$ is an R-local minimizer of f if that $x^\star$ is a minimizer of f in the closed ball $x \in \mathbb{R}^n : \|x - x^\star\| \leq R$. This means $x^\star$ is a local minimizer in the neighborhood of radius R around it.

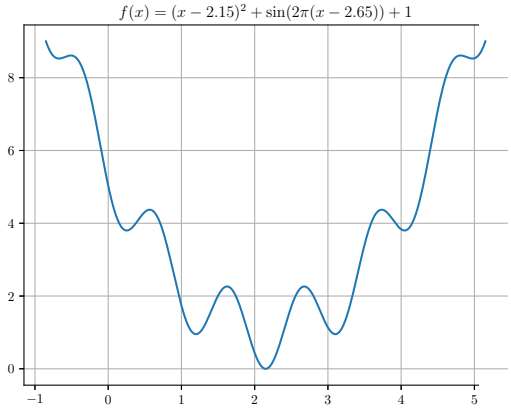

Figure 3.1: A function with spurious local minima. When R is sufficiently large, an R-local minimum is a global minimum.

The concept of an R-local minimizer is often used when we want to guarantee the solution found to have a sufficient quality in a vicinity of a defined size. This can be helpful when the global minimizer cannot be easily found or when the function is too complex to optimize over its entire domain. The value of R could be chosen based on some prior knowledge or through a process of trial and error.

3.1.2 DFO and R-local optimization

R-local optimization offers clear advantages over purely local methods in terms of discovering better solutions. However, it often incurs a higher computational cost, specifically requiring $\mathcal{O}(s \exp(O(d')))$ samples for s blocks of dimension d' as shown in [CSY19]. This can become prohibitively expensive in certain scenarios, particularly when the cost of sampling the objective function and its gradient is approximately equal. However, in derivative-free optimization, unless there are special structures (such as sparse gradients described in [CMY22b]), an additional factor of d, compared to its gradient-based counterpart, in the sample complexity is unavoidable. Consequently, performing inspections may not significantly impact the overall sample complexity, but only up to a constant factor.

3.1.3 Assumptions and Notation

Our proposed method, Inspect as you Run is a framework that can be applied to any local DFO method. We assume that the local DFO method $\mathcal{A}$ is given, and the objective function f satisfies the conditions for $\mathcal{A}$ to converge to a local minimum. We also assume that f is $\bar{L}$-Lipschitz continuous. Note that, however, if ∇f is already assumed to be Lipschitz, as in the analyses of many methods, this assumption implies the Lipschitz continuity of f.

Also, as in the previous chapter, let $\|\cdot\|$ denote the Euclidean norm, $\mathrm{Unif}(S)$ be the uniform distribution over S, and the unit sphere be written as $\mathcal{S}^{d-1}$. In addition, the closed ball of radius R centered at x is denoted as $B(x, R) = \{y \in \mathbb{R}^d : \|y - x\| \leq R\}$.

We then introduce a notion of an approximate R-local minimizer, which generalizes the R-local minimizer:

Definition 4. Given a descent threshold ν, a point $\bar{x}$ is said to be an *approximate R-local minimizer* if $f(\bar{x}) \leq f(x) + \nu$ for all $x \in B(\bar{x}, R)$.

Furthermore, defining the following notion of trapping would be useful, because we want to restrict how far the iterates wander around while we inspect nearby points.

Definition 5. Let $\{x_k\}_{k=0}^K$ be a sequence in $\mathbb{R}^d$. We say $\{x_k\}$ is trapped in a D-ball for $k_0 \leq k \leq k_1$ if $\|x_k - x_{k'}\| \leq D$ for all $k_0 \leq k, k' \leq k_1$.

3.2 Main Results

One of the core ideas in IR is, at each iteration, to restrict the cost of inspection up to same order of magnitude as $\mathcal{A}$. This ensures that if both $\mathcal{A}$ and $\mathcal{A}$ with IR are stuck at the same local minimum, the overall sample complexity differs only by a constant factor. Moreover, this restriction simplifies the complexity analysis imposed by the inspection as it mainly focuses on the iteration counts. For line-search type methods, like Nesterov-Spokoiny (NS) [NS17], Stochastic Three Point (STP) and its momentum variant (SMTP) [BGR20, GBS19],

and CARS from the previous chapter, the number of queries per iteration is $\mathcal{O}(1)$. On the other hand, for local methods requiring more queries per iteration, such as SPSA [Spa92], 2SPSA [Spa00], DGS [ZTL20], AdaDGS [TZ20] and others using an adjustable size of batch of sample directions [MGR18, SHC17], more inspections can be performed each iteration.

Algorithm 6 Inspect as You Run

Input: x_0: initial point; r: sampling radius; R: inspection radius; $\mathcal{A}$: One step of a DFO method, generating the next iterate; n_k: maximum number of inspections at k-th iteration; ν: descent threshold
Get the oracle $f(x_0)$.
for $k = 1, \cdots, K$ **do**
    Compute $x_{k,0} = \mathcal{A}(x_k)$
    for $j = 1, \cdots, n_k$ **do**
        Compute $f(x_{k,j})$
        if $f(x_{k,j}) < f(x_{k,0}) - \nu$ **then**
            Set $x_{k+1} = x_{k,j}$ and *break*
        end if
    end for
    If no successful inspections, set $x_{k+1} = x_{k,0}$
end for
Output: x_K: estimated optimum point.

3.2.1 Analysis on the High Probability Guarantee

This section presents an analysis of the point derived from Algorithm 6. The main result states that if the recent iterations are trapped in a small region without successful inspections, then it is an R_0-local minimum with high probability, for some $R_0 < R$.

Theorem 3.2.1. *Let $\{x_k\}_{k=1}^K$ be the sequence of points obtained from Algorithm 6. Assume that no successful inspections occur for all $k \geq k_0$, and let m be the total number of inspections for $k \geq k_0$. Suppose*

$$\|x_k - x_K\| \leq D < R \quad \textit{for all } k \geq k_0.$$

Define $R_0 = R - D$ and choose a positive $\tilde{r} < D$. Then, x_K is an R_0-local minimum up to

$\eta = \bar{L}\tilde{r} + \nu$ *with probability at least* $1 - \exp(-m(\tilde{r}/R)^d)$.

Proof. Begin by noting that the ball $B(x_K, R_0 + \tilde{r})$ is a subset of $B(x_k, R)$ for all $k \geq k_0$. Hence, for a random variable $z \sim \text{Unif}(B(x_k, R))$, the conditioned random variable $z|\mathcal{A}$, where $\mathcal{A}$ is the event $z \in B(x_K, R_0 + \tilde{r})$, follows the uniform distribution over $B(x_K, R_0 + \tilde{r})$.

At iteration $k = K$, define $S = \{y_i\}_{i=1}^M$ as the subset of the recent m inspection points contained within $B(x_K, R_0 + \tilde{r})$. Then M follows the binomial distribution $M \sim \text{Binomial}(m, \phi_1)$ with $\phi_1 = \left(\frac{R_0+\tilde{r}}{R}\right)^d$.

We now demonstrate that if S is sufficiently dense, x_K becomes an approximate R_0-local minimum. Consider

$$\tilde{x} := \underset{x \in B(x_K, R_0)}{\arg\min}\ f(x).$$

If there exists $y_i \in S \cap B(\tilde{x}, \tilde{r})$, knowing y_i is not a successful inspection, we obtain

$$f(x_K) \leq f(\tilde{x}) + (f(y_i) - f(\tilde{x})) + \nu \leq \min_{x \in B(x_K, R_0)} f(x) + \bar{L}\tilde{r} + \nu,$$

implying that x_K is an R_0-local minimum up to $\eta = \bar{L}\tilde{r} + \nu$.

Finally, we need to find the probability bound for $S \cap B(\tilde{x}, \tilde{r})$ not being empty. As each $y_j \sim \text{Unif}(B(x_K, R_0 + \tilde{r}))$ and $B(\tilde{x}, \tilde{r}) \subseteq B(x_K, R_0 + \tilde{r})$, we get $\phi_2 := \mathbb{P}[y_j \in B(\tilde{x}, \tilde{r})] =$

$\left(\frac{\tilde{r}}{R_0+\tilde{r}}\right)^d$. Consequently,

$$
\begin{aligned}
&\mathbb{P}[y_j \notin B(\tilde{x},\tilde{r}) \text{ for all } j=1,\cdots,M] \\
&= \sum_{M'=0}^{\infty} (1-\phi_2)^{M'} \mathbb{P}[M=M'] \\
&= \mathbb{E}_M(\exp(M\log(1-\phi_2))) \\
&\overset{(a)}{=} (1-\phi_1+\phi_1\exp(\log(1-\phi_2)))^m \\
&= (1-\phi_1\phi_2)^m \\
&\leq \exp(-m\phi_1\phi_2), \qquad\qquad (1-x\leq\exp(-x))
\end{aligned}
$$

where (a) is from the moment generating function of binomial distributions. And since $\phi_1\phi_2 = (\tilde{r}/R)^d$, $S\cap B(\tilde{x},\tilde{r})$ is nonempty with a probability of at least $1-\exp(-m(\tilde{r}/R)^d)$. □

A direct implication of Theorem 3.2.1 is that if the iterates get trapped and no further successful inspections are found since then, the solution obtained is an approximate R_0-local minimum with probability at least $1-\delta$, given the count of such consecutive inspections exceeds $\log(\delta^{-1})\left(\frac{R}{\tilde{r}}\right)^d$.

3.2.2 Discussion on IR

Unlike global DFO methods that constructs a surrogate model in the entire domain, it is easier for local methods to exploit the structure of the problem. For instance, if most variables are not highly correlated, Coordinate Descent (CD) [Wri15] or Block Coordinate Descent (BCD) [Tse01] analogue of DFO methods can be extremely useful. For instance, such variants can be easily obtained by replacing the sampling distribution in CARS. This room for the choice depending on the problem structure is one of the advantages of our method and really fills the gap between the local and global DFO methods.

3.3 Experimental Results

In this section we present the experimental results for supporting the effectiveness of inspection strategy for improving the solution quality of local DFO methods. As discussed in the previous section, we also provide the comparison between the local method equipped with inspection strategy and a global method, to emphasize the ease of problem structure exploitation.

Except for the last experiment, we benchmark a vanilla CARS and its IR version.

Spurious Local Minima

We start with a simple quadratic function with added sinusoidal noise. An 1-D illustration is shown in Figure 3.1. This function has spurious local minima, whose number grows exponentially with the problem dimension d.

$$f(x) = \sum_{i=1}^{d} \left((x_i)^2 + 0.2 \sin \left(10\pi (x_i - \frac{1}{20}) \right) + 0.2 \right)$$

As depicted in Figure 3.2, the inspection strategy consistently identifies superior local minima, often reaching the global minima in a relatively lower dimension of 6. We will revisit this example with a substantially higher dimension ($d = 300$) for the comparison with global methods.

Ackley's Functions

Ackley's functions are well-known non-convex functions with numerous spurious local minima:

$$f(x) = -20 \exp \left(-0.2 \sqrt{\|x\|^2 / d} \right) - \exp \left(\sum_{i=1}^{d} \cos(2\pi x_i)/d \right) + e + 20.$$

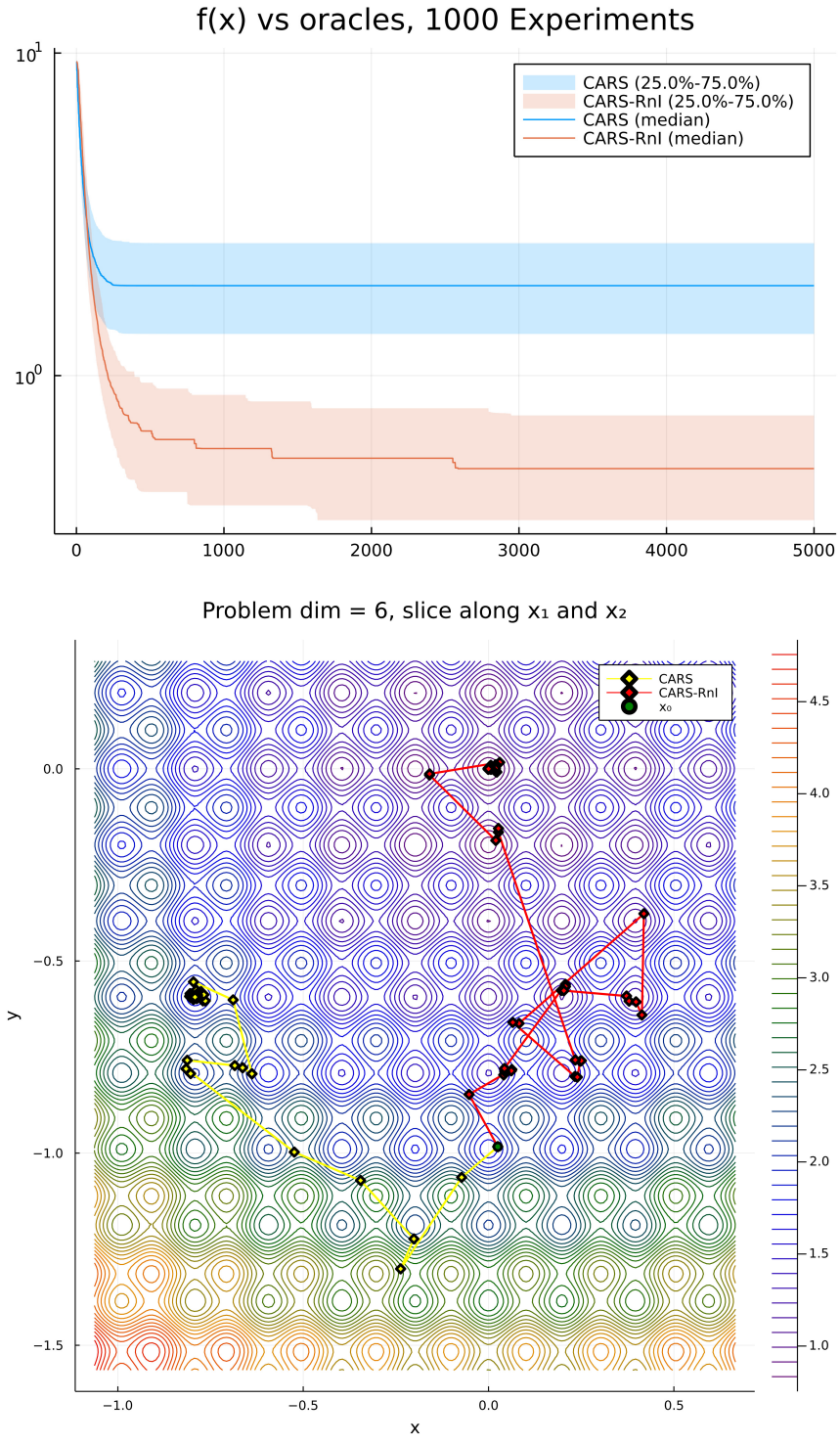

Figure 3.2: Comparison of CARS and the Inspect-as-Running version of CARS for the quadratic function with sinusoidal noise.

[CSY19] also introduced an asymmetric variant:

$$f(x,y) = -20\exp(-0.04(x^2+y^2)) - \exp(0.7(\sin(xy)+\sin y) + 0.2\sin(x^2)) + 20$$

We tested both functions and again, observed that the IR version consistently found better minima, while the vanilla CARS frequently got trapped in a non-global local minima.

K-means Clustering

Let $\{x_i\}_{i=1}^n$ be a set of n points in $\mathbb{R}^d$ and $\{z_j\}_{j=1}^K$ be K points in $\mathbb{R}^d$. Let the variable $Z = [z_1, \cdots, z_K] \in \mathbb{R}^{d\times K}$ be the matrix of the cluster centers. The K-means clustering problem is to find the optimal Z that minimizes the following objective function:

$$f(Z) = \frac{1}{2n}\sum_{i=1}^{n} \min_{j=1,\cdots,K} \|x_i - z_j\|^2.$$

We tackle this problem with derivative-free methods by treating Z as a vector in $\mathbb{R}^{dK}$.

Synthetic Gaussian data from [YPO18]

The first problem has synthetic Gaussian data, which is a mixture of four multivariate Gaussian distributions with $n = 1000$ each. This problem is introduced in [YPO18]. We used the same means and covariance matrices:

$$\mu_1 = [-5,-3], \mu_2 = [5,-3], \mu_3 = [0,5], \mu_4 = [2.5,4],$$

and

$$\Sigma_1 = \begin{bmatrix} 0.8 & 0.1 \\ 0.1 & 0.8 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1.2 & 0.6 \\ 0.6 & 0.7 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 0.5 & 0.05 \\ 0.05 & 1.6 \end{bmatrix}, \Sigma_4 = \begin{bmatrix} 1.5 & 0.05 \\ 0.05 & 0.6 \end{bmatrix}.$$

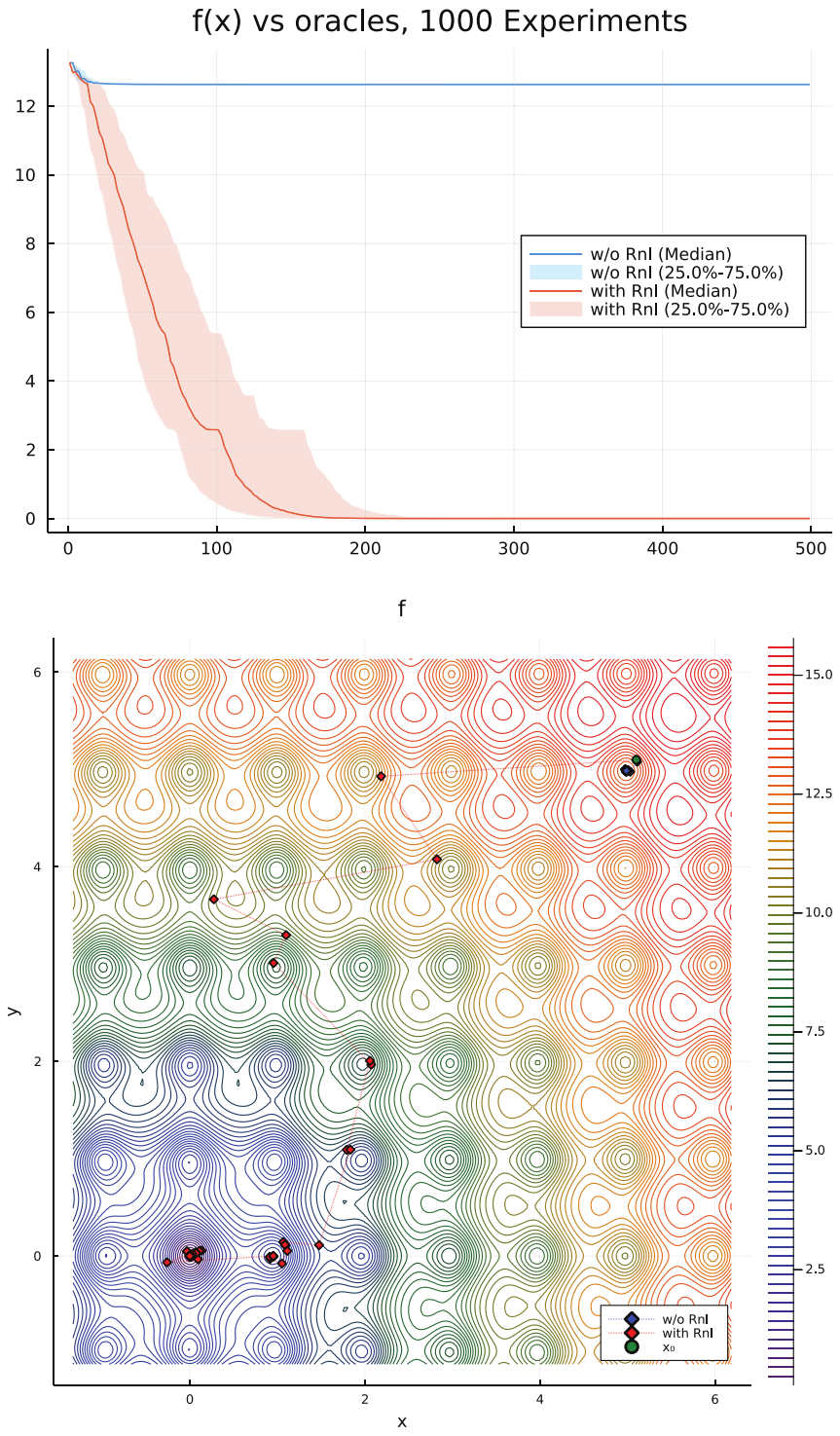

Figure 3.3: Comparison of CARS and the Inspect-as-Running version of CARS for the Ackley function

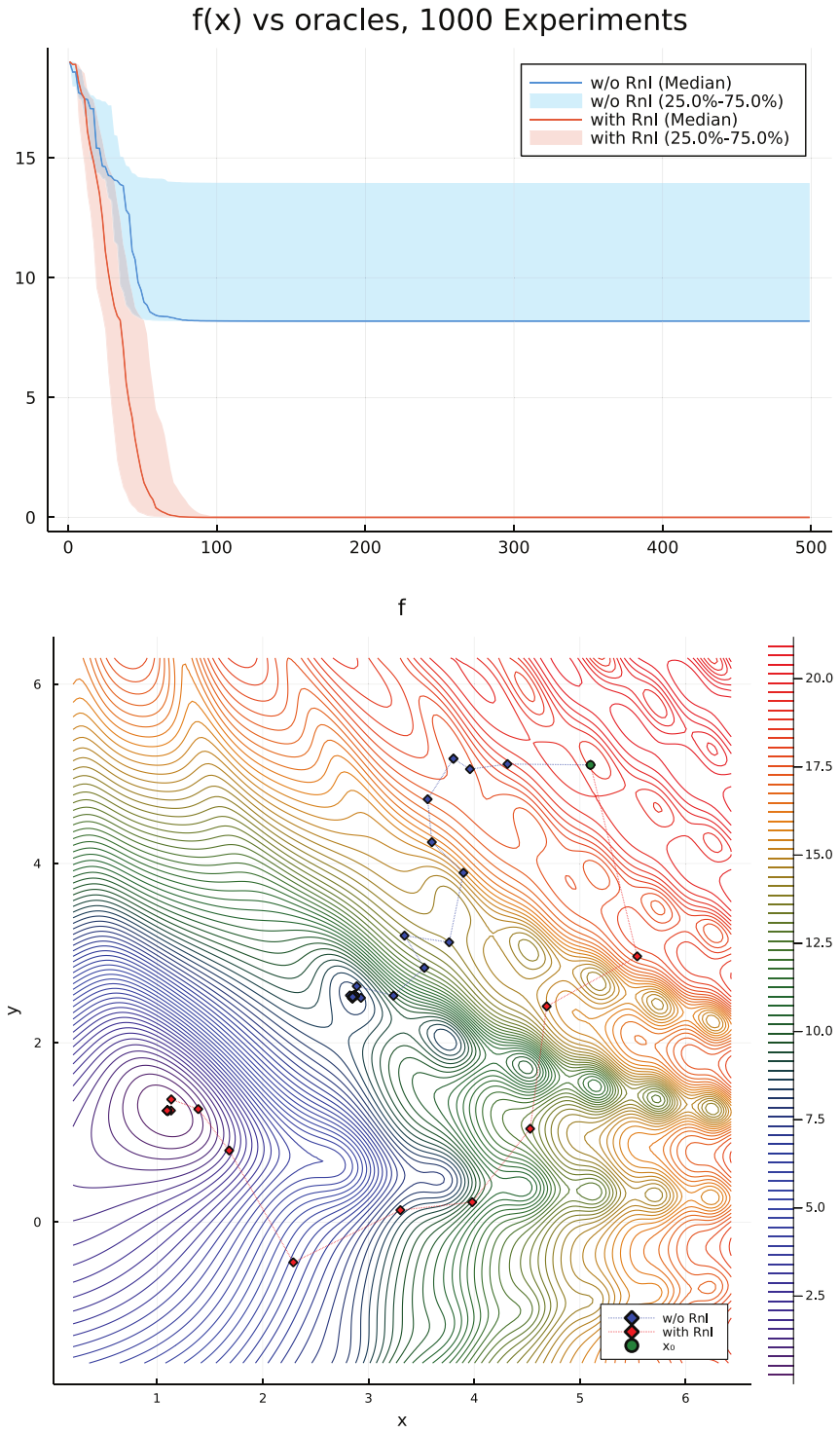

Figure 3.4: Comparison of CARS and the Inspect-as-Running version of CARS for the asymmetric Ackley function

Iris data

The second K-means clustering problem consists of the Iris dataset. This dataset contains 150 samples of three different species of Iris flowers. Each sample has four features: sepal length, sepal width, petal length, and petal width. We illustrate the results from 500 runs and the histogram of the final objective values in Figure 3.6. The figures demonstrate that the IR version of CARS finds better minima in most cases, and more rapidly.

Hyperparameter tuning

We applied the two versions of CARS to hyperparameter tuning for training a convolutional neural network on the MNIST dataset. The hyperparameters included the L^2 regularization parameter in the loss, the learning rate for the optimizer (AdaDelta), and the annealing rate for the scheduler (StepLR). To show the ability to escape the local minima more clearly, we started from a suboptimal initial point $x_0 = (0.5, 0.5, 0.5)$.

To reduce the variance, the objective function for the hyperparameter is set to be the mean of two samples, which measures the misclassification rate of the trained model with the given set of hyperparameters.

Spurious local minima in higher dimension

We revisit the function with spurious local minima in higher dimension, $d = 300$, and compare the performance of CARS and the IR version of CARS with the global method RBFOpt [CN18], which is one of the most efficient the global methods. RBFOpt took 15,073 seconds for 1,000 iterations (1,241 function evaluations), while CARS ran in 1.474 seconds for 30,000 function evaluations. Here we see that the computational cost of the solver itself (sub-problems) is much higher in the global methods, often making it impractical for

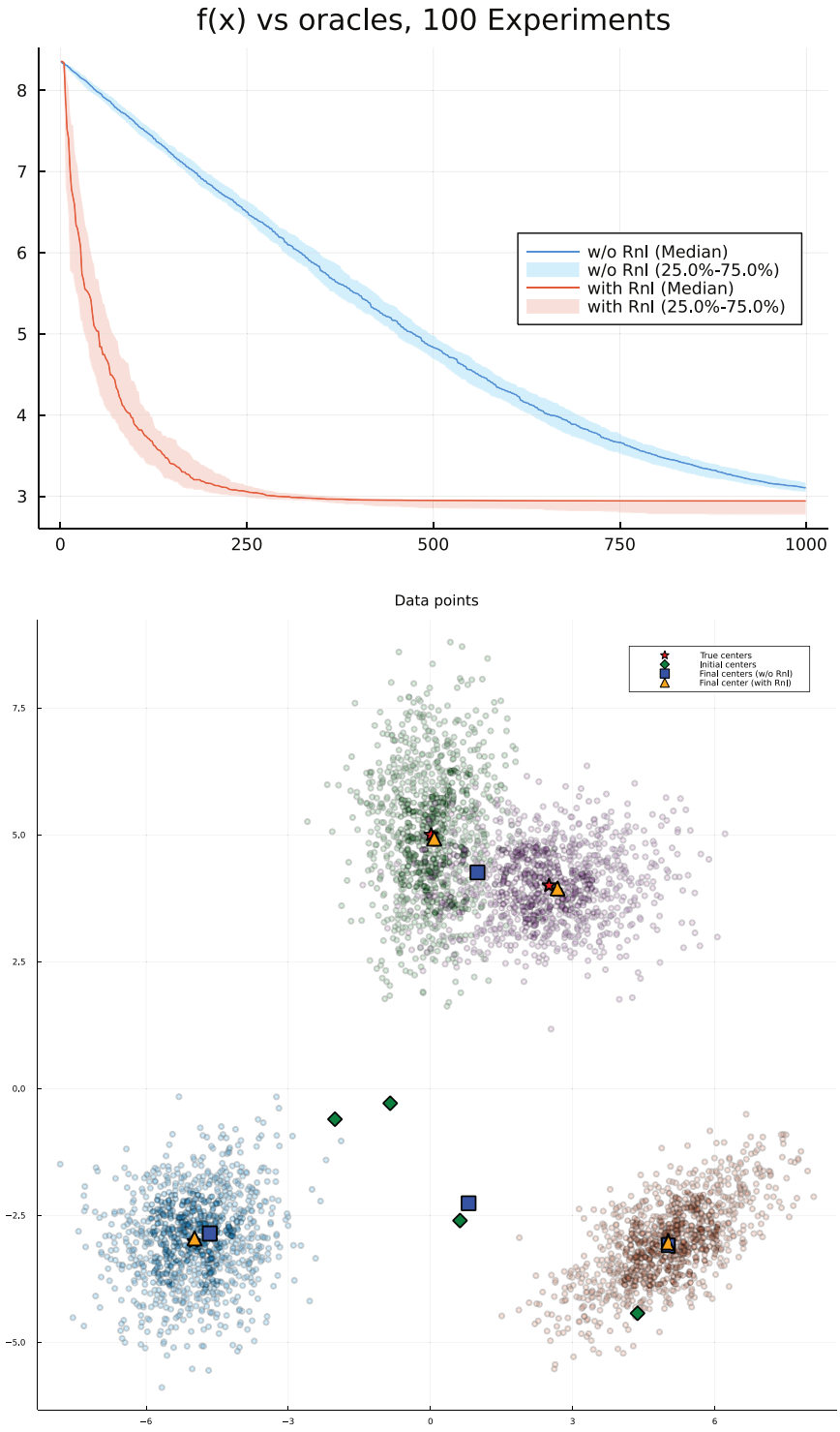

Figure 3.5: Comparison of CARS and the Inspect-as-Running version of CARS for the K-means clustering problem of synthetic Gaussian data

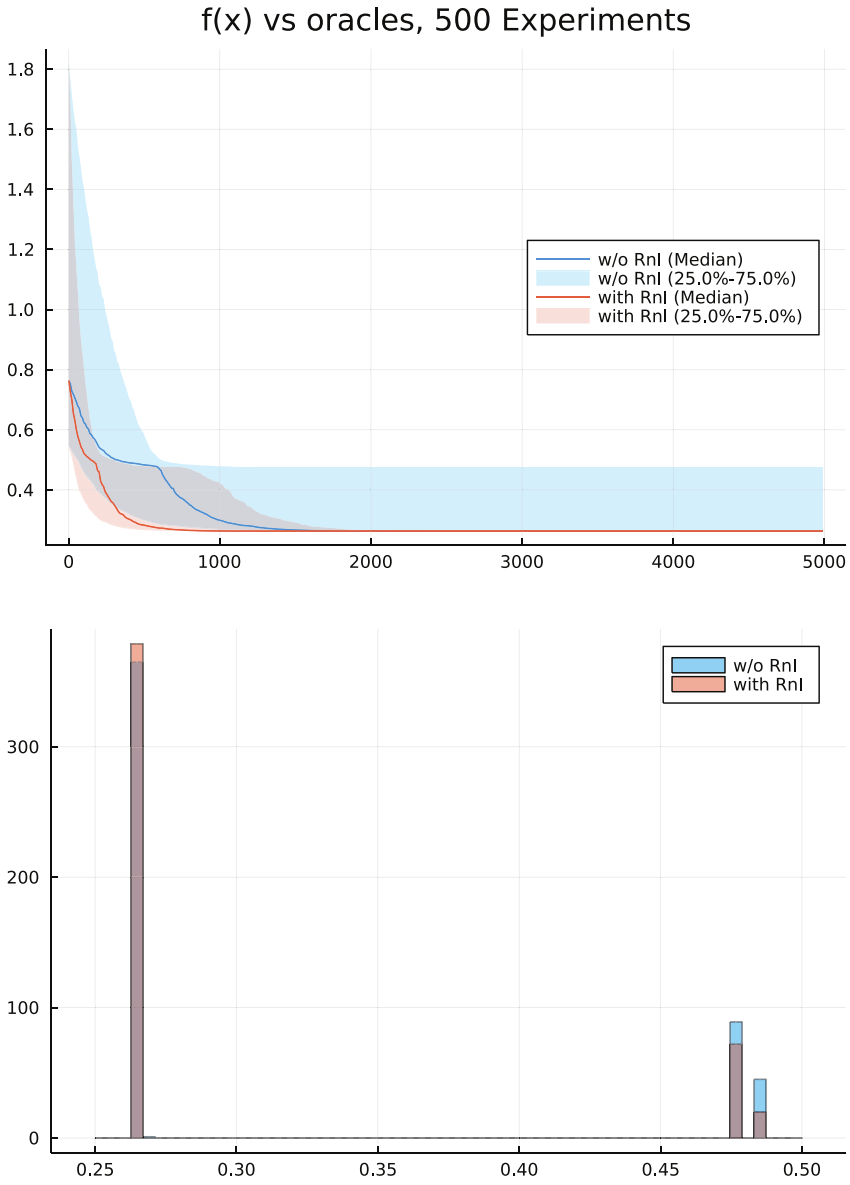

Figure 3.6: Comparison of CARS and the Inspect-as-Running version of CARS for the K-means clustering of the Iris dataset

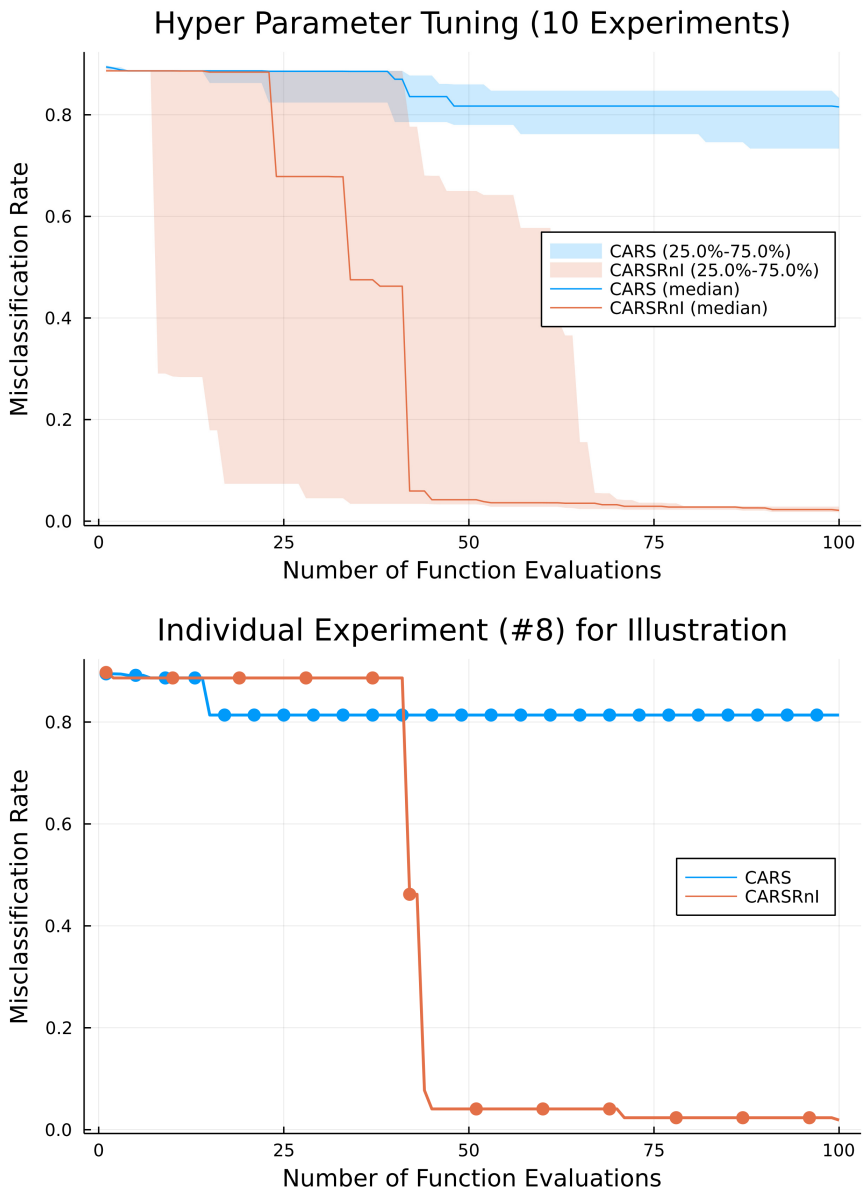

Figure 3.7: Comparison of CARS and the Inspect-as-Running version of CARS for hyperparameter tuning for training a convolutional neural network for MNIST dataset

high-dimensional problems. Furthermore, to utilize the structure of the problem, namely independence of each variable, we adopted the coordinate descent version, by simply replacing the sampling directions from $\mathrm{Unif}(\mathbb{S}^{d-1})$ to $\mathrm{Unif}(\{e_1, \cdots, e_d\})$, both for CARS and inspections.

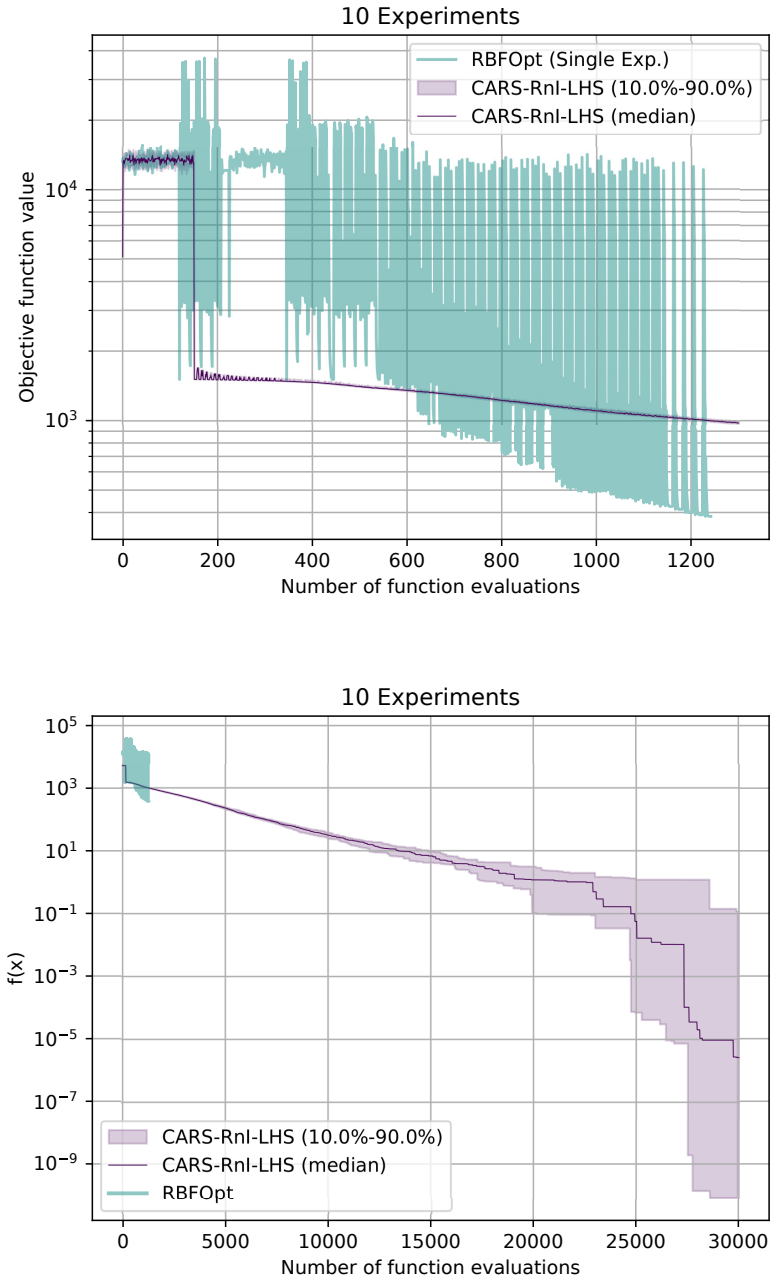

Figure 3.8: Comparison of CARS with IR and the RBFOpt in 300-dimensional problem with spurious local minima. Computation for RBFOpt is terminated after 1,000 iterations due to the computational cost.

APPENDIX A

A.1 Experimental Settings for Chapter 2

In this Section, we list the hyperparameters we used for each experiment. The code for all experiments can be found in `https://github.com/bumsu-kim/CARS`. We ran experiments on several machines to distribute the load. We used Intel i5-9400F with Nvidia RTX 2060 and i9-9940X with two RTX 2080.

Moré-Garbow-Hillstrom Problems. The Moré-Garbow-Hillstrom Problem set consists of 34 non-convex smooth functions, where the problem dimension lies between 2 and 100. This experiment is conducted in Matlab.

We consider a problem solved when $f(x_k) - f_\star \leq \varepsilon(f(x_0) - f_\star)$. The target accuracies used here are $\varepsilon = 10^{-1}, 10^{-3}$ and 10^{-5}. We used the recommended x_0 for each problem.

For CARS and CARS-NQ, we used the sampling radius $r_k = 0.01/\sqrt{k+1}$, $\hat{L} = 2$. For CARS-NQ, the number of quadrature points $q = 5$ is used. Having larger q increases the accuracy of the approximation of smoothed derivatives, but it also increases the cost (*i.e.* function queries required) at each step. Recall from Section 2.4 that using odd q is always more efficient than even q. Since $q = 3$ is essentially less accurate than CARS, we used several values of $q \geq 5$ and found that $q = 5$ is the best.

For STP [BGR20] and NSRS [NS17] we used the same hyperparameters as given in Section 8.1 of [BGR20]. We also used the same decreasing step-size for Stochastic Momentum Three Points method (SMTP) [GBS19]. For the momentum parameter β for SMTP, we followed [GBS19] and used $\beta = 0.5$. Namely, following the notations in [BGR20] and [GBS19],

$\mathcal{D} = \mathrm{Unif}(\mathbb{S}^{d-1})$ and

$$\alpha_k = \frac{1}{\sqrt{k+1}}, \tag{STP}$$

$$\alpha_k = \frac{1}{4(n+4)} \text{ and } \mu_k = 10^{-4}, \tag{NSRS}$$

$$\gamma_k = \frac{1}{\sqrt{k+1}}, \text{ and } \beta = 0.5. \tag{SMTP}$$

For SPSA [Spa92] and 2SPSA [Spa00], we used the Rademacher distribution (*i.e.* $(u_k)_i = \pm 1$ with probability 0.5) for $\mathcal{D}$, $\alpha = 0.602$, $\gamma = 0.101$, $A = 100$, $a = 0.16$, and $c = 10^{-4}$.

For AdaDGS [TZ20], we used the code provided by the authors, by implementing the original Python code in Matlab. Some modifications on hyperparameters are made due to the difference in the scale of problem dimension, and the lack of domain width. First, the original AdaDGS code performs experiments on high dimensional problems (*e.g.* $d = 1000$), whereas $2 \leq d \leq 100$ in this experiment. Also, the problems are unconstrained, and $\|x_0 - x_\star\|$ varies from order of 10^0 to 10^6. Thus we used the following modified hyperparameters (following the notation of [TZ20]):

1. The number of points used for line search $S = 100$, since the suggested value $0.05d(M-1)$ is too small for our experiments.

2. The initial smoothing(sampling) radius $\sigma_0 = 10^{-2}$. We tested $\sigma_0 = 5, 1, 10^{-1}, 10^{-2}$ and 10^{-3} and chose the best value. When $\sigma_0 \leq 10^{-1}$ then the results were similar.

For plotting the performance profile, we set the performance ratio $r_{p,s} = r_M$ when p is not solved by s. Having $r_M = \infty$ is ideal, but setting it by a sufficiently large number does not make any difference. We used $r_M = 10^{20}$.

Problems with Highly Oscillatory Noise. For the presence of oscillatory noise scenario, we performed experiments, by adding the noise to Moré-Garbow-Hillstrom Problems. For

each function in the benchmark set, we used the same noise

$$f_{\text{osc}}(x) = \psi \sum_{i=1}^{d} (1 - \cos(\phi_i x_i)),$$

with different noise level $\psi = 0.1\varepsilon(f(x_0) - f_\star)$, and the same frequency $\phi_i = 100\pi$.

We used the same hyperparameters for Moré-Garbow-Hillstrom Problems as in the previous experiment, except for CARS-NQ, where we used 50 times larger r.

Black-box Adversarial Attacks. In this section, we explain the experiment setting for black-box adversarial attacks and also provide the hyperparameters that we used. The CNN model we attack has two 5×5 convolutional layers with 6 and 16 output channels, followed by a 4×4 convolutional layer with 120 output channels. Then two fully connected layers with 84 and 10 units follows. Between layers we use ReLU, and between convolutional layers we use 2×2 max-pooling as well. Finally we apply log softmax to the output layer. The test accuracy of the trained model is 98.99%.

For more readable description, let the images have width and height both equal to s. Then the problem dimension d is s^2. For this particular experiment, we make three modifications to CARS. First, since the problem is highly non-convex ($h_r < 0$ at around 50% of the iteration), we don't compute x_{CARS} when $h_r < 0$ at k-th iteration. The second modification is due to the constraint of the problem. Let $\mathcal{F}$ be the feasible set:

$$\mathcal{F} = \{x \in [0,1]^d : \|x - x_0\| \leq \varepsilon_{\text{atk}}\}.$$

Inspired by [ACF20], we also compute $x_{\text{bdry}} = x_k - t_{\max} d_r u_k$, where

$$t_{\max} = \max\{t > 0 : x_k - t d_r u_k \in \mathcal{F}\}.$$

This is especially useful when $h_r < 0$ so we don't compute x_{CARS}. Therefore, CARS now

requires either 3 ($f(x_k \pm r_k u_k)$ and $f(x_{\text{bdry}})$) function queries or 4 ($f(x_{\text{CARS}})$ in addition) per iteration. To sum up,

$$x_{k+1} = \begin{cases} \arg\min\{f(x_k \pm r_k u_k), f(x_{\text{CARS}}), (x_{\text{bdry}})\} & \text{if } h_r > 0 \\ \arg\min\{f(x_k \pm r_k u_k), (x_{\text{bdry}})\} & \text{otherwise.} \end{cases}$$

Lastly, we perturbed x_0 by adding horizontal stripes. That is,

$$x_0 = \text{proj}_{\mathcal{F}}(x_{\text{orig}} + \varepsilon_{\text{atk}} v),$$

where $v_{i,:} = \pm(1, \cdots, 1)$ for $i = 1, \cdots, s$ and x_{orig} is the image to be attacked. This choice of initialization is found to be very effective in [ACF20].

We use the same sampling distribution as Square Attack [ACF20], which is known to be particularly well-suited for attacking CNN models. Identifying $s \times s$ images and d-vectors, this distribution generates a square block of all 1 or -1, *i.e.* $u_{r+1:r+w,c+1:c+w} = \pm 1$, where $0 \leq r, c \leq s - w$ and w is the window size. See [ACF20] for more details.

The window size w is determined by p, the fraction of the pixels changed by attack. Namely, w is the closest positive integer to $\sqrt{p}s$, and thus this p is an important hyperparameter in this distribution. In [ACF20], the authors used $p = 0.8$ in the initial stage and halve the value of p when the number of iteration k reaches $10, 50, 200, 1000, 2000, 4000, 6000$ and 8000, respectively.

For CARS, we used the initial $p = 0.2$ and halved it at $k = 2, 10, 40, 250, 500, 800, 1200$ and 1600. Note that the Square Attack only uses one function query at each iteration and CARS for black-box attack uses 3 or 4 queries, and thus p is being halved at similar stage of the algorithms, in terms of the function queries.

Finally, the sampling radius r_k is fixed by 1 for every iteration.

A.2 Visualization of Attacked Images

In this section we present the visualization of the attacked images, comparing them with the original images. To be more clear, the images in Figure A.1 have *Testset ID* (TID)'s from 100 to 109. When an image with label n has TID t, this means it is the t-th n appearing in the test set. See the code for more details on TID.

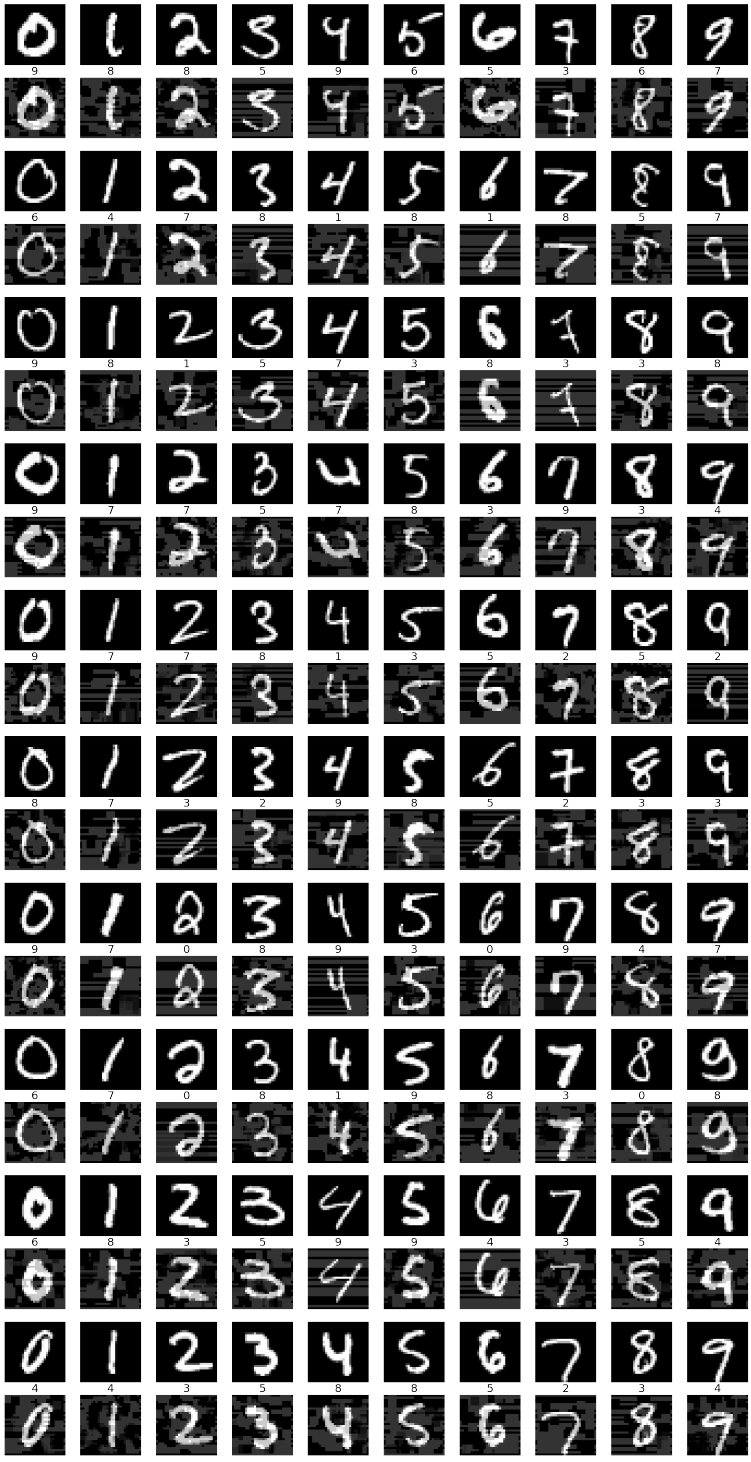

Figure A.1: Adversarial examples with misclassified labels on MNIST generated with CARS. For every two rows, a row of original images are shown, and the adversarial examples are right underneath them, with the misclassified labels in between.

REFERENCES

[ACF20] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. "Square attack: a query-efficient black-box adversarial attack via random search." In *European Conference on Computer Vision*, pp. 484–501. Springer, 2020.

[AS64] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, ninth Dover printing, tenth GPO printing edition, 1964.

[BB12] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization." *Journal of Machine Learning Research*, **13**(1):281–305, 2012.

[BBS20] Adel Bibi, El Houcine Bergou, Ozan Sener, Bernard Ghanem, and Peter Richtarik. "A Stochastic Derivative-Free Optimization Method with Importance Sampling: Theory and Learning to Control." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3275–3282, 2020.

[BCC21] Albert S Berahas, Liyuan Cao, Krzysztof Choromanski, and Katya Scheinberg. "A theoretical and empirical comparison of gradient approximations in derivative-free optimization." *Foundations of Computational Mathematics*, pp. 1–54, 2021.

[BCS21] Albert S Berahas, Liyuan Cao, and Katya Scheinberg. "Global convergence rate analysis of a generic line search algorithm with noise." *SIAM Journal on Optimization*, **31**(2):1489–1518, 2021.

[Ber97] Dimitri P Bertsekas. "Nonlinear programming." *Journal of the Operational Research Society*, **48**(3):334–334, 1997.

[BG21] Krishnakumar Balasubramanian and Saeed Ghadimi. "Zeroth-Order Nonconvex Stochastic Optimization: Handling Constraints, High Dimensionality, and Saddle Points." *Foundations of Computational Mathematics*, pp. 1–42, 2021.

[BGR20] El Houcine Bergou, Eduard Gorbunov, and Peter Richtarik. "Stochastic three points method for unconstrained smooth minimization." *SIAM Journal on Optimization*, **30**(4):2726–2749, 2020.

[CCG15] Yuxin Chen, Yuejie Chi, and Andrea J Goldsmith. "Exact and stable covariance estimation from quadratic sampling via convex programming." *IEEE Transactions on Information Theory*, **61**(7):4034–4059, 2015.

[CLM21] HanQin Cai, Yuchen Lou, Daniel McKenzie, and Wotao Yin. "A Zeroth-Order Block Coordinate Descent Algorithm for Huge-Scale Black-Box Optimization."

In *Proceedings of the 38th International Conference on Machine Learning*, pp. 1193–1203. PMLR, 2021.

[CMO22] Coralia Cartis, Estelle Massart, and Adilet Otemissov. "Global optimization using random embeddings." *Mathematical Programming*, pp. 1–49, 2022.

[CMY22a] HanQin Cai, Daniel Mckenzie, Wotao Yin, and Zhenliang Zhang. "A One-bit, Comparison-Based Gradient Estimator." *Applied and Computational Harmonic Analysis*, **60**:242–266, 2022.

[CMY22b] HanQin Cai, Daniel Mckenzie, Wotao Yin, and Zhenliang Zhang. "Zeroth-Order Regularized Optimization (ZORO): Approximately Sparse Gradients and Adaptive Sampling." *SIAM Journal on Optimization*, **32**(2):687–714, 2022.

[CN18] Alberto Costa and Giacomo Nannicini. "RBFOpt: an open-source library for black-box optimization with costly function evaluations." *Mathematical Programming Computation*, **10**:597–629, 2018.

[CO22] Coralia Cartis and Adilet Otemissov. "A dimensionality reduction technique for unconstrained global optimization of functions with low effective dimensionality." *Information and Inference: A Journal of the IMA*, **11**(1):167–201, 2022.

[CPP20] Krzysztof Choromanski, Aldo Pacchiano, Jack Parker-Holder, Yunhao Tang, Deepali Jain, Yuxiang Yang, Atil Iscen, Jasmine Hsu, and Vikas Sindhwani. "Provably robust blackbox optimization for reinforcement learning." In *Conference on Robot Learning*, pp. 683–696, 2020.

[CR21] Coralia Cartis and Lindon Roberts. "Scalable Subspace Methods for Derivative-Free Nonlinear Least-Squares Optimization." *arXiv preprint arXiv:2102.12016*, 2021.

[CR22] Coralia Cartis and Lindon Roberts. "Scalable subspace methods for derivative-free nonlinear least-squares optimization." *Mathematical Programming*, pp. 1–64, 2022.

[CRS18] Krzysztof Choromanski, Mark Rowland, Vikas Sindhwani, Richard Turner, and Adrian Weller. "Structured evolution with compact architectures for scalable policy optimization." In *International Conference on Machine Learning*, pp. 970–978. PMLR, 2018.

[CSC19] Minhao Cheng, Simranjit Singh, Patrick Chen, Pin-Yu Chen, Sijia Liu, and Cho-Jui Hsieh. "Sign-opt: A query-efficient hard-label adversarial attack." *arXiv preprint arXiv:1909.10773*, 2019.

[CSV09] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to derivative-free optimization*. SIAM, 2009.

[CSY19] Yifan Chen, Yuejiao Sun, and Wotao Yin. "Run-and-Inspect Method for nonconvex optimization and global optimality bounds for R-local minimizers." *Mathematical Programming*, **176**:39–67, 2019.

[CZS17] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. "ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models." In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 15–26, 2017.

[DM02] Elizabeth D Dolan and Jorge J Moré. "Benchmarking optimization software with performance profiles." *Mathematical Programming*, **91**(2):201–213, 2002.

[Fab71] Václav Fabian. "Stochastic approximation." In *Optimizing methods in statistics*, pp. 439–470. Elsevier, 1971.

[FGK18] Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. "Global convergence of policy gradient methods for the linear quadratic regulator." In *International Conference on Machine Learning*, pp. 1467–1476. PMLR, 2018.

[GBS19] Eduard Gorbunov, Adel Bibi, Ozan Sener, El Houcine Bergou, and Peter Richtárik. "A stochastic derivative free optimization method with momentum." *arXiv preprint arXiv:1905.13278*, 2019.

[GK20] Tobias Glasmachers and Oswin Krause. "The Hessian Estimation Evolution Strategy." In *International Conference on Parallel Problem Solving from Nature*, pp. 597–609. Springer, 2020.

[GKL19] Robert Gower, Dmitry Koralev, Felix Lieder, and Peter Richtárik. "RSN: Randomized subspace newton." In *Advances in Neural Information Processing Systems*, pp. 616–625, 2019.

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

[GLL88] Luigi Grippo, F Lampariello, and S Lucidi. "Global convergence and stabilization of unconstrained minimization methods without derivatives." *Journal of Optimization Theory and Applications*, **56**(3):385–406, 1988.

[GOT15] Nicholas IM Gould, Dominique Orban, and Philippe L Toint. "CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization." *Computational optimization and applications*, **60**(3):545–557, 2015.

[GR15] Luigi Grippo and F Rinaldi. "A class of derivative-free nonmonotone optimization algorithms employing coordinate rotations and gradient approximations." *Computational Optimization and Applications*, **60**(1):1–33, 2015.

[GS07] Luigi Grippo and Marco Sciandrone. "Nonmonotone derivative-free methods for nonlinear equations." *Computational Optimization and Applications*, **37**(3):297–328, 2007.

[GSS14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *arXiv preprint arXiv:1412.6572*, 2014.

[HDN20] Filip Hanzely, Nikita Doikov, Yurii Nesterov, and Peter Richtarik. "Stochastic Subspace Cubic Newton Method." In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4027–4038. PMLR, 13–18 Jul 2020.

[IEA18] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. "Black-box adversarial attacks with limited queries and information." In *International Conference on Machine Learning*, pp. 2137–2146. PMLR, 2018.

[JNR12] Kevin G Jamieson, Robert Nowak, and Ben Recht. "Query Complexity of Derivative-Free Optimization." In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[Kar74] VG Karmanov. "Convergence estimates for iterative minimization methods." *USSR Computational Mathematics and Mathematical Physics*, **14**(1):1–13, 1974.

[Kar75] VG Karmanov. "On convergence of a random search method in convex minimization problems." *Theory of Probability & Its Applications*, **19**(4):788–794, 1975.

[KBD21] David Kozak, Stephen Becker, Alireza Doostan, and Luis Tenorio. "A stochastic subspace approach to gradient-free optimization in high dimensions." *Computational Optimization and Applications*, **79**(2):339–368, 2021.

[KLT03] Tamara G Kolda, Robert Michael Lewis, and Virginia Torczon. "Optimization by direct search: New perspectives on some classical and modern methods." *SIAM review*, **45**(3):385–482, 2003.

[Kru83] VN Krutikov. "On the rate of convergence of the minimization method along vectors in a given directional system." *USSR Computational Mathematics and Mathematical Physics*, **23**(1):154–155, 1983.

[LCB10] Yann LeCun, Corinna Cortes, and CJ Burges. "MNIST handwritten digit database." *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, **2**, 2010.

[LCK20] Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O Hero III, and Pramod K Varshney. "A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications." *IEEE Signal Processing Magazine*, **37**(5):43–54, 2020.

[LKC18] Sijia Liu, Bhavya Kailkhura, Pin-Yu Chen, Paishun Ting, Shiyu Chang, and Lisa Amini. "Zeroth-order stochastic variance reduction for nonconvex optimization." *Advances in Neural Information Processing Systems*, **31**, 2018.

[LMW19] Jeffrey Larson, Matt Menickelly, and Stefan M Wild. "Derivative-free optimization methods." *Acta Numerica*, **28**:287–404, 2019.

[MGH81] Jorge J Moré, Burton S Garbow, and Kenneth E Hillstrom. "Testing unconstrained optimization software." *ACM Transactions on Mathematical Software (TOMS)*, **7**(1):17–41, 1981.

[MGR18] Horia Mania, Aurelia Guy, and Benjamin Recht. "Simple random search of static linear policies is competitive for reinforcement learning." In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 1805–1814, 2018.

[MR64] VA Mutsenieks and LA Rastrigin. "Extremal control of continuous multi-parameter systems by the method of random search." *Akademiia Nauk SSSR, Izvestiia1, Tekhnichekaia Kibernetika*, pp. 101–110, 1964.

[Nes12] Yu Nesterov. "Efficiency of coordinate descent methods on huge-scale optimization problems." *SIAM Journal on Optimization*, **22**(2):341–362, 2012.

[NM65] John A Nelder and Roger Mead. "A simplex method for function minimization." *The computer journal*, **7**(4):308–313, 1965.

[NP06] Yurii Nesterov and Boris T Polyak. "Cubic regularization of Newton method and its global performance." *Mathematical Programming*, **108**(1):177–205, 2006.

[NS17] Yurii Nesterov and Vladimir Spokoiny. "Random gradient-free minimization of convex functions." *Foundations of Computational Mathematics*, **17**(2):527–566, 2017.

[SC76] Günther Schrack and Mark Choit. "Optimized relative step size random searches." *Mathematical Programming*, **10**(1):230–244, 1976.

[SHC17] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. "Evolution strategies as a scalable alternative to reinforcement learning." *arXiv preprint arXiv:1703.03864*, 2017.

[SMG13] Sebastian U Stich, Christian L Muller, and Bernd Gartner. "Optimization of convex functions with random pursuit." *SIAM Journal on Optimization*, **23**(2):1284–1309, 2013.

[Spa92] James C Spall et al. "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation." *IEEE Transactions on Automatic Control*, **37**(3):332–341, 1992.

[Spa00] James C Spall. "Adaptive stochastic approximation by the simultaneous perturbation method." *IEEE Transactions on Automatic Control*, **45**(10):1839–1853, 2000.

[Ste72] Charles Stein et al. "A bound for the error in the normal approximation to the distribution of a sum of dependent random variables." In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*. The Regents of the University of California, 1972.

[Ste81] Charles M Stein. "Estimation of the mean of a multivariate normal distribution." *The annals of Statistics*, pp. 1135–1151, 1981.

[SY20] Ofer M Shir and Amir Yehudayoff. "On the covariance-hessian relation in evolution strategies." *Theoretical Computer Science*, **801**:157–174, 2020.

[Tao07] Terence Tao. "Lecture Notes 8 for 247B." *https://www.math.ucla.edu/∼tao/247b.1.07w/notes8.pdf*, 2007.

[Tse01] Paul Tseng. "Convergence of a block coordinate descent method for nondifferentiable minimization." *Journal of optimization theory and applications*, **109**(3):475, 2001.

[TZ20] Hoang Tran and Guannan Zhang. "AdaDGS: An adaptive black-box optimization method with a nonlocal directional Gaussian smoothing gradient." *arXiv preprint arXiv:2011.02009*, 2020.

[WDB18] Yining Wang, Simon Du, Sivaraman Balakrishnan, and Aarti Singh. "Stochastic zeroth-order optimization in high dimensions." In *International Conference on Artificial Intelligence and Statistics*, pp. 1356–1365. PMLR, 2018.

[Wri15] Stephen J Wright. "Coordinate descent algorithms." *Mathematical programming*, **151**(1):3–34, 2015.

[YHF18] Haishan Ye, Zhichao Huang, Cong Fang, Chris Junchi Li, and Tong Zhang. "Hessian-aware zeroth-order optimization for black-box adversarial attack." *arXiv preprint arXiv:1812.11377*, 2018.

[YPO18] Penghang Yin, Minh Pham, Adam Oberman, and Stanley Osher. "Stochastic backward Euler: an implicit gradient descent algorithm for k-means clustering." *Journal of Scientific Computing*, **77**:1133–1146, 2018.

[ZBZ21] Jiaxin Zhang, Sirui Bi, and Guannan Zhang. "A directional Gaussian smoothing optimization method for computational inverse design in nanophotonics." *Materials & Design*, **197**:109213, 2021.

[Zhu20] Jingyi Zhu. "Hessian Inverse Approximation as Covariance for Random Perturbation in Black-Box Problems." *arXiv preprint arXiv:2011.13166*, 2020.

[ZTL20] Jiaxin Zhang, Hoang Tran, Dan Lu, and Guannan Zhang. "A novel evolution strategy with directional gaussian smoothing for blackbox optimization." *arXiv preprint arXiv:2002.03001*, 2020.

[ZWS19] Jingyi Zhu, Long Wang, and James C Spall. "Efficient implementation of second-order stochastic approximation algorithms in high-dimensional problems." *IEEE Transactions on Neural Networks and Learning Systems*, **31**(8):3087–3099, 2019.