An improved randomized algorithm with noise level tuning for large-scale noisy unconstrained DFO problems

Morteza Kimiaei
Fakultät für Mathematik, Universität Wien
Oskar-Morgenstern-Platz 1, A-1090 Wien, Austria
email: kimiaeim83@univie.ac.at
WWW: `http://www.mat.univie.ac.at/~kimiaei`

Abstract. In this paper, a new randomized solver (called `VRDFON`) for noisy unconstrained derivative-free optimization (DFO) problems is discussed. Complexity result in the presence of noise for nonconvex functions is studied. Two effective ingredients of `VRDFON` are an improved derivative-free line search algorithm with many heuristic enhancements and quadratic models in adaptively determined subspaces. Numerical results show that, on the large scale unconstrained `CUTEst` test problems contaminated by the absolute uniform noise, `VRDFON` is competitive with state-of-the-art DFO solvers.

Keywords. Noisy derivative-free optimization, heuristic optimization, randomized line search method, complexity bound, sufficient decrease

2000 AMS Subject Classification: 90C15; 90C30; 90C56.

January 29, 2024

1 Introduction

We consider the problem of finding a minimizer of the unconstrained derivative-free optimization (DFO) problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

thoroughly discussed in the books by Audet & Hare [1] and Conn et al. [3]. Here the smooth real-valued function $f:\mathbb{R}^n\to\mathbb{R}$ is known only by a noisy oracle which, for a given $x\in\mathbb{R}^n$, gives an approximation $\tilde{f}(x)$ to the exact function value $f(x)$, contaminated by the noise $\tilde{f}(x)-f(x)$. This problem is called the **noisy DFO problem**. We denote by $g(x)$ the unknown exact gradient vector of f at x and by $\tilde{g}(x)$ its approximation. The algorithm does not use knowledge of g, the Lipschitz constants of f, the structure of f, or the statistical properties of noise. Noise may be deterministic (caused by modelling, truncation, and/or discretization errors) or stochastic (caused by inaccurate measurements or rounding errors).

There are many DFO methods for solving noisy DFO problems of the form (1) (see the survey paper by Larson et al. [4]):

• **Model-based methods** approximate $\tilde{f}$ at each trial point by an approximate quadratic model through fitting or interpolation, find an approximate solution of this model restricted to a region around the trial point to avoid large steps, and only accept trial points with low inexact function values.

• **Line search methods** perform extrapolation steps along random or coordinate directions or their opposite directions with the goal of accepting trial points with low inexact function values by using a line search condition.

• **Direct search methods** search along coordinate directions, directions from a fixed poll set, or random directions, using a decrease condition, like the line search condition, only to accept points with low inexact function values.

• **Matrix adaptation evolution strategy methods** repeatedly generate a finite number of individuals, select some individuals to generate parents, and choose a new mean for the distribution.

We propose a new randomized algorithm for noisy unconstrained DFO problems – called **Vienna noisy randomized derivative-free optimization** (`VRDFON`). Following the classifications of Larson et al. [4] and Rios & Sahinidis [5], `VRDFON` is a local randomized model-based line search-based solver. This solver is an improvement of the noiseless `VRBBO` solver (Kimiaei & Neumaier [6]) to handle the variability of noise intensity.

`VRDFON` repeatedly performs `DS`, a decrease search, using `MLS`, a multi-line search algorithm. `MLS` is likely to reduce inexact function values, reaching regions close to an approximate stationary point, and finally finding an approximate stationary point. This can be done by performing `EP`, an extrapolation step, along a finite number of random directions or their opposite directions, using a line search condition to accept points with low inexact function values.

An implemented version of `VRDFON` uses the following new features:

• New heuristics are used to find and update step sizes in an implemented version of `MLS`, so that step sizes are neither too small nor too large to avoid line search failure.

• Step sizes of `MLS` are heuristically changed at the end of an implemented version of `DS` if these step sizes are too small. The goal is to avoid getting stuck before an approximate stationary point is found.

• Surrogate quadratic models are constructed in adaptively determined subspaces that can handle medium and large scale problems.

• Several new directions (random approximate coordinate, perturbed random, and improved trust region) are generated so that an implemented version of `MLS` can be performed not only along random scaled directions required to achieve complexity results, but also along these new directions to avoid large steps, which is a source of line search failure.

Section 2 discusses some concepts and assumptions required to obtain complexity results for DFO methods. Then, Section 3 explains the `VRDFON` algorithm and its subalgorithms (`EP`, `MLS`, and `DS`). In Section 4, complexity bound on the number of function evaluations

used by VRDFON is found for the nonconvex case and a probabilistic bound on the unknown gradient norm for one of the points evaluated by VRDFON is found for all cases (nonconvex, convex, and strongly convex). Section 5 provides a comparison between VRDFON and several state-of-the-art DFO solvers on the large and very large scale noisy problems obtained from the noiseless unconstrained CUTEst test problems from the collection of Gould et al. [7]. In Section 6, it is shown that how VRDFON finds an approximate stationary point of a real-life problem. Our findings on the features of VRDFON from numerical results are summarized in Section 7.

2 Preliminaries

In this section, we present a list of known DFO solvers, state the assumptions required to obtain the complexity results of DFO methods and summarize some known limit accuracy and complexity results.

Table 1 includes a list of deterministic or randomized DFO solvers which use at least one of model-based, line search-based, direct search, and evolution strategy algorithms. By comparing these solvers, we can see the quality of a new composite algorithm compared to any single algorithm that forms it, or to other solvers that use a single or composite algorithm. This helps us know how to construct a new composite algorithm efficiently or robustly; for example, NOMAD and MCS use model-based direct search algorithms that are more robust and efficient than their model-free versions.

For DFO problems, a point satisfying

$$f(x) \leq \sup\Big\{f(y) \mid y \in \mathbb{R}^n, \quad f(y) \leq f(x^0), \quad \|g(y)\| \leq \varepsilon\Big\}, \tag{2}$$

where $\|.\|$ is the Euclidean norm, is called an ε-**approximate stationary point** for the DFO problem (1). The goal of a DFO solver is to find such a stationary point. Throughout the paper $\varepsilon > 0$ is a minimum threshold for the unknown gradient norm in the noiseless case.

To analyze the limit accuracy and the complexity of our algorithm (defined below) for solving (1), we assume, like other DFO methods, see, e.g., Bergou et al. [26], Gratton et al. [19], and Kimiaei & Neumaier [6], that
(A1) the function f is continuously differentiable on $\mathbb{R}^n$, and its gradient is Lipschitz continuous with Lipschitz constant L,
(A2) the level set $\mathcal{L}(x^0) := \{x \in \mathbb{R}^n \mid f(x) \leq f(x^0)\}$ of f at x^0 is compact, and
(A3) the approximation $\tilde{f}(x)$ of f at $x \in \mathbb{R}^n$ satisfies

$$|\tilde{f}(x) - f(x)| \leq \omega. \tag{3}$$

(A2) implies that

$$\widehat{f} := \inf\{f(x) \mid x \in \mathbb{R}^n\} = f(\widehat{x}) > -\infty \tag{4}$$

for any global minimizer $\widehat{x}$ of (1).

solver	model-based	line search	direct search	evolution strategy	deterministic	randomized	Reference
BCDFO	+	−	−	−	+	−	Gratton et al. [8]
UOBYQA	+	−	−	−	+	−	Powell [9]
NEWUOA	+	−	−	−	+	−	Powell [10]
SNOBFIT	+	−	−	−	+	−	Huyer & Neumaier [11]
GRID	+	−	−	−	+	−	Elster & Neumaier [12]
MCS	+	−	+	−	+	−	Huyer & Neumaier [13]
NOMAD	+	−	+	−	+	−	[14, 15, 16, 17]
VRDFON	+	+	−	−	+	+	present paper
subUOBYQA	+	−	−	−	+	−	present paper
subNEWUOA	+	−	−	−	+	−	present paper
VRBBO	−	+	−	−	+	+	Kimiaei & Neumaier [6]
SDBOX	−	+	−	−	+	−	Lucidi & Sciandrone [18]
FMINUNC	−	+	−	−	+	−	Matlab Optimization Toolbox
DSPFD	−	−	+	−	−	+	Gratton et al. [19]
BFO	−	−	+	−	+	+	Porcelli & Toint [20]
NMSMAX	−	−	+	−	+	−	Higham [21]
subNMSMAX	−	−	+	−	+	−	present paper
CMAES	−	−	−	+	−	+	Auger & Hansen [22]
LMMAES	−	−	−	+	−	+	Loshchilov et al. [23]
fMAES	−	−	−	+	−	+	Beyer [24]
BiPopMAES	−	−	−	+	−	+	Beyer & Sendhoff [25]

Table 1: A list of DFO solvers needed in this paper. subUOBYQA, subNEWUOA, and subNMSMAX are, respectively, UOBYQA, NEWUOA, and NMSMAX in random subspaces to handle problems in medium and high dimensions.

In the noiseless case $\omega = 0$, (A3) implies $\tilde{f} = f$. Larson et al. [4, Table 8.1] and Kimiaei & Neumaier [6, Tables 1–3] summarize the known results on complexity and corresponding references: To satisfy (2) (under the assumptions (A1) and (A2)), one needs
• $\mathcal{O}(\varepsilon^{-2})$ function evaluations for the general case,
• $\mathcal{O}(\varepsilon^{-1})$ function evaluations for the convex case,
• $\mathcal{O}(\log \varepsilon^{-1})$ function evaluations for the strongly convex case. In all cases, the factors are ignored. Randomized algorithms typically have complexity bounds that are a factor n better than those of deterministic algorithms, see [27].

For noisy DFO problems in the form (1) ($\omega > 0$), a **complexity bound** of an algorithm is an upper bound on the number of function evaluations to find an approximate stationary point x (unknown to us since L and $g(x)$ are unknown) near a local optimizer whose unknown exact gradient norm is below a given fixed threshold

$$\varepsilon_\omega := \mathcal{O}(L\sqrt{n\omega}) \tag{5}$$

($\omega > 0$ is unknown to us but appearing in our complexity bound) and whose function value $f(x)$ satisfies (2). Under the assumption (A3), we can only expect a gradient accuracy of at most ε_ω. Therefore we aim for an ε_ω-approximate stationary point of the noisy DFO problems (1) with $\omega > 0$.

In the presence of noise, the limit accuracy and complexity results of some algorithms have been investigated by several researchers. We only summarize the results of line search based algorithms; cf. Table 2. Other useful references for complexity results of stochastic DFO methods are Chen [28], Dzahini [29], and Blanchet et al. [30].

type of noise	theoretical result
deterministic	nonconvex: $\|g\| = \mathcal{O}(\sqrt{\omega})$
assumptions:	(A1)–(A3)
reference	Lucidi & Sciandrone [18] and Elster & Neumaier [12]
deterministic	strongly convex: $f - \widehat{f} = \mathcal{O}(\omega)$
assumptions:	(A1)–(A3)
reference:	Berahas et al. [31]
stochastic	nonconvex: $\mathcal{O}(\varepsilon^{-2})$ with $\mathbf{E}(\|g\|) \le \varepsilon$
	convex: $\mathcal{O}(\varepsilon^{-1})$ with $\mathbf{E}(\|g\|) \le \varepsilon$, $\mathbf{E}(f - \widehat{f}) \le \varepsilon$
	strongly convex: $\mathcal{O}(\log \varepsilon^{-1})$ with $\mathbf{E}(\|g\|) \le \varepsilon$, $\mathbf{E}(f - \widehat{f}) \le \varepsilon$
assumptions:	(A1)–(A3) and norm condition:
	$\|\tilde{g}(x) - g(x)\| \le \theta \|g(x)\|$ for some $0 < \theta < 1$
reference:	Berahas et al. [32]

Table 2: Known limit accuracy and complexity noisy DFO methods regardless of L and n. As stated in the introduction, $\tilde{g}(x)$ stands for the estimated gradient at x and $\widehat{f}$ is the function value at any global minimizer $\widehat{x}$. Here $\mathbf{E}$ denotes the expectation value.

The objective function f is convex if condition

$$f(y) \geq f(x) + g(x)^T(y-x) + \frac{1}{2}\sigma\|y-x\|^2 \quad \text{for } x, y \in \mathbb{R}^n \tag{6}$$

holds for $\sigma = 0$ and it is strongly convex if (6) holds for $\sigma > 0$.

The following well-known result (e.g., see [26, 6]) gives a bound for $f(\tilde{x}) - \widehat{f}$ in the convex and strongly convex cases and bounds for $\|\tilde{x} - \widehat{x}\|$ in the strongly convex case. Here $\widehat{f} = f(\widehat{x})$ is from (4) and $\tilde{x}$ satisfies

$$\|g(\tilde{x})\| = \mathcal{O}(\varepsilon). \tag{7}$$

2.1 Proposition. *Under (A1) and (A2), if* (7) *holds, then:*
(i) In the convex and strongly convex cases

$$f(\tilde{x}) - \widehat{f} = \begin{cases} \mathcal{O}(\varepsilon) & \textit{convex case,} \\ \mathcal{O}(\varepsilon^2) & \textit{strongly convex case,} \end{cases} \tag{8}$$

where r_0 is given by

$$r_0 := \sup\left\{\|x - \widehat{x}\| \mid x \in \mathbb{R}^n, \quad f(x) \leq f(x^0)\right\} < \infty, \tag{9}$$

where x^0 is the initial point.
(ii) In the strongly convex case

$$\|\tilde{x} - \widehat{x}\| = \mathcal{O}(\varepsilon). \tag{10}$$

Here $\widehat{f}$ is finite by (A1) and (A2).

Proof. (i) Now assume that f is convex. Then

$$\widehat{f} \geq f(x) + g(x)^T(\widehat{x} - x) \quad \text{for all } x \in \mathbb{R}^n.$$

We now conclude that for one $\tilde{x}$ of the old best points the condition

$$f(\tilde{x}) - \widehat{f} \leq g(\tilde{x})^T(\tilde{x} - \widehat{x}) \leq \|g(\tilde{x})\|\|\tilde{x} - \widehat{x}\| \overset{(7),(9)}{=} r_0\mathcal{O}(\varepsilon) = \mathcal{O}(\varepsilon)$$

holds.

Finally, assume that f is strongly convex. If x is assumed to be fixed, the right-hand side of (6) is a convex quadratic function with respect to y whose gradient in the components vanishes at $y_i = x_i - \sigma^{-1}g_i(x)$ for $i = 1, \ldots, n$, leading to

$$f(y) \geq f(x) - \frac{1}{2\sigma}\|g(x)\|^2.$$

We now obtain

$$f(\tilde{x}) - \widehat{f} \leq \frac{1}{2\sigma}\|g(\tilde{x})\|^2 \overset{(7)}{=} \mathcal{O}(\varepsilon^2).$$

(ii) Substituting $\widehat{x}$ for x and $\tilde{x}$ for y into (6), we get $f(\tilde{x}) \geq f(\widehat{x}) + \frac{\sigma}{2}\|\tilde{x} - \widehat{x}\|^2$ such that (i) leads to the fact that the condition

$$\|\tilde{x} - \widehat{x}\|^2 \leq \frac{2}{\sigma}\left(f(\tilde{x}) - \widehat{f}\right) \leq \frac{1}{\sigma^2}\|g(\tilde{x})\|^2 \overset{(7)}{=} \mathcal{O}(\varepsilon^2)$$

holds. □

3 Proposed VRDFON algorithm

In this section we describe the VRDFON algorithm and how it works. Until an approximate stationary point is found, VRDFON repeatedly calls a decrease search (DS), which has a finite number of calls to a multi line search (MLS), using an extrapolation step (EP) to leave regions close to saddle point or maximizer. EP uses a line search condition (defined in Subsection 3.3) to accept points with low inexact function values in regions close to an approximate stationary point.

An iteration of EP is called **successful** if a reduction of $\tilde{f}$ along a given direction is found and **unsuccessful** otherwise, while an iteration of MLS is called **successful** if EP has a successful iteration either along a given direction or its opposite direction is found and **unsuccessful** otherwise. An iteration of DS is called **successful** if MLS has at least one successful iteration. An iteration of VRDFON is called **successful** if DS has at least one successful iteration.

We denote by x_{best} the **best point found so far** and by $\tilde{f}_{\text{best}} := \tilde{f}(x_{\text{best}})$ the **best inexact function value so far**, i.e., the point with the lowest inexact function value found by DS in the final iteration. To simplify our algorithms, all tuning parameters are given once in line 1 of VRDFON and are not mentioned as input for the other algorithms.

In line 3, VRDFON computes the inexact function value $\tilde{f}(x^0)$ at the initial point x^0 and then initializes the initial best point x_{best}, its inexact function value $\tilde{f}_{\text{best}}$, and the initial step size $\delta := \delta_{\max} > 0$, which is a tuning parameter. In each iteration, in line 5, VRDFON calls DS to find a possible reduction of $\tilde{f}$. Once δ is below a minimum threshold $0 < \delta_{\min} < 1$, which is a tuning parameter, in line 6, VRDFON terminates. Otherwise, if the Boolean variable successDS is false (the current iteration of VRDFON is unsuccessful), in line 7, VRDFON reduces δ by a factor of $Q > 1$.

If the condition $\delta \leq \delta_{\min}$ is satisfied, in theory VRDFON finds an ε_ω-approximate stationary point (defined in Section 2) x_{best} that satisfies $\|g(x_{\text{best}})\| \leq \varepsilon_\omega$ for a threshold ε_ω; this cannot be checked numerically since the true gradient is not available.

Algorithm 1 `VRDFON`, **a randomized method for noisy DFO problems**

Tuning parameters: $Q > 1$ (factor for reducing δ), $0 < \gamma_{\mathrm{rd}} < 1$ (parameter for scaling random directions), $0 < \gamma < 1$ (parameter for line search), $\gamma_e > 1$ (factor for updating step size inside `MLS`), $0 < \delta_{\min} \leq 1$ (minimum threshold for δ), $\delta_{\max} > \delta_{\min} > 0$ (initial value for δ), $0 < \eta < \frac{1}{2}$ (parameter for R), $R \geq 2$ (number of random direction in each `MLS`), $T_0 \geq 1$ (number of calls to `MLS` by `DS`).
Input: $x^0 \in \mathbb{R}^n$

Compute $\tilde{f}(x^0)$ and set $x_{\mathrm{best}} := x^0$, $\tilde{f}_{\mathrm{best}} := \tilde{f}(x^0)$, and $\delta := \delta_{\max}$.
for $k = 1, 2, \ldots$ **do**
  $[x_{\mathrm{best}},\ \tilde{f}_{\mathrm{best}},\ \texttt{successDS}] = \texttt{DS}(x_{\mathrm{best}},\ \tilde{f}_{\mathrm{best}},\ \delta)$.
  if $\delta \leq \delta_{\min}$; **return**; **end if**
  if $\sim$`successDS` **then** $\delta := \delta/Q$; **end if**
end for
Output: x_{best} and $\tilde{f}_{\mathrm{best}}$

Since `VRDFON` has $1 \leq K < \infty$ calls to `DS` and `DS` has $1 \leq T_0 < \infty$ calls to `MLS`, `VRDFON` has KT_0 calls to `MLS`. For any given $0 < \eta < \frac{1}{2}$, `MLS` uses

$$R = \left\lceil (T_0 K)^{-1} \log_2 \eta^{-1} \right\rceil \geq 2 \tag{11}$$

random directions in each iteration of `MLS`.

3.1 `DS`, a decrease search

This subsection discusses `DS` and how it works. The goal of `DS` is to find a possible decrease in $\tilde{f}$ by performing `MLS`.

Algorithm 2 `DS`, **a decrease search**

function $[x_{\mathrm{best}},\ \tilde{f}_{\mathrm{best}},\ \texttt{successDS}] = \texttt{DS}(x_{\mathrm{best}},\ \tilde{f}_{\mathrm{best}},\ \delta)$

1: Set `successDS` $:= 0$, $y_{\mathrm{best}} := x_{\mathrm{best}}$, and $\tilde{f}(y_{\mathrm{best}}) := \tilde{f}_{\mathrm{best}}$.
2: **for** $t = 1, \ldots, T_0$ **do**,
3: $[y_{\mathrm{best}},\ \tilde{f}(y_{\mathrm{best}}),\ \texttt{successMLS}] = \texttt{MLS}(y_{\mathrm{best}},\ \tilde{f}(y_{\mathrm{best}}),\ \delta)$.
4: **if** `successMLS` **then** `successDS` $:= 1$; **end if**
5: **end for**
6: **if** `successDS` ▷ at least one reduction of $\tilde{f}$ was found
7: $x_{\mathrm{best}} := y_{\mathrm{best}}$ and $\tilde{f}_{\mathrm{best}} := \tilde{f}(y_{\mathrm{best}})$.
8: **end if**

In line 1 of `DS`, `DS` initially evaluates the Boolean variable `successDS` as false and initializes its initial best point $y_{\mathrm{best}} = x_{\mathrm{best}}$ and the corresponding function value $\tilde{f}(y_{\mathrm{best}}) = \tilde{f}_{\mathrm{best}}$. Subsequently, `DS` has T_0 calls to `MLS` to find a possible reduction of $\tilde{f}$. If a reduction of $\tilde{f}$ is found by `MLS` (`successMLS` $= 1$), in line 4, `DS` evaluates `successDS` as true. After the

termination of the for loop, if `successDS` is true, at least one reduction of $\tilde{f}$ is found by `MLS` and hence, in line 7, `DS` chooses the new best point and its function value found by `MLS`. Otherwise, `DS` has found no decease in $\tilde{f}$.

3.2 `MLS`, a multi line search

This subsection discusses the main component `MLS` of `DS`, whose goal is to perform extrapolation along scaled random directions (discussed below) or their opposite directions and enter or move along a valley to achieve an approximate stationary point.

Algorithm 3 `MLS`, **a multi line search**

function $[y_{\text{best}},\ \tilde{f}(y_{\text{best}}),\ \texttt{successMLS}] = \texttt{MLS}(y_{\text{best}},\ \tilde{f}(y_{\text{best}}), \delta)$

1: Set $\alpha := \delta$, $z_{\text{best}} := y_{\text{best}}$, $\tilde{f}(z_{\text{best}}) := \tilde{f}(y_{\text{best}})$, and `successMLS` $:= 0$.
2: **for** $r = 1, \ldots, R$ **do**
3: Compute the scaled random direction p.
4: $[z_{\text{best}}, \tilde{f}(z_{\text{best}}), z_{\text{trial}}, \tilde{f}_{\text{trial}}, \texttt{success}] = \texttt{EP}(z_{\text{best}}, \tilde{f}(z_{\text{best}}), \alpha, p)$.
5: **if** $\sim$ `success`, **then**
6: $p := -p$.
7: $[z_{\text{best}}, \tilde{f}(z_{\text{best}}), z_{\text{trial}}, \tilde{f}_{\text{trial}}, \texttt{success}] = \texttt{EP}(z_{\text{best}}, \tilde{f}(z_{\text{best}}), \alpha, p)$.
8: **if** $\sim$ `success`, **then** ▷ the rth iteration is unsuccessful
9: $z = z_- := z_{\text{trial}}$, $\tilde{f}(z) = \tilde{f}(z_-) := \tilde{f}_{\text{trial}}$.
10: **if** $r < R$ **then** $\alpha := \alpha/\gamma_e$; **end if**
11: **else** ▷ the rth iteration is successful
12: $z = z_- := z_{\text{best}}$, $\tilde{f}(z) = \tilde{f}(z_-) := \tilde{f}(z_{\text{best}})$.
13: **end if**
14: **else** ▷ the rth iteration is successful
15: $z = z_+ := z_{\text{best}}$, $\tilde{f}(z) = \tilde{f}(z_+) := \tilde{f}(z_{\text{best}})$.
16: **end if**
17: **if** `success`, **then** `successMLS` $= 1$; **end if**
18: **end for**
19: **if** `successMLS` ▷ at least one reduction of $\tilde{f}$ was found
20: $y_{\text{best}} := z_{\text{best}}$ and $\tilde{f}(y_{\text{best}}) := \tilde{f}(z_{\text{best}})$.
21: **end if**

We define a standard random direction as a random direction p drawn uniformly **i.i.d.** (independent and identically distributed) in $[-\frac{1}{2}, \frac{1}{2}]^n$. A **scaled random direction** is a standard random direction p scaled by $\gamma_{\text{rd}}/\|p\|$, where $0 < \gamma_{\text{rd}} < 1$ is a tiny tuning parameter, resulting in $\|p\| = \gamma_{\text{rd}}$. The scaling of the direction p by γ_{rd} is the same as the scaling of the direction p by δ in [6, (17)]. Therefore, our scaled random direction is the scaled random direction of [6, (17)].

In each iteration r, `MLS` evaluates

$$z := \begin{cases} z_+ & \texttt{MLS} \text{ is performed along } p, \\ z_- & \texttt{MLS} \text{ is performed along } -p \end{cases} \tag{12}$$

and computes $\tilde{f}(z)$ in lines 9, 12, 15. It chooses the initial extrapolation step size $\alpha = \delta$, the initial best point z_{best}, its inexact function value $\tilde{f}(z_{\text{best}})$, and the Boolean variable `successMLS` as false (no try to find possible reductions of $\tilde{f}$). `MLS` includes a for loop. In each iteration r, in line 3, `MLS` computes the scaled random direction p. Then, in lines 4 and 7, `EP` performs extrapolation along at least one of $\pm p$. If a reduction of $\tilde{f}$ is found (i.e., the Boolean variable `success` is true), `EP` finds a reduction of $\tilde{f}$ and generates z in either line 12 or line 15. Otherwise, the trial point $z_{\text{trial}} = z_{\text{best}} + \alpha p$ is chosen as z in line 9 and the new step size α is obtained by reducing α by a factor of γ_e (which is a tuning parameter) in line 10. Once `EP` finds a reduction of $\tilde{f}$, `MLS` evaluates the Boolean variable `successMLS` as true in line 17. After the termination of the for loop, in line 19, if the Boolean variable `successMLS` is true (i.e., in either line 4 or line 7 `EP` has found at least one reduction of $\tilde{f}$), `MLS` chooses the last point accepted by `EP` and its function value as the new best point and its function value.

3.3 `EP`, an extrapolation step

This subsection discusses the main component `EP` of `MLS`, whose goal is to discard points located in regions near a saddle point or maximizer and accept points located in regions near an approximate minimizer. This can be done by a line search condition.

A sufficient reduction of $\tilde{f}$ means that the **line search condition**

$$\tilde{f}(z_{\text{best}}) - \tilde{f}(z_{\text{best}} + \alpha p) > \gamma \alpha^2$$

holds, where $0 < \gamma < 1$. We say that a **γ-reduction** is found along the search direction p.

`EP` tries to find a possible reduction of $\tilde{f}$. As long as the line search condition holds, extrapolation step sizes are expanded by the tuning parameter $\gamma_e > 1$ and the new trial points $z_{\text{trial}} = z_{\text{best}} + \alpha p$ and their inexact function values $\tilde{f}_{\text{trial}} = \tilde{f}(z_{\text{trial}})$ are computed. Once, no more decrease in $\tilde{f}$ in the current iteration r is found, the line search condition is violated and `EP` ends.

Since in theory f is assumed to be bounded below, `EP` is terminated when the line search condition is violated. If the line search condition is violated in the first iteration, `EP` fails and the Boolean variable `success` is evaluated as false. Otherwise, `EP` has at least one reduction of $\tilde{f}$, but not in $\tilde{f}$ of the last trial point. In this case, `EP` is terminated. After the termination of `EP`, `VRDFON` is also terminated.

In practice, if `VRDFON` is applied to solve DFO problems with unbounded below objective function, `EP` and `VRDFON` end when $\tilde{f}_{\text{trial}}$ reaches -10^{12}.

After the termination of the while loop in `EP`, if `success` is true, the line search condition has been violated in the last trial point and the penultimate point has been accepted as the new best point in line 7, whose inexact function value has already been stored in $\tilde{f}_{\text{penult}}$ in line 3.

Algorithm 4 `EP`, **an extrapolation step**

function $[z_{\text{best}}, \tilde{f}(z_{\text{best}}), z_{\text{trial}}, \tilde{f}_{\text{trial}}, \texttt{success}] = \texttt{EP}(z_{\text{best}}, \tilde{f}(z_{\text{best}}), \alpha, p)$

Compute $z_{\text{trial}} := z_{\text{best}} + \alpha p$ and $\tilde{f}_{\text{trial}} := \tilde{f}(z_{\text{trial}})$. Then set $\texttt{success} = 0$.
while $\tilde{f}(z_{\text{best}}) - \tilde{f}_{\text{trial}} > \gamma\alpha^2$ **do**
    $\texttt{success} := 1$, $\tilde{f}_{\text{penult}} := \tilde{f}_{\text{trial}}$, and $\alpha := \gamma_e \alpha$.
    Compute $z_{\text{trial}} := z_{\text{best}} + \alpha p$ and $\tilde{f}_{\text{trial}} := \tilde{f}(z_{\text{trial}})$.
end while
if `success` **then** ▷ update the best point
    $\tilde{\alpha} := \alpha/\gamma_e$, $\tilde{f}(z_{\text{best}}) := \tilde{f}_{\text{penult}}$, and $z_{\text{best}} := z_{\text{best}} + \tilde{\alpha} p$.
end if

4 Accuracy and complexity analysis of `VRDFON`

In this section, we assume that the following assumptions holds.

4.1 Assumptions. *Assume that (A1)–(A3) hold and* $\delta_{\max} > \delta_{\min} > 0$, $Q > 1$, $0 < \gamma_{\text{rd}} < 1$, $\gamma_e > 1$, $0 < \gamma < 1$, $\overline{\tau} > \underline{\tau} > 0$, $\varepsilon_\omega > 0$, *defined by* (5), *and*

$$\underline{\tau}\sqrt{\omega L^{-1}} \leq \delta_{\min} \leq \overline{\tau}\sqrt{\omega L^{-1}}. \tag{13}$$

The tuning parameter $\gamma_e > 1$ *and* δ *satisfy the condition*

$$\gamma_e^{1-R}\delta \geq \alpha_{\min} \tag{14}$$

for a fixed minimum threshold $0 < \alpha_{\min} < \infty$ *and a fixed small positive integer* R.

Under Assumptions 4.1, we prove that, with a given probability arbitrarily close to 1, `VRDFON` terminates after at most $\mathcal{O}(nL^3\varepsilon_\omega^{-2})$ function evaluations in the nonconvex case and finds a point x satisfying (2) (see Corollary 4.6 below). Here ε_ω is from (5) and such a point is unknown to us because gradients and Lipschitz constants are unknown. The order of ω in our bound is the same as in Berahas et al. [32]. In contrast to the method of Berahas et al. [32], which uses the norm condition defined in Table 2, our line search does not use the approximate directional derivative $\tilde{g}^T p$ in the line search condition, but $\gamma\alpha^2$ with $0 < \gamma < 1$ because the estimation of the gradient may be inaccurate in the presence of high noise, leading to failure of the line search algorithm. However, we estimate the gradient to generate different heuristic directions in Section 5.2. Therefore, we obtain our complexity bound regardless of the norm condition since the nature of the line search algorithms is different. Our bound is obtained with high probability, while the results of Berahas et al. [32] are valid in expectation.

4.1 Complexity bound for `VRDFON` in the nonconvex case

This section discusses complexity bound for `VRDFON` in the nonconvex case.

The first result (Proposition 4.4 below) provides complexity bound on the number of function evaluations of trial points with a reduction of $\tilde{f}$ evaluated by all extrapolations for the nonconvex case.

The second result (Proposition 4.5 below) provides complexity bound on the number of function evaluations of trial points without a reduction of $\tilde{f}$ evaluated by all extrapolations for the nonconvex case.

The third result (Theorem 4.6 below) provides complexity bound on the number of function evaluations of trial points evaluated by all extrapolations for the nonconvex case.

In the kth iteration of `VRDFON`, we denote by $S^k := S^k_{\text{VRDFON}}$ the index set of successful iterations of `VRDFON` and by $U^k := U^k_{\text{VRDFON}}$ the index set of unsuccessful iterations of it. After the termination of `VRDFON`, we define the two index sets of all unsuccessful and successful iterations of `VRDFON`, respectively, by

$$U^k \subseteq \overline{U} := \{j_1, j_2, \ldots, j_{K'}\} \text{ and } S^k \subseteq \overline{S} := \{i_1, i_2, \ldots, i_{K''}\}.$$

Here $K' = |\overline{U}|$ (defined by (15), below) and $K'' = |\overline{S}|$. As defined before (11), K is the number of calls to `DS` by `VRDFON` and therefore $K = K' + K'' < \infty$. Then, the index set of all iterations of `VRDFON` is formed as

$$\overline{U} \cup \overline{S} := \{k_1, k_2, \ldots, k_K\}.$$

4.2 Proposition. *Given the tuning parameter $Q > 1$ and a given threshold $0 < \delta_{\min} < 1$,* `VRDFON` *ends after at most*

$$K' := \left\lceil \frac{\log(\delta_{\max}/\delta_{\min})}{\log Q} \right\rceil \tag{15}$$

unsuccessful iterations. Moreover,

$$\delta_k^{-1} = \mathcal{O}(\delta_{\min}^{-1}) \text{ for } k \in \overline{U} \cup \overline{S}. \tag{16}$$

Proof. From the rule for updating δ_k in lines 3, 8, 10 of Algorithm 1 and since δ_k is unchanged for $k \in \overline{S}$, we obtain

$$\delta_k \geq Q^{-k}\delta_{\max} \geq \delta_{\min} \text{ for } k \in \overline{U} \cup \overline{S}.$$

Let $j_{K'}$ be the last unsuccessful iteration. Then,

$$\delta_{j_{K'}} = Q^{-K'}\delta_{\max} \leq \delta_{\min}.$$

This condition results in (15) and `VRDFON` ends after K' unsuccessful iterations. Hence, the condition

$$\delta_k \geq Q^{-K'}\delta_{\max} \text{ for } k \in \overline{U} \cup \overline{S}$$

results in that condition (16) is obtained from

$$\delta_k^{-1} = \mathcal{O}(Q^{K'}) \overset{(15)}{=} \mathcal{O}(\delta_{\min}^{-1}) \text{ for } k \in \overline{U} \cup \overline{S}.$$

□

We denote by I the index set of all trial points generated by all extrapolations whose inexact function values are reduced and by I^c the index set of other trial points generated by all extrapolations whose inexact function values cannot be reduced. Moreover, we denote these two sets by I_k and I_k^c, respectively, in the kth iterations of `VRDFON`.

4.3 Proposition. *Let $\{x^{k_\ell}\}_{\ell\geq 0}$ be the sequence generated by* `VRDFON`*. Moreover, for $\ell \in I \cup I^c$, let $f_{k_\ell} := f(x^{k_\ell})$, $\tilde{f}_{k_\ell} := \tilde{f}(x^{k_\ell})$, and $g_{k_\ell} := g(x^{k_\ell})$. If Assumptions 4.1 hold then:*
(i) The number of trial points of all extrapolations without reduction of $\tilde{f}$ is

$$|I^c| = \mathcal{O}(K'T_0R). \tag{17}$$

(ii) The sequence $\{f_{k_\ell}\}_{\ell\in I}$ satisfies

$$\tilde{f}_{k_\ell} - \tilde{f}_{k_{\ell-1}} > \overline{\gamma}\delta_{k_{\ell-1}}^2, \tag{18}$$

where $\overline{\gamma} = \gamma\gamma_e^{2(2-R)} > 0$.

Proof. (i) holds since `VRDFON` has K' unsuccessful iterations and T_0 calls to `MLS`, and `MLS` has R calls to `EP`.
(ii) Since no reduction of $\tilde{f}$ is found at the last points generated by all extrapolations, the corresponding indices are not in I. From the structures of Algorithms 1–4, for each $\ell \in I$, there are three indices $k_\ell \geq 0$, $r_\ell \in \{1, \ldots, R\}$ and $t_\ell \in \{1, \ldots, T_0\}$ satisfying

$$x^{k_{\ell-1}} = x_{\text{best}}^{k_{\ell-1}} = y_{\text{best}}^{t_{\ell-1}} = z_{\text{best}}^{r_{\ell-1}}, \quad x^{k_\ell} = x_{\text{best}}^{k_\ell} = y_{\text{best}}^{t_\ell} = z_{\text{best}}^{r_\ell} = z_{\text{best}}^{r_{\ell-1}} + \alpha_{r_\ell}p^{r_\ell}.$$

According to the rule for updating α in lines 1 and 10 of `MLS` and (14),

$$\alpha_{r_\ell} \geq \gamma_e^{1-R}\delta_{k_{\ell-1}}. \tag{19}$$

Using (19) and since k_ℓ is a successful iteration for `VRDFON`, we obtain

$$\tilde{f}_{k_\ell} - \tilde{f}_{k_{\ell-1}} = \tilde{f}_{k_{\ell-1}} - \tilde{f}_{k_\ell} > \gamma\alpha_{r_\ell}^2 \geq \gamma(\gamma_e^{1-R}\delta_{k_{\ell-1}})^2 = \overline{\gamma}\delta_{k_{\ell-1}}^2.$$

□

4.4 Proposition. *If Assumptions 4.1 hold, the number N of function evaluations those trial points with a reduction of $\tilde{f}$ evaluated by all extrapolations is at most $\mathcal{O}\big(L\omega^{-1}\big)$ in the nonconvex case.*

Proof. Let $\{x^{k_\ell}\}_{\ell\geq 0}$ be the sequence generated by `VRDFON`. The result

$$N = |I| = \mathcal{O}\big(\delta_{\min}^{-2}\big) = \mathcal{O}\big(L\omega^{-1}\big)$$

is obtained from

$$\overline{\gamma}|I| \;\leq\; \sum_{\ell\in I}\overline{\gamma} \overset{(18)}{\leq} \sum_{k\in\overline{S}}\sum_{\ell\in I_k}\frac{\tilde{f}_{k_{\ell-1}}-\tilde{f}_{k_\ell}}{\delta^2_{k_{\ell-1}}} \overset{(16)}{\leq} \delta^{-2}_{\min}\sum_{k\in\overline{S}}\sum_{\ell\in I_k}(\tilde{f}_{k_{\ell-1}}-\tilde{f}_{k_\ell}) \overset{(3)}{\leq} \delta^{-2}_{\min}(f_0-\widehat{f}+2\omega).$$

□

4.5 Proposition. *Under Assumptions 4.1, the number $\overline{N}$ of function evaluations of all last points generated by all extrapolations is at most $\mathcal{O}(T_0R\log\omega^{-1})$.*

Proof. Since `VRDFON` has at most K' unsuccessful iterations with at most $2T_0R$ calls to `EP`,

$$\overline{N} \overset{(17)}{=} \mathcal{O}(|I^c|) = \mathcal{O}(T_0RK') \overset{(15)}{=} \mathcal{O}(T_0R\log\delta^{-1}_{\min}) = \mathcal{O}(T_0R\log\omega^{-1}).$$

□

The following result gives complexity for `VRDFON` for the nonconvex case.

4.6 Corollary. *Under Assumptions 4.1, the total number N_{total} of function evaluations by* `VRDFON` *is at most $\mathcal{O}(nL^3\varepsilon_\omega^{-2})$ function evaluations.*

Proof. Let $N_{\text{total}} = 1+N+\overline{N}$. Here the first term of N_{total} is for computing $\tilde{f}(x^0)$. From Propositions 4.4 and 4.5, the results are obtained from

$$N_{\text{total}} = 1+\mathcal{O}(L\omega^{-1})+\mathcal{O}(RT_0\log\omega^{-1}) = \mathcal{O}(L\omega^{-1}) \overset{(5)}{=} \mathcal{O}(nL^3\varepsilon_\omega^{-2}).$$

□

4.2 Bound on g^Tp

For $z_\pm := z_{\text{best}} \pm \alpha p$, (A1) results in

$$\pm\alpha g(z_{\text{best}})^Tp - \frac{1}{2}L\alpha^2\|p\|^2 \leq f(z_\pm)-f(z_{\text{best}}) \leq \pm\alpha g(z_{\text{best}})^Tp + \frac{1}{2}L\alpha^2\|p\|^2. \tag{20}$$

We use (20) to generalize Proposition 2 of [6] to the noisy case. The following result shows that if the rth iteration of `MLS` is unsuccessful, a useful bound for the directional derivative can be found.

4.7 Proposition. *If Assumptions 4.1 hold, then for all $z\in\mathbb{R}^n$ evaluated by (12) and $p\in\mathbb{R}^n$ at least one of the following holds:*
(i) $\tilde{f}(z_+) < \tilde{f}(z_{\text{best}}) - \gamma\alpha^2$, where $z_+ = z_{\text{best}}+\alpha p$.

(ii) $\tilde{f}(z_+) > \tilde{f}(z_{\text{best}}) + \gamma\alpha^2$ *and* $\tilde{f}(z_-) < \tilde{f}(z_{\text{best}}) - \gamma\alpha^2$*, where* $z_- = z_{\text{best}} - \alpha p$*.*
(iii) $|g(z_{\text{best}})^T p| \leq \gamma\alpha + 2\omega/\alpha + \frac{1}{2}L\alpha\|p\|^2$*.*
Here, z_{best} *is the old best point found by* `MLS`*.*

Proof. We assume that (iii) is violated. Then, for both signs, we have

$$\pm\alpha g(z_{\text{best}})^T p + \frac{1}{2}L\alpha^2\|p\|^2 + 2\omega < -\gamma\alpha^2. \tag{21}$$

We distinguish two cases:
Case 1. If $g(z_{\text{best}})^T p \leq 0$, then

$$\begin{aligned} \tilde{f}(z_+) - \tilde{f}(z_{\text{best}}) &\overset{(3)}{\leq} f(z_+) - f(z_{\text{best}}) + 2\omega \\ &\overset{(20)}{\leq} \alpha g(z_{\text{best}})^T p + \frac{1}{2}L\alpha^2\|p\|^2 + 2\omega \overset{(21)}{<} -\gamma\alpha^2 \end{aligned} \tag{22}$$

and (i) must hold.
Case 2. If $g(z_{\text{best}})^T p \geq 0$, then

$$\begin{aligned} \tilde{f}(z_-) - \tilde{f}(z_{\text{best}}) &\overset{(3)}{\leq} f(z_-) - f(z_{\text{best}}) + 2\omega \\ &\overset{(20)}{\leq} g(z_{\text{best}})^T(-\alpha p) + \frac{1}{2}L\alpha^2\|p\|^2 + 2\omega \overset{(21)}{<} -\gamma\alpha^2. \end{aligned} \tag{23}$$

Therefore, the second inequality in (ii) holds and the first half

$$\begin{aligned} \tilde{f}(z_+) - \tilde{f}(z_{\text{best}}) &\overset{(3)}{\geq} f(z_+) - f(z_{\text{best}}) - 2\omega \\ &\overset{(20)}{\geq} \alpha g(z_{\text{best}})^T p - \frac{1}{2}L\alpha^2\|p\|^2 - 2\omega \overset{(21)}{>} \gamma\alpha^2 \end{aligned}$$

is satisfied. Hence the first inequality in (ii) holds. □

4.3 Bounds on $\|g\|$

We write z^r_{best} for the point among all best points z_{best} encountered in iterations $\leq r$ for which $\|g(z_{\text{best}})\|$ is smallest. In practice, we do not know which of the best points is z^r_{best}, as gradients and Lipschitz constants are unknown.

For any $p, x \in \mathbb{R}^n$, we define

$$w(p, x) := \frac{\|g(x)\|\|p\|}{2|g(x)^T p|} \in \mathbb{R} \cup \{\infty\}. \tag{24}$$

w_{EP} denotes the value of $w(p, x)$ for the search direction p and the point x evaluated in `EP`. The point x is either the accepted point in each successful iteration of `EP` or the unaccepted

point in each unsuccessful iteration of EP. Then, in R calls to EP by MLS, w_{MLS} denotes the minimum value of all values $w(p,x)$ for the search directions p and the points x evaluated in EP. In T_0 calls to MLS by DS, w_{DS} denotes the minimum value of all values $w(p,x)$ for the search directions p and the points x evaluated in EP and, in K calls to DS by VRDFON, w_{VRDFON} denotes the minimum value of all values $w(p,x)$ for the search directions p and the points evaluated in EP.

In Subsections 4.3.1–4.3.4, we find upper bounds on $\|g\|$ for one of the points evaluated in EP, MLS, DS, and VRDFON. Moreover, in Subsection 4.3.5, we find a probabilistic bound on $\|g\|$ for one of the points evaluated in VRDFON.

4.3.1 A bound for the gradient of the result of EP

We first use Proposition 4.7 to get an upper bound on $\|g\|$ in each unsuccessful iteration of EP.

4.8 Lemma. *In each unsuccessful iteration,* EP *satisfies the condition*

$$\|g(z_{\mathrm{best}})\| \leq w_{\mathrm{EP}}\Gamma_+(\delta) \quad \textit{with } \Gamma_+(\delta) := L_+\delta + \gamma_e^{R-1}\frac{4\omega}{\gamma_{\mathrm{rd}}\delta}. \tag{25}$$

Proof. Let $\Delta_+(\alpha) := L_+\alpha + 4\omega/(\gamma_{\mathrm{rd}}\alpha)$. The extrapolation step size α_r in iteration r satisfies

$$\alpha_1 := \max_{r=1:R} \alpha_r = \delta > \alpha_R := \min_{r=1:R}\{\alpha_r\} = \gamma_e^{1-R}\delta$$

because of the rule for updating step size in line 3 of EP and since $\gamma_e > 1$ and $R \geq 2$ from (11). The fact that $\Delta_+(\alpha)$ for $\alpha > 0$ is a convex function results in

$$\begin{aligned} \Delta_+(\alpha) &\leq \max\{\Delta_+(\alpha_1), \Delta_+(\alpha_R)\} < L_+\alpha_1 + \frac{4\omega}{\gamma_{\mathrm{rd}}\alpha_R} \\ &= L_+\delta + \gamma_e^{R-1}\frac{4\omega}{\gamma_{\mathrm{rd}}\delta} = \Gamma_+(\delta). \end{aligned} \tag{26}$$

Since $\tilde{f}(z_\pm)$ does not decrease by more than $\gamma\alpha^2$ for each unsuccessful iteration of EP and $\|p\| = \gamma_{\mathrm{rd}}$ from the definition of the scaled random direction in Section 3.2, Proposition 4.7(iii) applies and gives

$$|g(z_{\mathrm{best}})^T p| \leq \gamma\alpha + 2\omega/\alpha + \frac{L}{2}\alpha\|p\|^2 = \gamma\alpha + 2\omega/\alpha + \frac{L}{2}\gamma_{\mathrm{rd}}^2\alpha.$$

Hence

$$\begin{aligned} \|g(z_{\mathrm{best}})\| &= 2\gamma_{\mathrm{rd}}^{-1} w_{\mathrm{EP}}|g(z_{\mathrm{best}})^T p| \leq w_{\mathrm{EP}}\left(L_+\alpha + \frac{4\omega}{\gamma_{\mathrm{rd}}\alpha}\right) \\ &\leq w_{\mathrm{EP}}\Delta_+(\alpha) \overset{(26)}{<} w_{\mathrm{EP}}\Gamma_+(\delta). \end{aligned}$$

□

4.9 Proposition. *Define*

$$L_- := L\gamma_{\rm rd} - 2\gamma/\gamma_{\rm rd} = \mathcal{O}(L), \quad L_+ := L\gamma_{\rm rd} + 2\gamma/\gamma_{\rm rd} = \mathcal{O}(L). \tag{27}$$

If Assumptions 4.1 hold, then, in each iteration of `EP`*,* `EP` *ends after at most*

$$E := \left\lceil \left(\log \sqrt{(f(x^0) - \widehat{f} + 2\omega)(\delta^{-2}\gamma^{-1})} \right) / \log \gamma_e \right\rceil + 1 \tag{28}$$

iterations, and satisfies the condition

$$\|g(z_{\rm best})\| \le w_{\rm EP}\Gamma(\delta) \quad \text{with } \Gamma(\delta) := \gamma_e^{\max(R,E)-1}\left(L_+\delta + \frac{4\omega}{\gamma_{\rm rd}\delta} \right). \tag{29}$$

Here R is from (11)*. Moreover, if the iteration is unsuccessful (*`success` *is false),* `EP` *ends after one iteration ($E = 1$).*

Proof. Put $z_\pm := z_{\rm best} \pm \alpha p$. We first show that `EP` ends with at most E iterations. From (4), the function is bounded below and therefore extrapolation must be terminated after at most j iterations. Since in extrapolation α is expanded by $\gamma_e > 1$, we have

$$\tilde{f}(z_{\rm best}) - \tilde{f}(z_{\rm best} + \gamma_e^{j-1}\delta) \ge \gamma\gamma_e^{2(j-1)}\delta^2,$$

so that

$$\gamma_e^{2(j-1)} \le \frac{\tilde{f}(z_{\rm best}) - \tilde{f}(z_{\rm best} + \gamma_e^{j-1}\delta)}{\delta^2\gamma} \le \frac{f(x^0) - \widehat{f} + 2\omega}{\delta^2\gamma}.$$

Therefore `EP` ends after at most $1 \le j \le E$ iterations; hence E computed by (28) is verified.

We now show that `EP` satisfies condition

$$\|g(z_{\rm best})\| \le w_{\rm EP}\Gamma_-(\delta) \quad \text{with } \Gamma_-(\delta) := L_-\gamma_e^{E-1}\delta_+ \frac{4\omega}{\gamma_{\rm rd}\delta}. \tag{30}$$

To do so, we define

$$\Delta_-(\alpha) := L_-\alpha + 4\omega/(\gamma_{\rm rd}\alpha) \tag{31}$$

and then show that

$$\Delta_-(\alpha) < \Gamma_-(\delta). \tag{32}$$

The extrapolation step size α_r in iteration r satisfies

$$\alpha_{E-1} := \max_{r=1:R} \alpha_r = \gamma_e^{E-1}\delta > \alpha_1 := \min_{r=1:R} \alpha_r = \delta$$

because of the rule for updating step size in line 3 of `EP` and since $\gamma_e > 1$. If $L_- > 0$, since $\Delta_-(\alpha)$ for $\alpha > 0$ is a convex function, (32) is obtained from

$$\begin{aligned} \Delta_-(\alpha) &\le \max\{\Delta_-(\alpha_1), \Delta_-(\alpha_{E-1})\} < L_-\alpha_{E-1} + \frac{4\omega}{\gamma_{\rm rd}\alpha_1} \\ &= L_-\gamma_e^{E-1}\delta + \frac{4\omega}{\gamma_{\rm rd}\delta} = \Gamma_-(\delta), \end{aligned}$$

Otherwise, $L_- \leq 0$ holds. Since $\Delta_-(\alpha)$ for $\alpha > 0$ is a concave function, it can be easily seen that $\Delta_-(\alpha)$ does not has a maximizer. In this case, (32) is obtained from

$$\Delta_-(\alpha) = L_-\alpha + 4\omega/(\gamma_{\rm rd}\alpha) < 4\omega/(\gamma_{\rm rd}\alpha) \leq 4\omega/(\gamma_{\rm rd}\alpha_1) = 4\omega/(\gamma_{\rm rd}\delta) < \Gamma_-(\delta).$$

In each successful iteration of EP, one of the following two cases happens:
Case 1. $\tilde{f}(z_+) - \tilde{f}(z_{\rm best}) < -\gamma\alpha^2$, where $z_+ = z_{\rm best} + \alpha p$ is the new best point found by EP. Then, we obtain from (20)

$$\alpha g(z_{\rm best})^T p - \frac{1}{2}L\alpha^2\|p\|^2 \leq f(z_+) - f(z_{\rm best}) \overset{(3)}{\leq} \tilde{f}(z_+) - \tilde{f}(z_{\rm best}) + 2\omega < 2\omega - \gamma\alpha^2,$$

so that

$$|g(z_{\rm best})^T p| < -\gamma\alpha + 2\omega/\alpha + \frac{L}{2}\alpha\|p\|^2 = -\gamma\alpha + 2\omega/\alpha + \frac{L}{2}\gamma_{\rm rd}^2\alpha. \tag{33}$$

Here $\|p\| = \gamma_{\rm rd}$ is from the definition of the scaled random direction in Section 3.2.
Case 2. $\tilde{f}(z_-) - \tilde{f}(z_{\rm best}) < -\gamma\alpha^2$, where $z_- := z_{\rm best} - \alpha p$ is the new best point found by EP. Then, we obtain from (20)

$$-\alpha g(z_{\rm best})^T p - \frac{1}{2}L\alpha^2\|p\|^2 \leq f(z_-) - f(z_{\rm best}) \overset{(3)}{\leq} \tilde{f}(z_-) - \tilde{f}(z_{\rm best}) + 2\omega < 2\omega - \gamma\alpha^2,$$

so that condition (33) holds.

In both cases, we conclude from (24) that

$$\begin{aligned}\|g(z_{\rm best})\| &= \frac{\|g(z_{\rm best})\|\|p\|}{\gamma_{\rm rd}} = \frac{\|g(z_{\rm best})\|\|p\|}{\gamma_{\rm rd}}\frac{2|g(z_{\rm best})^T p|}{2|g(z_{\rm best})^T p|} = 2\gamma_{\rm rd}^{-1} w_{\rm EP}|g(z_{\rm best})^T p| \\ &\leq w_{\rm EP}\left(L_-\alpha + \frac{4\omega}{\gamma_{\rm rd}\alpha}\right) = w_{\rm EP}\Delta_-(\alpha) \overset{(32)}{<} w_{\rm EP}\Gamma_-(\delta);\end{aligned}$$

hence condition (30) holds.

Together with Lemma 4.8, EP satisfies condition (29): Since $L_+ = \max(L_-, L_+)$, (30) and (25) imply that in each iteration

$$\begin{aligned}\|g(z_{\rm best})\| &\leq w_{\rm EP}\max(\Gamma_+(\delta), \Gamma_-(\delta)) < w_{\rm EP}\left(L_+\gamma_e^{E-1}\delta + \gamma_e^{R-1}\frac{4\omega}{\gamma_{\rm rd}\delta}\right) \\ &\leq w_{\rm EP}\gamma_e^{\max(E,R)-1}\left(L_+\delta + \frac{4\omega}{\gamma_{\rm rd}\delta}\right) = w_{\rm EP}\Gamma(\delta),\end{aligned}$$

giving condition (29). □

4.3.2 A bound for the gradient of the result of MLS

Here, we prove that one of the following holds:
(i) If MLS has at least one successful iteration, a γ-reduction of $\tilde{f}$ is found.
(ii) An upper bound on the unknown gradient norm of an old best point $z_{\rm best}$ is found.

4.10 Theorem. *If Assumptions 4.1 hold, then*

$$\min_{r=1:R} \|g(z^r_{\text{best}})\| \leq w_{\text{MLS}}\Gamma(\delta). \tag{34}$$

Here R is from (11) *and $\Gamma(\delta)$ is from* (29)*. Moreover, if* `MLS` *has at least one successful iteration, then $\tilde{f}$ decreases by at least $\gamma\alpha^2$.*

Proof. We denote by z^r $(r = 1, 2, \ldots, R)$ the sequence generated by `MLS`. For each iteration $r = 1, 2, \ldots, R$, we obtain from Proposition 4.9

$$\|g(z^{r-1}_{\text{best}})\| \leq w_{\text{EP}}\Gamma(\delta), \tag{35}$$

where all z^{r-1}_{best} for $r = 1, 2, \ldots, R$ have been evaluated by `EP` and are not necessarily distinct since there is no guarantee that each iteration r of `EP` is successful. Using (35) and the definition of w_{MLS}, condition (34) is obtained.

If `MLS` has at least one successful iteration, the line search condition

$$\tilde{f}(z_{\text{best}}) - \tilde{f}(z_{\text{best}} + \tilde{\alpha}p) > \gamma\tilde{\alpha}^2 \geq \gamma\alpha^2$$

at the trial point $z_{\text{best}} + \tilde{\alpha}p$ holds; here $\tilde{\alpha} \geq \alpha$ because of expanding step sizes in `EP`, see lines 3 and 7 of `EP`. Consequently, $\tilde{f}$ decreases by at least $\gamma\alpha^2$. □

Such a bound on $\|g\|$ can be found in the recent paper by Brilli et al. [2, Proposition 3.2] for a different deterministic derivative-free line search method.

4.3.3 A bound for the gradient of the result of `DS`

Here, we prove that either a γ-reduction of $\tilde{f}$ is found (if `DS` has at least one successful iteration) or an upper bound on the unknown gradient norm of an old best point y_{best} is found.

4.11 Theorem. *Let $f(x^0)$ be the initial value of f. If Assumptions 4.1 hold, then:*

$$\min_{t=1:T_0} \|g(y^t_{\text{best}})\| < w_{\text{DS}}\Gamma(\delta). \tag{36}$$

Here, $\Gamma(\delta)$ is from (29)*. Moreover, if* `DS` *has at least one successful iteration, then it decreases $\tilde{f}$ by at least $\overline{\gamma}\delta^2$, where $\overline{\gamma} := \gamma\gamma_e^{2(2-R)} > 0$.*

Proof. By Theorem 4.10, for any fixed $t = 1, \ldots, T_0$, we have

$$\|g(y^t_{\text{best}})\| = \min_{r=1:R} \|g(z^r_{\text{best}})\| \leq w_{\text{MLS}}\Gamma(\delta).$$

Here all z^r_{best} for $r = 1, \ldots, R$ have been evaluated by `MLS`, one of which with the smallest $\tilde{f}$ has been chosen as y^t_{best} for each iteration t of `DS`; see line 20 of `MLS` and line 3 of `DS`. Therefore, condition (36) is obtained from condition (34) and the definition of w_{DS}.

If DS has at least one successful iteration, there exists a $y_{\text{best}} = z_{\text{best}}$ (see line 20 of MLS), so that

$$\tilde{f}(y_{\text{best}}) - \tilde{f}(y_{\text{best}} + \tilde{\alpha}p) = \tilde{f}(z_{\text{best}}) - \tilde{f}(z_{\text{best}} + \tilde{\alpha}p) > \gamma\tilde{\alpha}^2 \geq \gamma\alpha^2. \tag{37}$$

By setting (19) into (37), we have

$$\tilde{f}(y_{\text{best}}) - \tilde{f}(y_{\text{best}} + \tilde{\alpha}p) > \gamma\alpha^2 \geq \gamma(\gamma_e^{1-R}\delta)^2 = \overline{\gamma}\delta^2,$$

where $\overline{\gamma} := \gamma\gamma_e^{2(1-R)}$. Consequently, $\tilde{f}$ decreases by at least $\overline{\gamma}\delta^2$.

□

4.3.4 A bound for the gradient of the result of VRDFON

This subsection discusses complexity for VRDFON. Under Assumptions 4.1, we prove that an upper bound for the unknown gradient norm of one of the old best point is found for the nonconvex case.

The following result is obtained from Theorem 4.11 and Proposition 4.2.

4.12 Theorem. *If Assumptions 4.1 hold, then:*

$$\min_{k=0:K} \|g(x^k_{\text{best}})\| = \mathcal{O}\Big(w_{\text{VRDFON}} L\sqrt{\omega}\Big). \tag{38}$$

Proof. Let $\{x^k\}_{k\geq 0}$ be the sequence generated by VRDFON. By Theorem 4.2, VRDFON ends after at most K iterations (K' unsuccessful iterations and K'' successful iterations). From Theorem 4.11 for any fixed $t = 1, \ldots, T_0$, we have

$$\|g(x^k_{\text{best}})\| = \min_{t=1:T_0} \|g(y^t_{\text{best}})\| \leq w_{\text{DS}}\Gamma(\delta).$$

Here all y^t_{best} for $t = 1, \ldots, T_0$ have been evaluated by DS, one of which with the smallest $\tilde{f}$ has been chosen as x^k_{best} for each iteration k of VRDFON; see line 7 of DS and line 5 of VRDFON. Therefore, condition (38) is obtained from condition (36), the definition of w_{VRDFON}, and

$$\min_{k=0:K} \|g(x^k_{\text{best}})\| \leq w_{\text{VRDFON}} \min_{k=1:K+1} \Gamma(\delta_{k-1}). \tag{39}$$

Let $j_{K'} \in \overline{U}$ be the last unsuccessful iteration of VRDFON. Then, we have

$$\Gamma(\delta_{\underline{k}-1}) = \min_{k=1:K+1} \Gamma(\delta_{k-1}) \leq \Gamma(\delta_{j_{K'}-1}). \tag{40}$$

We now reformulate condition (29) as

$$\Gamma(\delta_{\underline{k}-1}) := \gamma_e^{\max\left(R, E_{\underline{k}}\right)-1}\Delta(\delta_{\underline{k}-1}) \;\text{ with }\; \Delta(\delta_{\underline{k}-1}) = L_+\delta_{\underline{k}-1} + \frac{4\omega}{\gamma_{\text{rd}}\delta_{\underline{k}-1}}.$$

Put $c_f := \sqrt{\gamma^{-1}(f(x^0) - \widehat{f} + 2\omega)} = \mathcal{O}(\sqrt{\omega})$. Since $\delta_{j'_K}$ is the smallest value for δ_k for $k \in \overline{U} \cup \overline{S}$, $\delta_{k-1}^{-1} \leq \delta_{j'_K-1}^{-1}$ is obtained. Then, using conditions (16) and (28), we obtain

$$\gamma_e^{\max\left(R, E_{\underline{k}}\right)-1} = \begin{cases} \mathcal{O}\Big(\sqrt{\omega}\delta_{j'_K-1}^{-1}\Big) & \text{if } E_{\underline{k}} > R, \\ \gamma_e^{R-1} & \text{otherwise.} \end{cases} \tag{41}$$

We distinguish two cases:

Case 1. The first term $L_+\delta_{\underline{k}-1}$ in $\Delta(\delta_{\underline{k}-1})$ dominates the second term $\dfrac{4\omega}{\gamma_{\mathrm{rd}}\delta_{\underline{k}-1}}$. Then

$$\Delta(\delta_{\underline{k}-1}) = \mathcal{O}(L_+\delta_{\underline{k}-1}) \overset{(27)}{=} \mathcal{O}(L\delta_{\underline{k}-1}) = \mathcal{O}(L\delta_{\max}) = \mathcal{O}(L).$$

Here $\delta_{\underline{k}-1} \leq \delta_{\max}$ holds according to the rule of updating for δ_k in lines 3, 8, 10 of `VRDFON`.

Case 2. The second term $\dfrac{4\omega}{\gamma_{\mathrm{rd}}\delta_{\underline{k}-1}}$ in $\Gamma(\delta_{\underline{k}-1})$ dominates the first term. Then

$$\Delta(\delta_{\underline{k}-1}) = \mathcal{O}(\omega\delta_{\underline{k}-1}^{-1}) \overset{(16)}{=} \mathcal{O}(\omega\delta_{\min}^{-1}) \overset{(13)}{=} \mathcal{O}(\sqrt{L\omega}).$$

From (41), if $E_{\underline{k}} > R$, then

$$\gamma_e^{\max\left(R, E_{\underline{k}}\right)-1} = \mathcal{O}\Big(\sqrt{\omega}\delta_{j'_K-1}^{-1}\Big) \overset{(16)}{=} \mathcal{O}(\sqrt{\omega}\delta_{\min}^{-1}) \overset{(13)}{=} \mathcal{O}(\sqrt{L});$$

otherwise, $\gamma_e^{\max\left(R, E_{\underline{k}}\right)-1} = \gamma_e^{R-1}$ is a constant. Therefore

$$\Gamma(\delta_{\underline{k}-1}) = \gamma_e^{\max\left(R, E_{\underline{k}}\right)-1}\Delta(\delta_{\underline{k}-1}) = \mathcal{O}(\sqrt{L}\sqrt{L\omega}) = \mathcal{O}(L\sqrt{\omega}). \tag{42}$$

which verifies (38). From (39), (40), and (42), we obtain

$$\begin{aligned} \min_{k=0:K} \|g(x_{\mathrm{best}}^k)\| &\leq w_{\mathrm{VRDFON}} \min_{k=1:K+1} \Gamma(\delta_{k-1}) \\ &= w_{\mathrm{VRDFON}}\Gamma(\delta_{\underline{k}-1}) = \mathcal{O}\Big(w_{\mathrm{VRDFON}}L\sqrt{\omega}\Big), \end{aligned}$$

which verifies (38). □

4.3.5 A probabilistic bound

Essential for our complexity bounds is the following result (Proposition 2 in [6]) for the unknown gradient $g(x)$ of $f(x)$ at $x \in \mathbb{R}^n$. It holds for any norm; hence for any scaling vector $s \in \mathbb{R}^n$ and shows that scaled random directions satisfy a two-sided angle condition with probability at least 0.5.

4.13 Proposition. *Any scaled random direction p satisfies the inequality*

$$\Pr\left(\|g(x)\|\|p\| \leq 2\sqrt{cn}|g(x)^T p|\right) \geq \frac{1}{2} \tag{43}$$

with a positive constant $c \leq 12.5$.

4.14 Corollary. *If p is a scaled random direction and $x \in \mathbb{R}^n$, then $w(p,x)$ from* (24) *satisfies*

$$\Pr\left(w(p,x) > \sqrt{cn}\right) < \frac{1}{2} \tag{44}$$

with a positive constant $c \leq 12.5$.

The condition (43) is called **two-sided angle condition** because we cannot check whether any scaled random direction is a descent direction or not. Hence, instead of searching along one ray $\alpha > 0$ only, our line search allows to search the line $x + \alpha p$ in both directions ($\alpha \in \mathbb{R}$).

4.15 Theorem. *If Assumptions 4.1 hold, then for any given $0 < \eta < \frac{1}{2}$ the bound* (38) *is of order $\mathcal{O}(\varepsilon_\omega)$ with probability $\geq 1 - \eta$. Moreover, the bound* (8) *for the convex and strongly convex cases and the bound* (10) *for the strongly convex case are obtained with probability $\geq 1 - \eta$.*

Proof. By Corollary 4.14 and the definition of w_{EP}, for each iteration of `EP`

$$\Pr\left(w_{\mathrm{EP}} > \sqrt{cn}\right) < \frac{1}{2}.$$

Since `MLS` has R calls to `DS`, we obtain from the definition of w_{MLS} and (44), for each iteration of `MLS`,

$$\Pr\left(w_{\mathrm{MLS}} > \sqrt{cn}\right) = \prod_{r=1}^{R} \Pr\left(w_{\mathrm{EP}} > \sqrt{cn}\right) < 2^{-R}. \tag{45}$$

Then, since `DS` has T_0 calls to `MLS`, we obtain from the definition of w_{DS} and (45), for each iteration of `DS`,

$$\Pr\left(w_{\mathrm{DS}} > \sqrt{cn}\right) = \prod_{t=1}^{T_0} \Pr\left(w_{\mathrm{MLS}} > \sqrt{cn}\right) < 2^{-RT_0}. \tag{46}$$

Finally, since `VRDFON` has K calls to `DS`, we obtain from the definition of w_{VRDFON} and (46)

$$\Pr\left(w_{\mathrm{VRDFON}} > \sqrt{cn}\right) = \prod_{k=1}^{K} \Pr\left(w_{\mathrm{DS}} > \sqrt{cn}\right) < 2^{-RT_0K}.$$

In other words, by the definition of R in (11), we have

$$\Pr\left(w_{\mathrm{VRDFON}} \leq \sqrt{cn}\right) \geq 1 - 2^{-RT_0K} \geq 1 - \eta. \tag{47}$$

From (47) and (38), the first result follows. The second and third results are obtained by applying the first result in (8) and (10), respectively. □

The order of ω in the bound (38) is the same as that in the conditions from the literature defined in Table 2.

5 Numerical performance of VRDFON

In this section, we provide numerical experiments on a number of test problems of dimensions $300 < n \leq 5000$. We used all 96 noiseless problems of large dimensions $300 < n \leq 1000$ and all 90 noiseless problems of very large dimensions $1000 < n \leq 5000$. With the different noise levels $\omega = 10^{-5}, 10^{-4}, 10^{-3}$, this produced 3×96 large noisy problems and 3×90 very large noisy problems. The test environment of Kimiaei & Neumaier [33] was used to produce these results. The results were averaged over five runs. We compare VRDFON with the three state-of-the-art DFO solvers, VRBBO, SDBOX, and LMMAES on noisy large and very large test problems.

5.1 Choice of solvers

Table 1 lists 21 DFO solvers. One is our new solver VRDFON and 17 others are state-of-the-art solvers from literature. The 3 remaing solvers subUOBYQA, subNEWUOA, and subNMSMAX were obtained by modifying the two model-based solvers UOBYQA and NEWUOA, and the Nelder–Mead solver NMSMAX to proceed in random subspaces.

To select the DFO solvers likely to be competitive, we made some preliminary tests by comparing all 21 DFO solvers listed in Table 1 on the 192 noiseless unconstrained CUTEst test problems of small dimensions ≤ 30. With the different noise levels $\omega = 10^{-3}, 10^{-2}, 0.1, 0.9$, this produced 4×192 noisy problems.

The nine solvers (NOMAD, UOBYQA, NEWUOA, BCDFO, GRID, SNOBFIT, FMINUNC, DSPFD, MCS) were ignored for solving noisy medium scale problems since they were not competitive competitive or were model-based. Indeed, since model-based solvers need a large number of sample points to construct full quadratic models, they are too slow and are not recommended to solve noisy medium to very large scale problems.

We then compared the 11 remaining solvers (VRBBO, SDBOX, BiPopMAES, LMMAES, fMAES, CMAES, BFO, NMSMAX, subUOBYQA, subNEWUOA, subNMSMAX) with VRDFON on all 171 noiseless unconstrained CUTEst test problems of medium dimensions $30 < n \leq 300$. With the different noise levels $\omega = 10^{-4}, 10^{-3}, 10^{-2}, 0.1$, this produced 4×171 noisy problems. The only solvers that remained competitive were VRDFON, VRBBO, SDBOX, and LMMAES.

VRDFON was found competitive with VRBBO, SDBOX, and LMMAES on the noisy small and medium scale problems. The other solvers were not competitive on medium scale problems. Detailed numerical results on 1452 noisy small and mediums scale problems of dimensions

2 to 300 can be found in `impVRDFON.pdf` from the publicly available `VRDFON` package [34]. This file describes details of enhancements, the solvers compared, and testing and tuning for `VRDFON`.

5.2 New practical enhancements

As explained in detail in [34], `VRDFON` uses several different directions to enrich `MLS` in the presence of noise. It is well known that the complexity of randomized DFO methods is better than that of deterministic methods by a factor of n in the worst case (cf. [27]); therefore, using random directions seems preferable to using deterministic ones. Even better directions than random directions are random approximate coordinate directions. Improved trust region directions are found by minimizing surrogate quadratic models in adaptively determined subspaces within a trust region. Perturbed random directions are perturbations of random directions by scaled approximate descent directions in adaptively determined subspaces. `VRDFON` constructs surrogate quadratic models in adaptively determined subspaces that can handle medium and large scale problems. Although these models have lower accuracy in higher dimensions, their usefulness has been confirmed in extensive numerical experiments in the presence of strong noise. `VRDFON` changes extrapolation step sizes heuristically when they become too small in the presence of substantial noise to prevent generating zero steps.

5.3 Starting and stopping

The starting point: As in [6], the starting point $x^0 := \xi$

$$\xi_i := (-1)^{i-1}\frac{2}{2+i}, \text{ for all } i = 1, \ldots, n$$

is chosen and the initial inexact function value $\tilde{f}_0 := \tilde{f}(x^0)$, while we compute the other inexact function values by $\tilde{f}_\ell := \tilde{f}(x^\ell + \xi)$ for all $\ell \geq 0$. The reason for this choice is that there are some toy problems in the `CUTEst` library with a simple solution whose solution can be easily guessed by the solver.

Measure for the convergence speed: The quotients

$$q_s := (f_s - f_{\text{opt}})/(f_0 - f_{\text{opt}}) \quad \text{for } s \in \mathcal{S} \tag{48}$$

are measures to identify the convergence speed of the solver s to reach a minimum of the smooth true function f. These quotients are not available in real applications. Here
- f_s is the best function value found by the solver s,
- f_0 is the function value at the starting point (common for all solvers),
- f_{opt} is the function value at the best known point (in most cases a global minimizer or at least a better local minimizer) found by running a sequence of gradient-based and local/global gradient free solvers; see Appendix B in [6].

Type of noise: In the numerical results reported here, uniform random noise is used, which is consistent with the assumption (A3). The function values are calculated by $\tilde{f} = f + (2 * \text{rand} - 1)\omega$, where f is the true function value and $\omega \geq 0$ is a noise level whose size identifies the difficulty of the noisy problems. Here rand stands for the uniformly distributed random number on $[0, 1]$.

Stopping: We consider a problem **solved** by the solver s if $q_s \leq \varepsilon_q$ and neither the maximum number `nfmax` of function evaluations nor the maximum allowed time `secmax` in seconds was reached, and **unsolved** otherwise. The following choices were found valuable for large and very large scale noisy problems:

$$\texttt{secmax} := 420, \quad \texttt{nfmax} := 500n, \quad \varepsilon_q := 0.05.$$

ε_q, `secmax` and `nfmax` were chosen so that the best solver can solve more than half of the problems.

5.4 Profiles

Efficiency and robustness: Efficiency measures the ability of a solver $s \in \mathcal{S}$ relative to an ideal solver. The number `nf` of function evaluations is taken as a suitable cost measure, and the efficiency relative to this measure is called the `nf` efficiency. The **robustness** of a solver counts the number of problems it solves. A solver with a high number of solved problems is called **robust** and a solver with a low relative cost of function evaluations is called **efficient**.

Performance and data profile: Two important tools for figuring out which solver is **robust** and **efficient** are the data profile of Moré & Wild [35] and the performance profile of Dolan & Moré [36], respectively. $\mathcal{S}$ denotes the list of compared solvers and $\mathcal{P}$ denotes the list of problems. The fraction of problems that the solver s can solve with κ groups of $n_p + 1$ function evaluations is the data profile of the solver s, i.e.,

$$\delta_s(\kappa) := \frac{1}{|\mathcal{P}|}\Big|\Big\{p \in \mathcal{P} \;\Big|\; cr_{p,s} := \frac{c_{p,s}}{n_p + 1} \leq \kappa\Big\}\Big|. \tag{49}$$

Here n_p is the dimension of the problem p, $c_{p,s}$ is the **cost measure** of the solver s to solve the problem p and $cr_{p,s}$ is the **cost ratio** of the solver s to solve the problem p. The fraction of problems that the performance ratio $pr_{p,s}$ is at most τ is the performance profile of the solver s, i.e.,

$$\rho_s(\tau) := \frac{1}{|\mathcal{P}|}\Big|\Big\{p \in \mathcal{P} \;\Big|\; pr_{p,s} := \frac{c_{p,s}}{\min(c_{p,\bar{s}} \mid \bar{s} \in S)} \leq \tau\Big\}\Big|. \tag{50}$$

Note that $\rho_s(1)$ is the fraction of problems that the solver s wins compared to the other solvers, while $\rho_s(\tau)$ ($\delta_s(\kappa)$) is the fraction of problems for sufficiently large τ (κ) that the solver s can solve. The data and performance profiles are based on the problem scales, but not on the noise levels.

Noise profiles: To identify the behaviour of the compared solvers in the presence of low to high noise, we plot the number of problems solved and the efficiency of the compared solvers against the noise level, yielding two noise profiles for efficiency and robustness with respect to the noise levels.

5.5 Large scale: $300 < n \leq 1000$

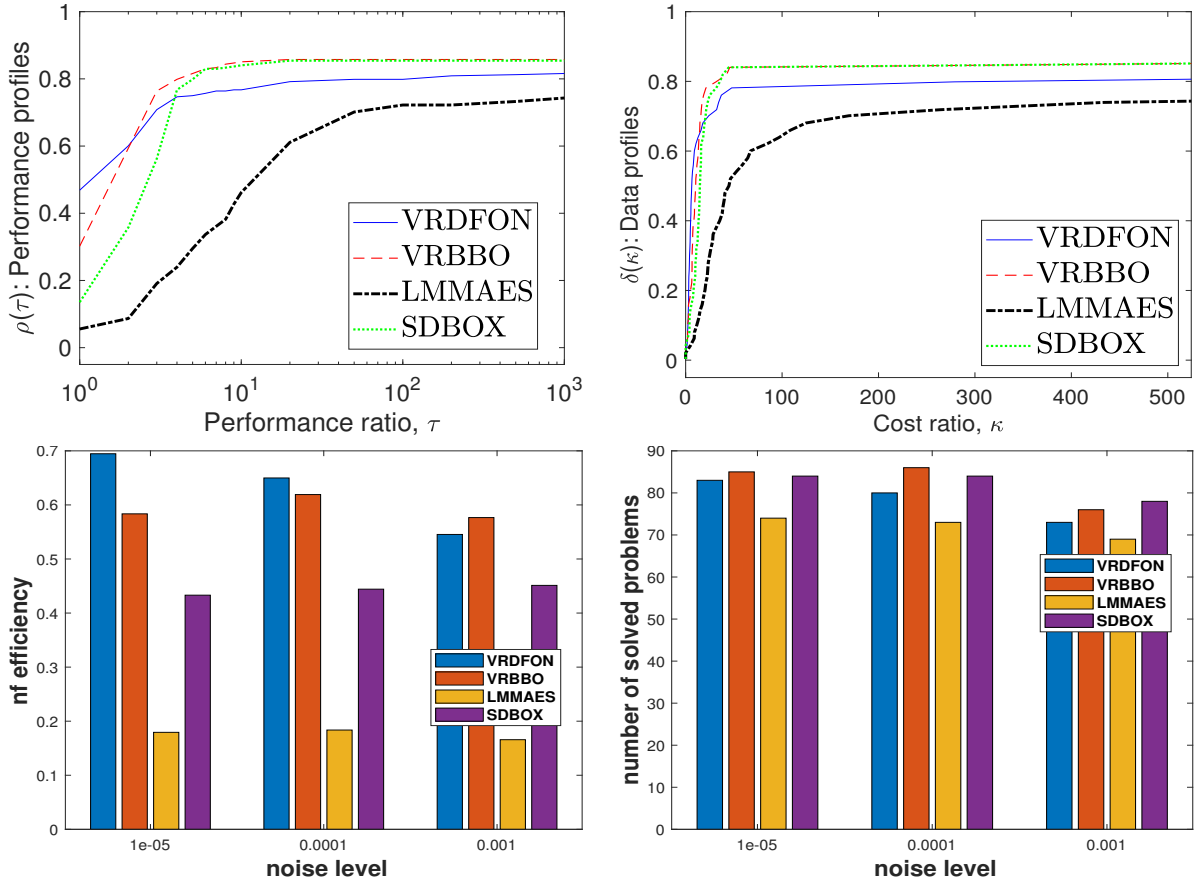

Figure 1: First row: Comparison between `VRDFON` and the effective solvers on the large scale problems $300 < n \leq 1000$ for the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$. Data profile $\delta(\kappa)$ in dependence of a bound κ on the cost ratio while performance profile $\rho(\tau)$ in dependence of a bound τ on the performance ratio. Problems solved by no solver are ignored. Second row: Noisy profiles for more robust and efficient DFO solvers listed in Table 1 on large scale problems $300 < n \leq 1000$. Here '# solved problems' counts the number of solved problems

This subsection contains a comparison between VRDFON and the three more robust and efficient solvers (VRBBO, LMMAES, and SDBOX) on the noisy problems in large dimensions $300 < n \leq 1000$.

In terms of the number of function evaluations and for the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$, The first row of Figure 1 shows the cumulative (over all noise levels used) performance and data profiles, while its second row shows noise profiles with respect to the noise levels.

For all noise levels, VRDFON solved 236 noisy problems out of 288 noisy problems, while VRBBO, SDBOX, and LMMAES solved 247, 246, and 216 noisy problems out of 288 noisy problems, respectively. In terms of relative cost for nf, VRDFON is the winner with 44% noisy problems compared to the others. Thus, we conclude that VRDFON is more efficient than others, while VRBBO and SDBOX are more robust than VRDFON.

5.6 Very large scale: $1000 < n \leq 5000$

In terms of the number of function evaluations and for the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$, the first row of Figure 2 shows the cumulative (over all noise levels used) performance and data profiles, while its second row shows noise profiles with respect to the noise levels. For all noise levels, VRDFON solved 147 noisy problems out of 270 noisy problems, while VRBBO, SDBOX, and LMMAES solved 173, 169, and 48 noisy problems out of 270 noisy problems, respectively. In terms of relative cost for nf, VRDFON is the winner at 43% noisy problems compared to the others. Hence, we conclude from these subfigures that VRDFON is more efficient than others, while VRBBO and SDBOX are more robust than VRDFON.

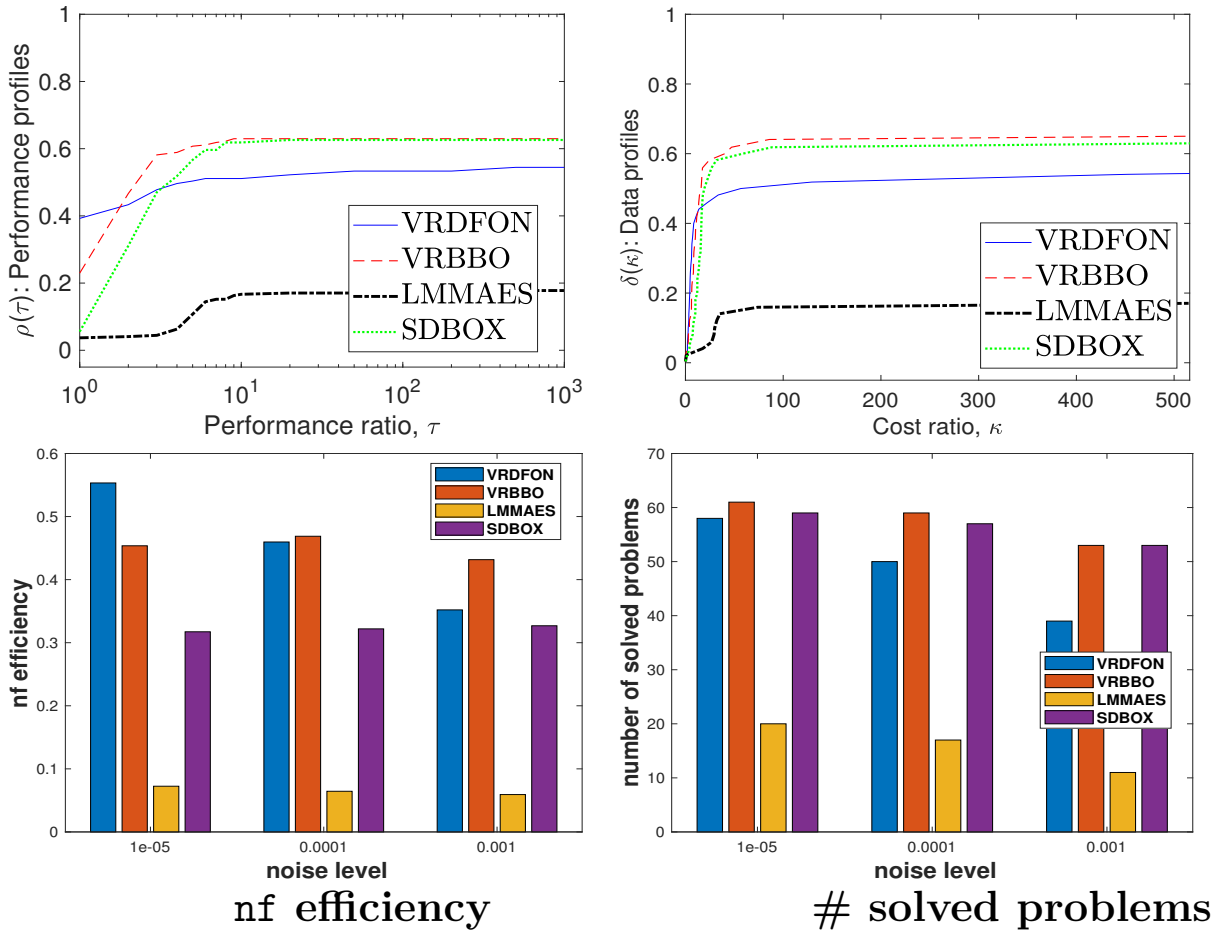

Figure 2: First row: Comparison between VRDFON and the effective solvers on large dimensions $1000 < n \leq 5000$ for the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$. Data profile $\delta(\kappa)$ in dependence of a bound κ on the cost ratio while performance profile $\rho(\tau)$ in dependence of a bound τ on the performance ratio. Problems solved by no solver are ignored. Second row: Noisy profiles for more robust and efficient DFO solvers listed in Table 1 on the very large scale problems $1000 < n \leq 5000$. Here '# solved problems' counts the number of solved problems.

6 Real-life applications

This section gives a comparison between VRDFON and several stochastic DFO solvers to solve the real-life problem MM1 from SimOpt by Dong et al. [37], available at

https://github.com/simopt-admin/simopt/wiki,

originally from Cheng & Kleijnen [38].

SimOpt is a test environment for simulation optimization problems and solvers with the goal of promoting the development and constructive comparison of simulation optimization solvers. In practice, SimOpt tests the performance of solvers in finite time rather than the asymptotic results often found in the related literature. It provides a solvers library with ten solvers, 8 of which are listed in Table 3 (except the SPSA and GASSO solvers that were not used for our comparison because they have the lowest performance compared to others), and a problem library with 9 test problems, where MM1 is the only unconstrained problem. We here compare VRDFON with VRBBO and 8 solvers listed in Table 3 to solve the MM1 problem. This problem has 3 variables and describes the parameter estimation in a queueing problem.

As in [37], we choose a simulation budget for finite termination and evaluation between the compared solvers so that each compared solver can find the estimated best solution before the budget is reached. One objective function evaluation is calculated by replicating the simulation r times and averaging the result. The simulation budget is in terms of the number nf of objective function evaluations (i.e., $\texttt{nf} * r$ simulation runs). A **macroreplication** is a single execution of an algorithm on a given problem instance using the simulation budget. We denote by $f(x_n)$ the true objective function value of the estimated best solution x_n visited in the first n objective function evaluations on a given macroreplication. Since x_n is random, $f(x_n)$ is a random variable. Conditional on x_n, the objective function value $f(x_n)$ is not random, but it is unlikely that we can compute it accurately, since we evaluate the objective function by simulations.

In our experiments, we follow the testing procedure of [37]. Thus, we perform additional replications in a post-processing step to obtain fairly precise estimates of x_n conditional on x_n. These replications are not counted in the budget of the algorithm. Since the plot of the $f(x_n)$ curve for one macroreplication is of limited value and the location of the curve is random, it is more informative to run several macroreplications and average them to obtain a mean performance curve.

To solve the MM1 problem, we performed 15 macroreplications of each algorithm. For each macroreplication, we used a post-processing step to generate a sequence of estimated best solutions x_n whose objective function values $f(x_n)$ are the average of a run with $r = 1$ and a run with $r = 30$. These post-processing replications are independent of the replications used to determine the sequence of solutions, and they use common random numbers for all algorithms. We then averaged the 15 estimates of $f(x_n)$ to produce the $f_{\text{mean}}(x_n)$ curve in the following figures. In these figures, we computed 95% normal confidence intervals around $f_{\text{mean}}(x)$ by plotting $f_{\text{mean}}(x) + 1.96\sigma$ and $f_{\text{mean}}(x) - 1.96\sigma$ of the 15 (macroreplication) samples of $f(x_n)$, where σ is the sample standard deviation and the value 1.96 is chosen

such that 95% of $N(0,1)$ distributed random numbers are in $[-1.96, 1.96]$.

Figure 3 shows a comparison between `VRDFON` with $r = 1$ and `VRDFON` with $r = 30$ (note that for $r = 30$ each function value average uses a budget of 30 function evaluations; hence curves with $r = 30$ start later). In this figure, `VRDFON` with $r = 1$ and `VRDFON` with $r = 30$ are independent of each other and each have the three curves f_{mean} (middle), $f_{\text{mean}}(x) + 1.96\sigma$ (top) and $f_{\text{mean}}(x) - 1.96\sigma$ (bottom). From this figure, we conclude that, for a sufficiently large budget, `VRDFON` with $r = 30$ reaches a better accuracy than `VRDFON` with $r = 1$. Consequently, `VRDFON` with large replications has better performance than with small replications. This is the result of the variance reduction effect due to the 30-fold averaging in the oracle for the objective function.

With $r = 1$ and $r = 30$, Figure 4 shows a comparison between `VRDFON` and `VRBBO` and all 8 solvers from Table 3, and Figure 5 shows a comparison between three best solvers. From these figures, to reach a better accuracy, we conclude that:

• With $r = 1$, `VRDFON` is the third best solver, while `STRONG` and `SASGD` are the best and second best solvers, respectively.

• With $r = 30$, `VRDFON` is the third best solver, while `VRBBO` and `STRONG` are the best and second best solvers, respectively.

solver	algorithm
`ANDFER`	direct search algorithm for noisy DFO [39]
`ASTRDF`	adaptive sampling trust-region algorithm for stochastic DFO [40]
`SASGD`	adaptive sampling stochastic Gradient Descent [37]
`SSSGD`	stopping sampling stochastic Gradient Descent [37]
`KWCDLS`	Kiefer-Wolfowitz SA with central differences and line search [37]
`NELDMD`	Nelder-Mead for simulation optimization [41]
`RANDSH`	random search [37]
`STRONG`	stochastic trust-region response-surface method [42]

Table 3: The 8 solvers from the solver library of `SimOpt`.

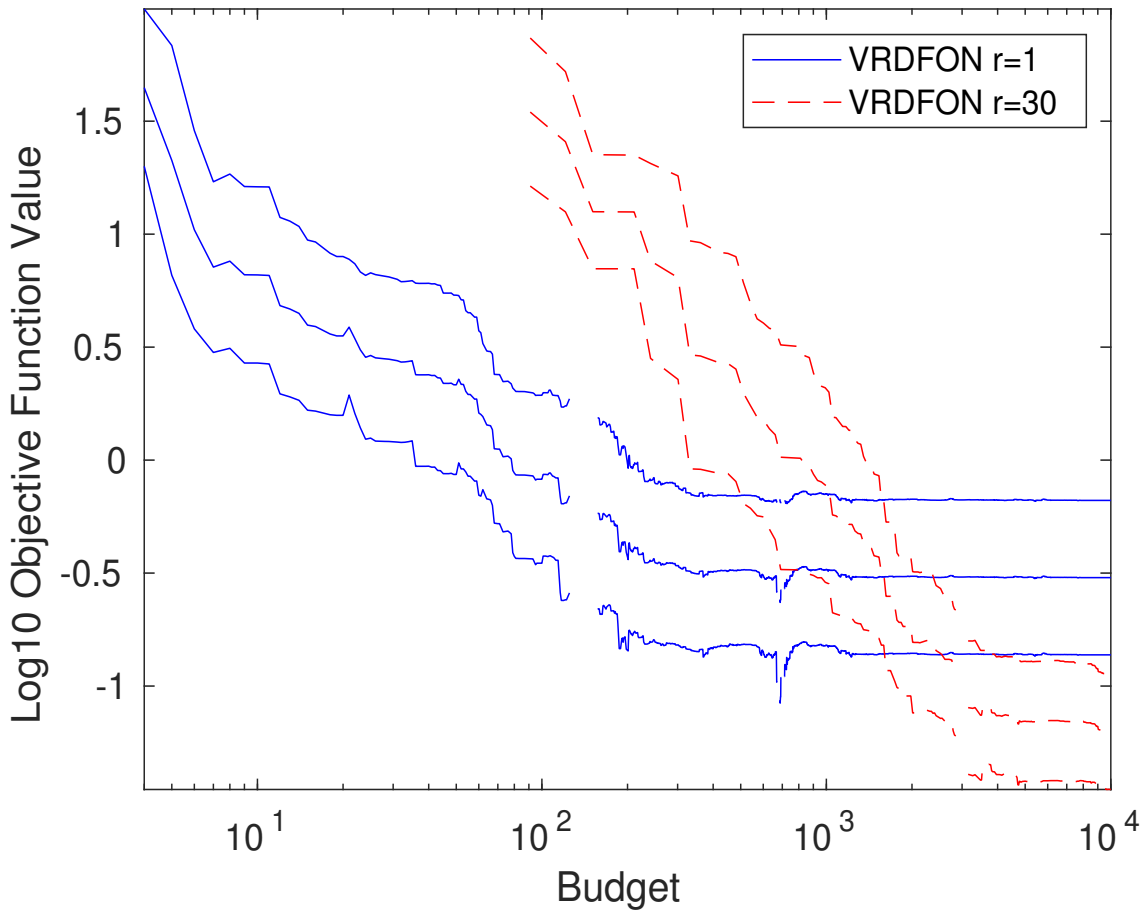

Figure 3: Average performances (mean-confidence interval) f_{mean} with 95% normal confidence intervals ($f_{\text{mean}}(x) + \lambda\sigma$ and $f_{\text{mean}}(x) - \lambda\sigma$ of the 15 (macroreplication) samples of $f(x_n)$) around it for `VERDFON` with $r = 1$ and `VERDFON` with $r = 30$ to solve the `MM1` problem with the dimension $n = 3$ and `nfmax` $= 10000$. Here σ is the sample standard deviation. Moreover, `VERDFON` with $r = 1$ and `VERDFON` with $r = 30$ were performed independently. In both cases, we show the three curves f_{mean} (middle), $f_{\text{mean}}(x) + 1.96\sigma$ (top), and $f_{\text{mean}}(x) - 1.96\sigma$ (bottom).

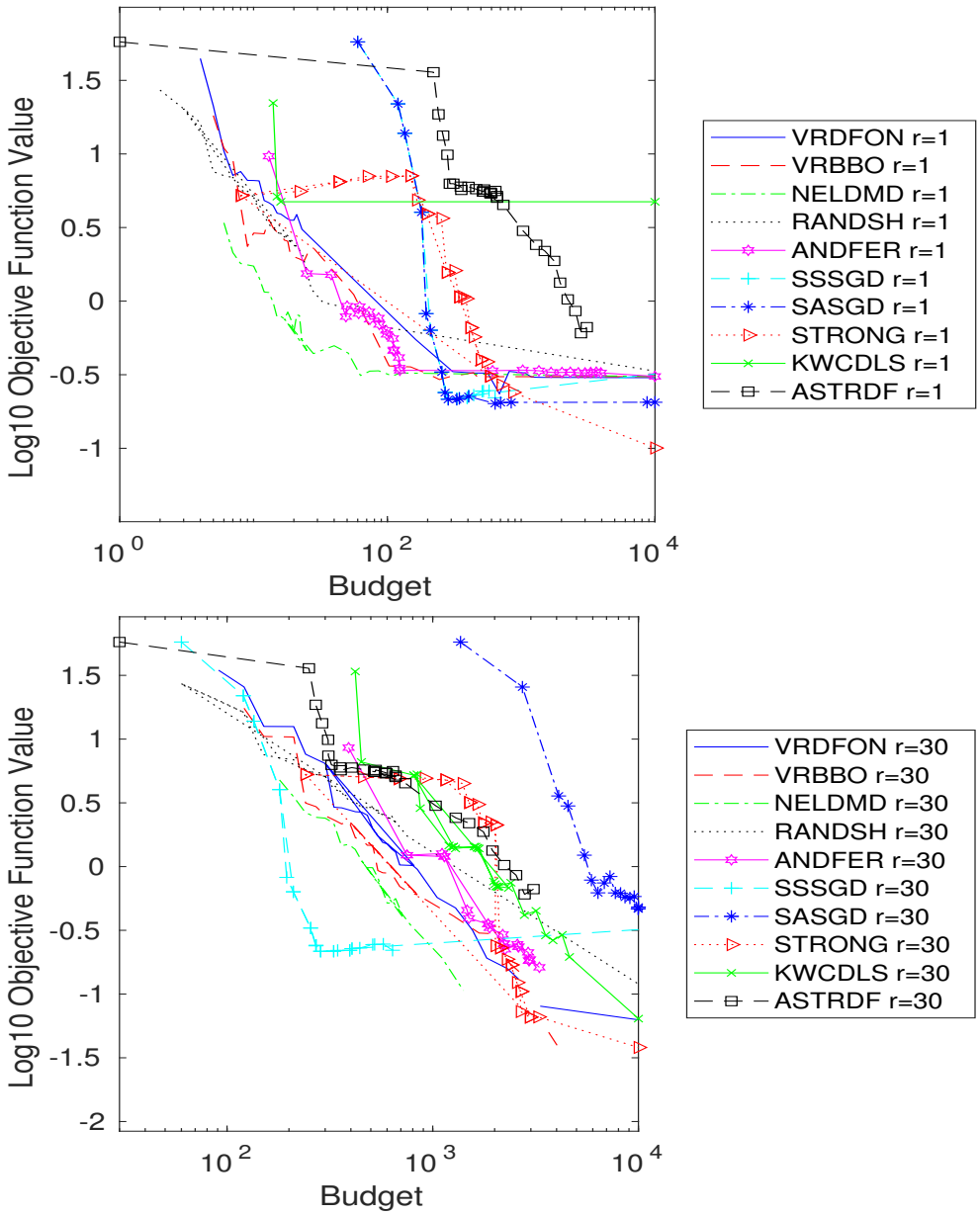

Figure 4: For the `MM1` problem with the dimension $n = 3$ and `nfmax` $= 10000$, average performances of all 10 solvers for $r = 1$ (top) and $r = 30$ (bottom). Other details are as in Figure 3.

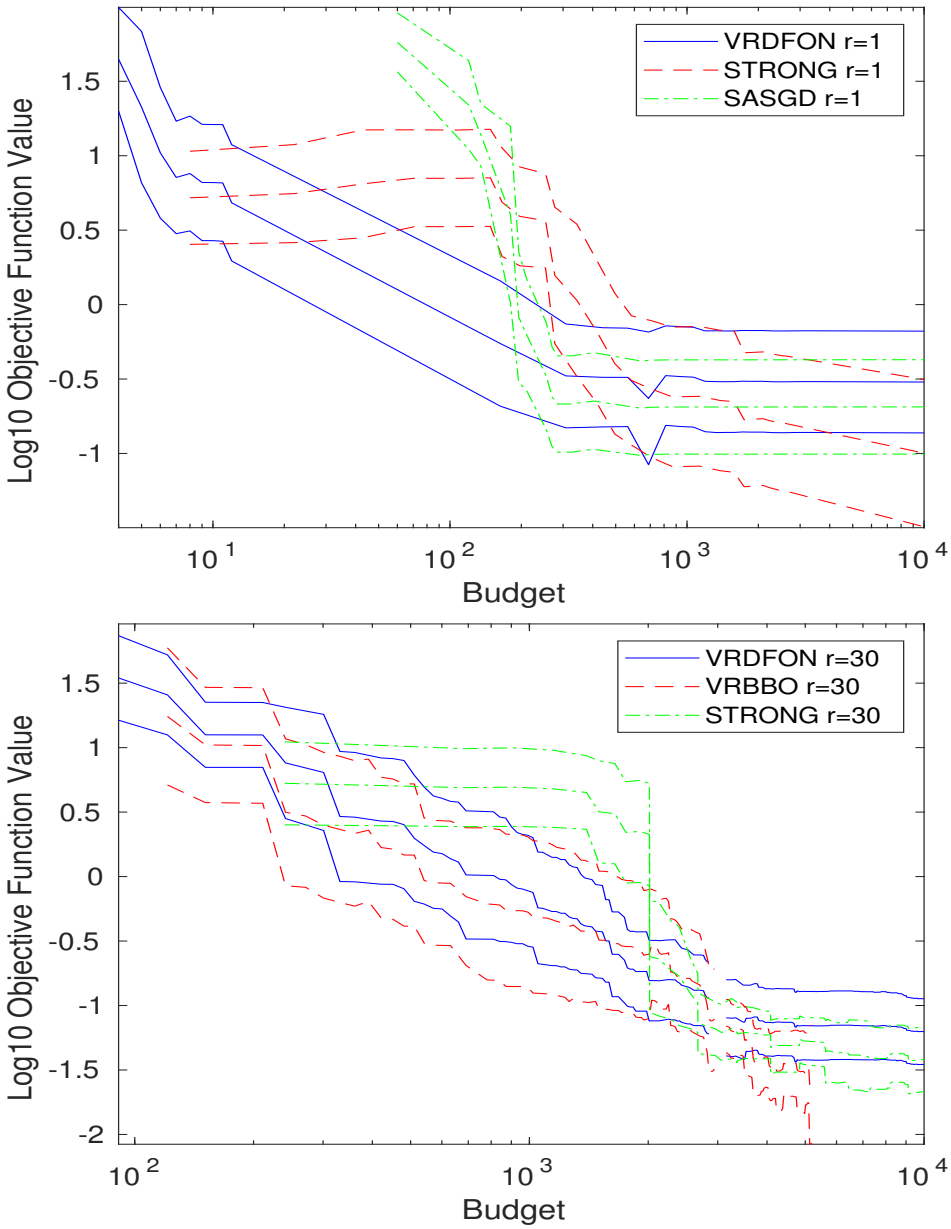

Figure 5: For the `MM1` problem with the dimension $n = 3$ and `nfmax` $= 10000$, average performances (mean confidence interval) of the 3 best solvers for $r = 1$ (top) and $r = 30$ (bottom). Other details are as in Figure 3.

7 Conclusion

This paper discusses VRDFON, a generalized randomized line search algorithm for noisy unconstrained large scale DFO problems. Complexity results for VRDFON in the nonconvex, convex, and strongly convex cases with a given probability arbitrarily close to one are proved.

Due to the use of quadratic models in adaptively determined subspaces and other new heuristic techniques (discussed in impVRDFON.pdf), VRDFON is much more efficient and robust than VRBBO for small and medium scale problems. As the quality of these quadratic models decreases with increasing dimension, VRBBO is more robust than VRDFON, but still more efficient than VRBBO for large problems due to the use of other heuristic techniques.

As a consequence of our results, VRDFON is highly recommended for solving noisy unconstrained large scale problems when the computation of the function value is expensive and efficiency is more important than robustness. It also has good performance in solving the real-life problem MM1 [38] with large replications. This is the result of the variance reduction effect due to the 30-fold averaging in the oracle for the objective function.

Future work could be to increase the quality of quadratic models in the subspace, so that a new version of VRDFON can be not only the most efficient, but also the most robust for large scale DFO problems.

Acknowledgment. Thanks also to Giampaolo Liuzzi for preparing a MEX file for calling UOBYQA from Matlab and to Arnold Neumaier for useful discussions. The author appreciates the unknown referee's valuable and profound comments.

Funding. The author acknowledges the financial support of the Doctoral Program *Vienna Graduate School on Computational Optimization* (*VGSCO*) funded by the Austrian Science Foundation under Project No W1260-N35.

Data Availability. The VRDFON package is available at [34] and CUTEst is available at https://github.com/ralna/CUTEst.

References

[1] C. Audet, W. Hare, Derivative-Free and Blackbox Optimization, Springer International Publishing, Gewerbestrasse, Switzerland, 2017. doi:10.1007/978-3-319-68913-5.

[2] A. Brilli, M. Kimiaei, G. Liuzzi, S. Lucidi. Worst case complexity bounds for linesearch-type derivative-free algorithms, 2023, ttps://arxiv.org/abs/2302.05274.

[3] A. R. Conn, K. Scheinberg, L. N. Vicente, Introduction to Derivative-Free Optimization, Society for Industrial and Applied Mathematics, Philadelphia, USA, 2009. doi:10.1137/1.9780898718768.

[4] J. Larson, M. Menickelly, S. M. Wild, Derivative-free optimization methods, Acta Numer. 28 (2019) 287–404. doi:10.1017/s0962492919000060.

[5] L. M. Rios, N. V. Sahinidis, Derivative-free optimization: a review of algorithms and comparison of software implementations, J. Global. Optim. 56 (3) (2012) 1247–1293. doi:10.1007/s10898-012-9951-y.

[6] M. Kimiaei, A. Neumaier, Efficient unconstrained black box optimization, Math. Program. Comput. 14 (2022) 365–414. doi:10.1007/s12532-021-00215-9.

[7] N. I. M. Gould, D. Orban, P. L. Toint, CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization, Comput. Optim. Appl. 60 (2015) 545–557. doi:10.1007/s10589-014-9687-3.

[8] S. Gratton, P. L. Toint, A. Tröltzsch, An active-set trust-region method for derivative-free nonlinear bound-constrained optimization, Optim. Methods Softw. 26 (4-5) (2011) 873–894. doi:10.1080/10556788.2010.549231.

[9] M. J. D. Powell, UOBYQA: unconstrained optimization by quadratic approximation, Math. Program. 92 (3) (2002) 555–582. doi:10.1007/s101070100290.

[10] M. J. D. Powell, Developments of NEWUOA for minimization without derivatives, IMA. J. Numer. Anal. 28 (4) (2008) 649–664. doi:10.1093/imanum/drm047.

[11] W. Huyer, A. Neumaier, SNOBFIT – stable noisy optimization by branch and fit, ACM. Trans. Math. Softw. 35 (2) (2008) 1–25. doi:10.1145/1377612.1377613.

[12] C. Elster, A. Neumaier, A grid algorithm for bound constrained optimization of noisy functions, IMA J. Numer. Anal. 15 (4) (1995) 585–608. doi:10.1093/imanum/15.4.585.

[13] W. Huyer, A. Neumaier, Global optimization by multilevel coordinate search, J. Glob. Optim. 14 (4) (1999) 331–355. doi:10.1023/a:1008382309369.

[14] C. Audet, J. Dennis, Jr., Mesh adaptive direct search algorithms for constrained optimization, SIAM J. Optim. 17 (1) (2006) 188–217. doi:doi:10.1137/040603371.

[15] C. Audet, S. Le Digabel, V. Rochon Montplaisir, C. Tribes, The NOMAD project, Software available at `https://www.gerad.ca/nomad/`.

[16] C. Audet, S. Le Digabel, C. Tribes, The Mesh Adaptive Direct Search Algorithm for Granular and Discrete Variables, SIAM J. Optim. 29 (2) (2019) 1164–1189. doi:10.1137/18M1175872.

[17] S. Le Digabel, Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm, ACM. Trans. Math. Softw. 37 (4) (2011) 1–15. doi:10.1145/1916461.1916468.

[18] S. Lucidi, M. Sciandrone, A derivative-free algorithm for bound constrained optimization, Comput. Optim. Appl. 21 (2) (2002) 119–142. doi:10.1023/a:1013735414984.

[19] S. Gratton, C. W. Royer, L. N. Vicente, Z. Zhang, Direct search based on probabilistic descent, SIAM J. Optim 25 (3) (2015) 1515–1541. doi:10.1137/140961602.

[20] M. Porcelli, P. L. Toint, Exploiting problem structure in derivative free optimization, ACM. Trans. Math. Softw. 48 (1) (2022) 1–25. doi:10.1145/3474054.

[21] N. J. Higham, Optimization by direct search in matrix computations, SIAM J. Matrix Anal. Appl. 14 (2) (1993) 317–333. doi:10.1137/0614023.

[22] A. Auger, N. Hansen, A restart CMA evolution strategy with increasing population size, in: 2005 IEEE Congress on Evolutionary Computation, IEEE, Edinburgh, UK, 2005. doi:10.1109/cec.2005.1554902.

[23] I. Loshchilov, T. Glasmachers, H. G. Beyer, Large scale black-box optimization by limited-memory matrix adaptation, IEEE Trans. Evol. Comput. 23 (2) (2019) 353–358. doi:10.1109/tevc.2018.2855049.

[24] H. G. Beyer, Design principles for matrix adaptation evolution strategies, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020, pp. 682–700. doi:10.1145/3377929.3389870.

[25] H. G. Beyer, B. Sendhoff, Simplify your covariance matrix adaptation evolution strategy, IEEE Trans. Evol. Comput. 21 (5) (2017) 746–759. doi:10.1109/tevc.2017.2680320.

[26] E. H. Bergou, E. Gorbunov, P. Richtárik, Stochastic three points method for unconstrained smooth minimization, SIAM J. Optim. 30 (4) (2020) 2726–2749. doi:10.1137/19m1244378.

[27] A. S. Bandeira, K. Scheinberg, L. N. Vicente, Convergence of trust-region methods based on probabilistic models, SIAM J. Optim 24 (3) (2014) 1238–1264. doi:10.1137/130915984.

[28] R. Chen, Stochastic derivative-free optimization of noisy functions, Ph.D. thesis, Lehigh University, theses and Dissertations. 2548 (2015). `https://preserve.lehigh.edu/etd/2548`

[29] K. J. Dzahini, Expected complexity analysis of stochastic direct-search, Comput. Optim. Appl. 81 (1) (2021) 179–200. doi:10.1007/s10589-021-00329-9.

[30] J. Blanchet, C. Cartis, M. Menickelly, K. Scheinberg, Convergence rate analysis of a stochastic trust-region method via supermartingales, INFORMS J. Comput. 1 (2) (2019) 92–119. doi:10.1287/ijoo.2019.0016.

[31] A. S. Berahas, R. H. Byrd, J. Nocedal, Derivative-free optimization of noisy functions via quasi-newton methods, SIAM J. Optim. 29 (2) (2019) 965–993. doi:10.1137/18m1177718.

[32] A. S. Berahas, L. Cao, K. Scheinberg, Global convergence rate analysis of a generic line search algorithm with noise, SIAM Journal on Optimization 31 (2) (2021) 1489–1518. doi:10.1137/19m1291832.

[33] M. Kimiaei, A. Neumaier, Testing and tuning optimization algorithm, in preparation (2023).

[34] M. Kimiaei, The VRDFON solver, software available at `https://github.com/GS1400/VRDFON` (2022).

[35] J. J. Moré, S. M. Wild, Benchmarking derivative-free optimization algorithms, SIAM J. Optim. 20 (1) (2009) 172–191. doi:10.1137/080724083.

[36] E. D. Dolan, J. J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2) (2002) 201–213. doi:10.1007/s101070100263.

[37] N. A. Dong, D. J. Eckman, X. Zhao, S. G. Henderson, M. Poloczek, Empirically comparing the finite-time performance of simulation-optimization algorithms, in: 2017 Winter Simulation Conference (WSC), IEEE, 2017. doi:10.1109/wsc.2017.8247952.

[38] R. C. H. Cheng, J. P. C. Kleijnen, Improved design of queueing simulation experiments with highly heteroscedastic responses, Operations Research 47 (5) (1999) 762–777. doi:10.1287/opre.47.5.762.

[39] E. J. Anderson, M. C. Ferris, A direct search algorithm for optimization with noisy function evaluations, SIAM J. Optim. 11 (3) (2001) 837–857. doi:10.1137/s1052623496312848.

[40] S. Shashaani, F. S. Hashemi, R. Pasupathy, ASTRO-DF: A class of adaptive sampling trust-region algorithms for derivative-free stochastic optimization, SIAM J. Optim. 28 (4) (2018) 3145–3176. doi:10.1137/15m1042425.

[41] R. R. Barton, J. S. Ivey Jr, Nelder-mead simplex modifications for simulation optimization, Manag. Sci. 42 (7) (1996) 954–973. doi:https://doi.org/10.1287/mnsc.42.7.954.

[42] K. H. Chang, L. J. Hong, H. Wan, Stochastic trust-region response-surface method (STRONG)—a new response-surface framework for simulation optimization, INFORMS J. Comput. 25 (2) (2013) 230–243. doi:10.1287/ijoc.1120.0498.