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

VRDFON – line search in noisy unconstrained derivative-free optimization

Morteza Kimiaei

Received: date / Accepted: date

Abstract In this paper, a new randomized solver (called VRDFON) for noisy unconstrained derivative-free optimization problems is discussed. Complexity bounds in the presence of noise for nonconvex, convex, and strongly convex functions are 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. A recommendation is made as to which solvers are robust and efficient based on dimension and noise level. It turns out that VRDFON is more robust and efficient than the state-of-the-art solvers, especially for medium and high dimensions.

Keywords Noisy derivative-free optimization · heuristic optimization · randomized line search method · complexity bounds · sufficient decrease

Mathematics Subject Classification (2020) 90C15 · 90C30 · 90C56

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. Thanks also to Giampaolo Liuzzi for preparing a MEX file for calling UOBYQA from Matlab and to Arnold Neumaier.

M. Kimiaei
Faculty of Mathematics, University of Vienna, Oskar-Morgenstern-Platz 1, A-1090, Vienna, Austria
E-mail: kimiaeim83@univie.ac.at

1 Introduction

We consider the problem of finding a minimizer of the smooth real-valued function $f : \mathbb{R}^n \to \mathbb{R}$

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

Here f 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 noise. This problem is called the noisy derivative-free optimization (DFO) problem. We denote by $g(x)$ the unknown exact gradient vector of f at x and by $\tilde{g}(x)$ its approximation, and by $B(x)$ the Hessian matrix of f at x and by $\tilde{B}(x)$ its approximation. The algorithm uses no knowledge of g, the Lipschitz constants of f, the structure of f, and 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).

A *complexity bound* of an algorithm for the noisy DFO problem (1) is an upper bound on the number of function evaluations to find an approximate point near a local optimizer whose unknown exact gradient norm is below a given fixed threshold $\omega > 0$ (which is unknown to us but appears in our complexity bound) and whose function value is as small as possible compared to the initial function value. In practice, such a point is unknown because the Lipschitz constants and gradients are unknown. However, the inexact function value of this point is equal to or better than a point whose gradient is small only near a global optimizer. As mentioned above, to solve the noisy DFO problem (1), although our algorithm does not require knowledge of the true gradient, the Lipschitz constants of the objective function, the structure of the objective function, and the statistical properties of noise, to obtain such a complexity bound, we assume 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 $\mathcal{L}(x^0) := \{x \in \mathbb{R}^n \mid f(x) \le f(x^0)\}$ of f at x^0 is compact,
(A3) any noise estimate $\tilde{f}(x)$ of f at $x \in \mathbb{R}^n$ satisfies

$$|\tilde{f}(x) - f(x)| \le \omega. \tag{2}$$

In the noiseless case $\omega = 0$ (A3) implies $\tilde{f} = f$. (A2) implies that

$$\widehat{f} := \inf\{f(x) \mid x \in \mathbb{R}^n\} = f(\widehat{x}) > -\infty \tag{3}$$

for any global minimizer $\widehat{x}$ of (1). Other noisy DFO methods assumed these assumptions to obtain their limit accuracy and complexity results; for more details, see Section 1.2 for relevant references. For further references, see, e.g., Bergou et al. [9], Gratton et al. [22], and Kimiaei & Neumaier [33], who used (A1) and (A2) to obtain the complexity results of their noiseless DFO methods.

To determine which solvers are competitive for the noisy DFO problem (1), two main tools are used: *robustness* (highest number of problems solved) and *efficiency* (lowest relative cost of function evaluations). The data profile of Moré & Wild [38] and the performance profile of Dolan & Moré [19] are used to compare these solvers in terms of robustness and efficiency, respectively. A solver with the highest number of solved problems is called *robust* and with the lowest relative cost of function evaluations is called *efficient*. In fact, a solver is competitive if it is efficient and robust.

1.1 Related work

There are many efficient and robust methods for the noisy DFO problem (1), e.g., see the books Audet & Hare [5] and Conn et al. [14]. Larson et al. [34] have discussed these methods and their complexity bounds (if any). These methods are based on line search, direct search, model-based, etc., and are either deterministic or randomized or both. We focus here on
• line search methods, e.g., see [34, Section 2.3.4] (Grippo & Rinaldi [25], Grippo & Sciandrone [26], Lucidi & Sciandrone [36], and Neumaier et al. [39]),
• model-based methods, e.g., see [34, Section 2.2] (Bandeira et al. [6], Buhmann [12], Conn & Toint [15], Gratton et al. [22,23], Gratton et al. [24], Powell [41,42], Huyer & Neumaier [29], and Wild et al. [46]),
• randomized methods, e.g., see [34, Section 3.2] (Bandeira et al. [6], Diniz-Ehrhardt et al. [18], Gratton et al. [22,23], and Van Dyke & Asaki [45]).

Two good references for studying the efficiency and robustness of these solvers are Rios & Sahinidis [43] and Kimiaei & Neumaier [33]. As a result of their numerical results, some efficient and robust solvers are:
• Line search: `VRBBO` (randomized) by Kimiaei & Neumaier [33], `SDBOX` (deterministic) by Lucidi & Sciandrone [36], `FMINUNC` (Wolfe conditions along standard BFGS directions) by Matlab Optimization Toolbox.
• Nelder–Mead: `NMSMAX` by Higham [27].
• Direct search: `NOMAD` by Abramson et al. [1] (an implementation of Audet & Dennis [3] and the mesh adaptive direct search using orthogonal directions Abramson et al. [2]), `DSPFD` (randomized) by Gratton et al. [22], `BFO` by Porcelli & Toint [40], and `MCS` (multi coordinate search) by Huyer & Neumaier [28].
• Model-based: `BCDFO` by Gratton et al. [24], `UOBYQA` and `NEWUOA` by Powell [41,42], and `SnobFit` by Huyer & Neumaier [29].
• Covariance matrix adaptation evolution: `CMAES` (stochastic) by Auger & Hansen [4].

Other covarince matrix adaptation evolution solvers are `LMMAES` (limited memory) by Loshchilov et al. [35], `fMAES` (fast) by Beyer [10], and `BiPopMAES`

(Bi-Population) by Beyer & Sendhoff [11]. `GRID` by Elster & Neumaier [20] is another model-based solver for noisy derivative-free optimization problems with bound constrained.

We here discuss the advantages and disadvantages of the above solvers:
• Model-based solvers are only effective for problems in low dimensions in the presence of noise, but they cannot handle problems in medium to high dimensions because $n(n+3)/2$ sample points are needed to construct fully quadratic models.
• Line search solvers (`VRBBO` and `SDBOX`) are more efficient and robust than direct search solvers because they use extrapolations, which are an accelerated component of these line search methods. Extrapolations expand step sizes along with fixed directions until the inexact function values are improved. Both direct search and line search solvers can handle problems in small to large dimensions. Although another line search solver, `FMINUNC`, is effective in the noiseless case, it is numerically inefficient in the presence of noise because a finite difference technique for estimating the gradient leads to misleading information and poor quasi-Newton directions (for similarities and differences of our line search based algorithm with the mentioned solvers, see Sections 2.1 and 2.2).
• Nelder–Mead solvers are effective for small scale problems; however, they can also handle problems in medium dimensions, although their efficiency and robustness decrease with increasing dimension, e.g., Torczon [44] and Wright [47].
• Loshchilov et al. [35] discussed and showed that solvers based on full covariance matrix adaptation evolution strategies are costly for large problems because the covariance matrices are stored. Therefore, they proposed a limited memory covariance matrix adaptation evolution method (`LMMAES`) for large scale problems. However, this method is not effective for large scale problems because it ignores some parts of the covariance matrix, see Section 7.5.3.

1.2 Known limit accuracy and complexity bounds

In this section, we discuss the achievable limit accuracy and complexity bounds of several well-known noisy DFO methods under standard assumptions (A1)-(A3).

Given a positive scaling vector $s \in \mathbb{R}^n$ (fixed in throughout the paper), we define the scaled 2-norm $\|p\|$ of $p \in \mathbb{R}^n$ and the dual norm $\|g\|_*$ of $g \in \mathbb{R}^n$ by

$$\|p\| := \sqrt{\sum_i p_i^2/s_i^2} \quad \text{and} \quad \|g\|_* := \sqrt{\sum_i s_i^2 g_i^2}.$$

For the noiseless case, see LARSON et al. [34, Table 8.1] for a summary of known results on worst-case complexity and corresponding references. To obtain $\|g(x)\|_* \le \varepsilon$ (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 [6].

In the presence of noise, the limit accuracy of some algorithms was investigated:
• For the *unconstrained case*, BERAHAS et al. [7] proved convergence results for the problem (1) when f is *strongly convex*. Assuming strong convexity of f and boundedness of noise in the approximation gradient, they proved that a quasi-Newton method with a fixed step size has linear convergence to a neighborhood of the solution; the gradient is estimated by the forward or central finite differences. Under the additional assumption (A3), they showed that a quasi-Newton method with step sizes found by a relaxed Armijo line search, called `FDLM`, has asymptotic accuracy

$$f - \widehat{f} = \mathcal{O}(L\omega). \tag{4}$$

CHEN [13] proposed a randomized algorithm with Gaussian directions and estimated step sizes, called `STRRS` for various types of noise, one of which is discussed here. Under the assumptions
(i) $\tilde{f}(x) - f(x) = \omega(x;\zeta)$ is a stochastic noise component, where ζ is a random vector with probability distribution $\Pr(\zeta)$,
(ii) for all $x \in \mathbb{R}^n$, ω is *independent and identically distributed* (*i.i.d.*) with bounded variance $\mathrm{var}(\omega) > 0$,
(iii) for all $x \in \mathbb{R}^n$, noise is unbiased, i.e., $\mathbf{E}_\zeta(\omega) = 0$,
(iv) f is *convex* and (A1) holds,
(v) $\mathrm{var}(\omega) \le \mathcal{O}(\varepsilon/n)$,
`STRRS` needs at most $\mathcal{O}(nL\varepsilon^{-1})$ to ensure that

$$x^N := \operatorname*{argmin}_x \{f(x) \mid x \in \{x^0, \cdots, x^N\}\}$$

satisfies $\mathbf{E}[f(x^N)] - \widehat{f} \le \varepsilon$.
• For the *bound constrained case*, ELSTER & NEUMAIER [20] introduced a grid algorithm, called `GRID`. Here we restrict their results to the unconstrained case. Under the assumptions (A1)–(A3), [20, Theorem 2] ensures that there exists a constant C_n such that

$$\|g(x^k)\|_* \le C_n(2\omega/h^k + Lh^k) \quad \text{for } x^k \in \mathbb{R}^n$$

at the end of the kth refinement step, where h^k denotes the kth grid size. If $h^k := \Theta(\sqrt{\omega})$, then the best order of magnitude can be obtained at least for a point with the gradient

$$\|g\|_* = \mathcal{O}(C_n\sqrt{\omega}). \tag{5}$$

The dependence of C_n on n is not specified. Under the same assumptions, Lucidi & Sciandrone [36] constructed a derivative-free line search algorithm, called `SDBOX`, using only the coordinate directions. They proved that, for any k,

$$\|g(x^k)\|_* = \mathcal{O}\Big(n^{3/2}L\mathbf{a}^k_{\max} + \frac{n\omega}{\mathbf{a}^k_{\min}}\Big), \quad \text{for } x^k \in \mathbb{R}^n.$$

Here $\mathbf{a}^k_{\min}$ and $\mathbf{a}^k_{\max}$ are minimum and maximum values, respectively, for the n step sizes used along the coordinate directions in the iteration k. If $\mathbf{a}^k_{\max} := \Theta(\sqrt{\omega})$, the best order of magnitude can be obtained at least for a point with the gradient

$$\|g\|_* = \mathcal{O}\Big(n^{3/2}\sqrt{\omega}\Big). \tag{6}$$

The order of ω in (6) is the same as in (5) but their factors are different.

As stated in the introduction, $\tilde{g}(x)$ stands for the estimated gradient at x. Berahas et al. [8] used the line search discussed in [7]

$$\tilde{f}(x+\alpha p) \le \tilde{f}(x) + c_1\alpha p^T\tilde{g}(x) + 2\omega \quad \text{with } 0 < c_1 < 1, \tag{7}$$

but step sizes are updated in a different way. Here $\tilde{f}(x) - f(x) = \omega(x;\zeta)$ is stochastic where ζ may be either dependent, independent or identical. Under the assumptions (A1)–(A3) (neither stochastic noise nor reduction or control of noise is assumed) and

$$\text{norm condition: } \|\tilde{g}(x) - g(x)\| \le \theta\|g(x)\|, \quad \text{for some } 0 < \theta < 1, \tag{8}$$

they found for all cases (nonconvex, convex, and strongly convex) that the expected complexity results for a given accuracy $\varepsilon > 0$ (sufficiently larger than ω) exceed a near optimal neighborhood to noise. In the general case, $\mathcal{O}(\varepsilon^{-2})$ function evaluations with $\mathbf{E}(\|g\|_*) \le \varepsilon$ are used in the worst case. In the convex and strongly convex cases, $\mathcal{O}(\varepsilon^{-1})$ and $\mathcal{O}(\log\varepsilon^{-1})$ function evaluations with $\mathbf{E}(\|g\|_*) \le \varepsilon$ and $\mathbf{E}(f - \widehat{f}) \le \varepsilon$, respectively, are used in the worst case.

2 An overview of our method

We propose a new randomized solver for noisy unconstrained DFO problems – called *Vienna noisy randomized derivative-free optimization* (`VRDFON`). Its basic version is called `VRDFON-basic`. Following the classifications of Larson et al. [34] and Rios & Sahinidis [43], our new solver `VRDFON` is a local model-based randomized solver.

The VRDFON package is publicly available at [31]. Supplemental information can be found in suppMat.pdf on the VRDFON web site. suppMat.pdf includes new practical enhancement and details of the codes compared.

2.1 Similarity of VRDFON with other solvers

(i) (*Algorithmic*) VRDFON is an adaptation of our recent solver VRBBO (Kimiaei & Neumaier [33]) to the noisy case, while retaining the main structure of VRBBO, namely a multi-line search algorithm.
(ii) (*Practical enhancement*) VRDFON uses only one of the practical enhancements of VRBBO, namely random subspace directions.
(iii) (*Complexity results*) In all cases, the order of ω in our bounds is the same as in Berahas et al. [8].

2.2 Difference between VRDFON and other solvers

(i) (*Algorithmic*) VRDFON repeatedly calls an improved decrease search called DS (implemented in Section 1.8 of suppMat.pdf) whose basic version is DS-basic (described in Section 4). DS uses an improved multi-line search algorithm called MLS (implemented in Section 1.7 of suppMat.pdf) that is likely to reduce function values, whose basic version is MLS-basic (described in Section 4). MLS uses heuristics to find and update step sizes that are significantly different from the way step sizes are updated in other solvers, e.g., SDBOX, VRBBO, and FMINUNC. After a few calls to MLS by DS, without decreasing the inexact function value, step sizes may become too small if noise is high. All derivative-free line search methods have this drawback when noise is high. To remedy this, DS *reconstructs step sizes heuristically* (lines 29-38 of DS in Section 1.8 of suppMat.pdf).
(ii) (*New practical enhancements*) Unlike VRBBO, SDBOX, and FMINUNC, VRDFON uses many new practical enhancements (Section 1 of suppMat.pdf). This solver constructs surrogate quadratic models in adaptively determined subspaces that can handle medium and large scale problems (Section 1.4 of suppMat.pdf). Although these models have lower accuracy in higher dimensions, their usefulness has been confirmed in extensive numerical experiments in the presence of strong noise. MLS is performed along the new directions, either *random approximate coordinate* (Section 1.1 of suppMat.pdf), *perturbed random* (Section 1.5 of suppMat.pdf) or *improved trust region directions* (Section 1.6 of suppMat.pdf):
• 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. [6]); 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.
(iii) (*Complexity results* for `VRDFON-basic`) For `VRDFON-basic` that uses only scaled random directions and no practical enhancements, we prove the complexity results *with probability arbitrarily close to one* for nonconvex, convex, and strongly convex functions in the presence of noise. In contrast to the method of Berahas et al. [8], which uses the norm condition (8), our line search does not use the term $c_1\alpha p^T\tilde{g}(x)$ of the condition (7), 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*, e.g., see behaviour of `FMINUNC` in Section 7. However, we estimate the gradient to generate different heuristic directions in Section 1 of `suppMat.pdf`. Therefore, we obtain our complexity bound regardless of the norm condition (8) since the nature of the line search algorithms is different. On the other hand, our bounds are obtained with high probability. Therefore, they are more likely to be appropriate and differ from the results of Berahas et al. [8], which are valid only in expectation.
(iv) (*Complexity results for* `VRDFON`) To obtain the complexity results for `VRDFON` (implemented version), random scaled directions should be used; this is not a problem to guarantee the complexity results when other directions are used. In fact, the order of the complexity bounds of `VRDFON` does not change for all cases after applying the heuristic improvements, although the constant factors may be become larger, as in the deterministic cases (as mentioned in Section 2.2(ii) the complexity of deterministic DFO methods is in the worst case by a factor n worse than that of randomized methods). However, the robustness and efficiency of `VRDFON` are numerically increased.

2.3 Organization

In Section 3 we discuss how to generate scaled random directions. Then, in Section 4, we describe a basic version of `VRDFON` using scaled random directions. Complexity results of a basic version of `VRDFON` for all cases with a given probability arbitrarily close to one in the presence of noise are proved in Section 5. Complexity results of `VRDFON` (implemented version) are discussed in Section 6. Section 7 provides a comparison between `VRDFON` and the solvers discussed in Section 1.1 on the 549 unconstrained `CUTEst` test problems from the collection of Gould et al. [21] and *makes a recommendation as to which solvers are robust and efficient based on dimension and noise level.* It turns out that `VRDFON` is more robust and efficient, especially for medium and high dimensions.

3 Search direction

In this section, we describe how to generate random directions. Then it is shown that these directions satisfy the two-sided angle condition (defined below) with probability at least half.

We define a standard random direction as a random direction p drawn uniformly *i.i.d.* in $[-\frac{1}{2}, \frac{1}{2}]^n$. As explained in the introduction, i.i.d. stands for independent and identically distributed. A *scaled random direction* is a standard random direction p scaled by $\gamma_{\rm rd}/\|p\|$, where $0 < \gamma_{\rm rd} < 1$ is a tiny tuning parameter, resulting in $\|p\| = \gamma_{\rm rd}$.

The scaling of the direction p by $\gamma_{\rm rd}$ is the same as the scaling of the direction p by δ in [33, (17)]. Therefore, our scaled random direction is the same as the scaled random direction obtained by [33, (17)]. Thus, the next result holds for any scaling vector $s \in \mathbb{R}^n$ (defined in Section 1.2), although we do not intend to estimate s in this paper; it is assumed that $s_i = 1$ for $i = 1, 2, \cdots, n$.

Essential for our complexity bounds is the following result (Proposition 2 in [33]) for the unknown gradient $g(x)$ of $f(x)$ at $x \in \mathbb{R}^n$.

Proposition 1 *Any scaled random direction p satisfies the inequality*

$$\Pr(\|g(x)\|_* \|p\| \leq 2\sqrt{cn}|g(x)^T p|) \geq \frac{1}{2} \tag{9}$$

with a positive constant $c \leq 12.5$. The approximate value for the constant c has been discussed in [33, Section 9.1].

The condition (9) 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 `VRDFON-basic`, a basic version of `VRDFON`

This section discusses a basic randomized method for the noisy DFO problem (1), called `VRDFON-basic`.

`VRDFON-basic` counts the number $k \in \mathbb{N}_0 := \mathbb{N} \cup \{0\}$ of calls to `DS-basic` and the number $t \in \{1, 2, \cdots, T_0\}$ of calls to `MLS-basic`. Here $1 \leq T_0 < \infty$ is a tuning parameter. Moreover, `MLS-basic` counts the number $r \in \{1, 2, \cdots, R_m\}$ of scaled random directions. The upper bound ($2 \leq R_m < \infty$) of this number is a tuning parameter. To simplify our algorithms, once all tuning parameters are given in line 1 of `VRDFON-basic`, not mentioned as input.

`MLS-basic` counts the number $n_{\texttt{succ}}^{\texttt{MLS}}$ of reductions in the inexact function values, while `DS-basic` counts the number $n_{\texttt{succ}}^{\texttt{DS}}$ of reductions in the inexact function values.

An iteration of `MLS-basic` is called *successful* if a reduction of the inexact function value is found and *unsuccessful* otherwise. If after R_m iterations ($n_{\texttt{succ}}^{\texttt{MLS}} = 0$) no reduction of the inexact function value is found, `MLS-basic` is called *inefficient* and *efficient* otherwise.

An iteration of `DS-basic` is called *successful* if `MLS-basic` is efficient ($n_{\texttt{succ}}^{\texttt{MLS}} > 0$) and *unsuccessful* otherwise. `DS-basic` is called *efficient* if it has at least one successful iteration ($n_{\texttt{succ}}^{\texttt{DS}} > 0$) and *inefficient* otherwise. An iteration of `VRDFON-basic` is called *successful* if `DS-basic` is efficient ($n_{\texttt{succ}}^{\texttt{DS}} > 0$); and *unsuccessful* otherwise.

We denote by x_{best} the *overall best point* and by $\tilde{f}_{\text{best}} := \tilde{f}(x_{\text{best}})$ the *overall best inexact function value* of `VRDFON-basic`, i.e., the final best point and its inexact function value found by `DS-basic`. Indeed, the overall best point is an ε-approximate stationary point of the sequence x^k $(k = 1, 2, 3, \cdots)$ after `VRDFON-basic` terminates at a finite number of iterations.

`VRDFON-basic` initializes the number n_{f} of function evaluations and the initial step size $\delta_0 := \delta_{\max}$, which is a tuning parameter, and computes the inexact function value $\tilde{f}(x^0)$ at the initial point x^0 in line 2. In each iteration, `VRDFON-basic` calls `DS-basic` to hopefully find a reduction of the inexact function value (line 4). Once the step size δ_k is below a minimum threshold $0 \le \delta_{\min} < 1$ (line 5), which is a tuning parameter, `VRDFON-basic` terminates with the overall best point $x_{\text{best}} := x^{k+1}$ and its inexact function value $\tilde{f}(x_{\text{best}}) := \tilde{f}(x^{k+1})$ in the $(k+1)$th iteration. Otherwise, if `DS-basic` cannot find a reduction of the inexact function value (i.e., $n_{\texttt{succ}}^{\texttt{DS}} = 0$) the step size δ_{k+1} is reduced by a factor of $Q > 1$ (line 7), which is a tuning parameter; otherwise, δ_{k+1} is δ_k (line 9). `VRDFON-basic` tries to find an ε-approximate stationary point x_{best} that satisfies $\|g(x_{\text{best}})\|_* \le \varepsilon$ for a threshold $\varepsilon = \sqrt{\omega}$ before the condition $\delta_k \le \delta_{\min}$ is satisfied. When $\delta_{\min} = 0$ and since the gradient $g(x_{\text{best}})$ is unknown, `nfmax` is required as an upper bound on the number of function evaluations for a finite termination.

If the $(k+1)$th iteration of `VRDFON-basic` is unsuccessful ($n_{\texttt{succ}}^{\texttt{DS}} = 0$), then $x^{k+1} = x^k$ and $\tilde{f}(x^{k+1}) = \tilde{f}(x^k)$. If $n_{\texttt{succ}}^{\texttt{DS}} > 0$, which is an output of `DS-basic` (line 4), then the $(k+1)$th iteration of `VRDFON-basic` is successful.

`DS-basic` generates the sequences y^t and $\tilde{f}(y^t)$ $(t = 1, \cdots, T_0)$. First, it initializes the number $n_{\texttt{succ}}^{\texttt{DS}}$ of successful iterations of `DS-basic` in line 12 and then

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

1: **Tuning paramters:** $Q > 1$ (factor for reducing δ_k), $0 < \gamma_{\rm rd} < 1$ (parameter for scaling random directions), $0 < \gamma < 1$ (parameter for line search), $\gamma_e > 1$ (factor for updating step size inside `MLS-basic`), $\delta_{\max} > 0$ (initial value for δ_k), $0 \leq \delta_{\min} \leq 1$ (minimum threshold for δ_k), $0 < \eta < \frac{1}{2}$ (parameter for R_m), $R_m := \lceil \log_2 \eta^{-1} \rceil \geq 2$ (number of random direction in each `MLS-basic`), $T_0 \geq 1$ (number of calls to `MLS-basic` by `DS-basic`).

2: Set $n_{\rm f} := 1$ and $\delta_0 := \delta_{\max}$. Then compute $\tilde{f}(x^0)$;
3: **for** $k = 0, 1, 2, \cdots$ **do**
4: run $[x^{k+1},\ \tilde{f}(x^{k+1}),\ n^{\tt DS}_{\tt succ},\ n_{\rm f}]$ =`DS-basic`($\delta_k,\ x^k,\ \tilde{f}(x^k),\ n_{\rm f},\ $ `nfmax`);
5: **if** $\delta_k \leq \delta_{\min}$, set $x_{\rm best} := x^{k+1}$ and $\tilde{f}_{\rm best} := \tilde{f}(x^{k+1})$; **stop**; **end if**
6: **if** $n^{\tt DS}_{\tt succ}$ is zero **then** ▷ the kth iteration is *unsuccessful*
7: set $\delta_{k+1} := \delta_k/Q$; ▷ δ_{k+1} is reduced
8: **else** ▷ the kth iteration is *successful*
9: set $\delta_{k+1} := \delta_k$; ▷ δ_{k+1} remains fixed
10: **end if**
11: **end for**

initializes two sequences y^t and $\tilde{f}(y^t)$ ($t = 1, \cdots, T_0$). Subsequently, `DS-basic` has T_0 calls to `MLS-basic` to find reductions of the inexact function values in lines 14-17, while counting the number $n^{\tt DS}_{\tt succ}$ of successful iterations in line 16. If $n^{\tt DS}_{\tt succ} = 0$, then $x^{k+1} = y^{T_0} = x^k$ and $\tilde{f}(x^{k+1}) = \tilde{f}(y^{T_0}) = \tilde{f}(x^k)$ (line 18). Otherwise, at least one reduction of the inexact function value is found and so $x^{k+1} = y^{T_0} \neq x^k$ and $\tilde{f}(x^{k+1}) = \tilde{f}(y^{T_0}) < \tilde{f}(x^k)$ (line 18).

function
$[x^{k+1},\ \tilde{f}(x^{k+1}),\ n^{\tt DS}_{\tt succ},\ n_{\rm f}] =$ `DS-basic`($\delta_k,\ x^k,\ \tilde{f}(x^k),\ n_{\rm f},\ $ `nfmax`)

12: Initialize $n^{\tt DS}_{\tt succ} := 0$; ▷ the number of successful iterations of `DS-basic`
13: set $y^0 := x^k$ and $\tilde{f}(y^0) := \tilde{f}(x^k)$;
14: **for** $t = 1, \cdots, T_0$ **do**
15: run $[y^t,\ \tilde{f}(y^t), n^{\tt MLS}_{\tt succ},\ n_{\rm f}] =$ `MLS-basic`($\delta_k,\ y^{t-1},\ \tilde{f}(y^{t-1}),\ n_{\rm f},\ $ `nfmax`);
16: **if** $n^{\tt MLS}_{\tt succ} > 0$ **then** set $n^{\tt DS}_{\tt succ} := n^{\tt DS}_{\tt succ} + 1$; **end if**
17: **end for**
18: set $x^{k+1} := y^{T_0}$ and $\tilde{f}(x^{k+1}) := \tilde{f}(y^{T_0})$;

`MLS-basic` generates the sequences z^r and $\tilde{f}(z^r)$ ($r = 1, \cdots, R_m$). It initializes the extrapolation step size in line 19 and the number $n^{\tt MLS}_{\tt succ}$ of successful iterations in line 20, then initializes the point $z_{\rm best}$ (found in the $(t-1)$th iteration of `DS-basic`) and its inexact function value $\tilde{f}(z_{\rm best})$ in line 21. `MLS-basic` has a for loop (lines 22-41) that includes a while loop (lines 24-34). In line 23, the rth scaled random direction p^r is computed and the Boolean variable `good` is indicated (`good`=false). We discuss this further below. Then the while loop begins, containing *extrapolation*, which increases the convergence speed to achieve an $\varepsilon = \sqrt{\omega}$-approximate stationary point. The goal of extrapolation is to expand the *extrapolation step sizes* α_r by a tuning parameter $\gamma_e > 1$ (line 29) along the fixed direction p^r and compute the corresponding trial point $z^r = z_{\text{best}} + \alpha_r p^r$ and its inexact function value $\tilde{f}(z^r)$ (line 25), as long as the condition

$$\tilde{f}(z_{\text{best}}) < \tilde{f}(z^r) - \gamma\alpha_r^2 \tag{10}$$

is satisfied (line 28) and the maximum number `nfmax` of function evaluations is not reached, where $0 < \gamma < 1$ is a tuning parameter. Define a *gain* by $\tilde{f}(x) - \tilde{f}(z^r)$. Since this gain along p^r is at least $\gamma\alpha_r^2$, we say that $\gamma\alpha_r^2$-*sufficient gain* along p^r was found, which means that the Boolean variable `good` is indicated to be true (line 29). The rth iteration of `MLS-basic` is called *successful* if `good` is true and *unsuccessful* otherwise. If the condition (10) is not satisfied (line 30), the rth opposite direction is chosen in line 31, since no $\gamma\alpha_r^2$-sufficient gain is found along p^r unless it has already been used. Otherwise, the condition (10) along $\pm p^r$ is not satisfied and thus no $\gamma\alpha_r^2$-sufficient gain is found along $\pm p^r$ (line 32). In this case, the while loop is stopped. Then, if `good` is true (line 35), since the extrapolation was stopped, the last point generated by the extrapolation does not provide $\gamma\alpha_r^2$-sufficient gain along p^r, we set $\alpha_{r+1} = \alpha_r/\gamma_e$ to obtain the corresponding step size of the penultimate point $z_{\text{best}} := z_{\text{best}} + \alpha_r p^r$ (line 36), while its inexact function value $\tilde{f}(z_{\text{best}}) := \tilde{f}_e^r$ is updated and the number $n_{\texttt{succ}}^{\texttt{MLS}}$ of successful iterations is updated by `MLS-basic` (line 37). Otherwise, if `good` is false (line 38), the extrapolation step size is reduced by γ_e in line 39. Finally, after the for loop is terminated, y^t and $\tilde{f}(y^t)$ are updated in line 42.

function $[y^t,\ \tilde f(y^t), n^{\texttt{MLS}}_{\texttt{succ}},\ n_{\rm f}]$ =`MLS-basic`$(\delta_k,\ y^{t-1},\ \tilde f(y^{t-1}),\ n_{\rm f},$ `nfmax`$)$

Initialize the extrapolation step size $\alpha_1 := \delta_k$;
initialize $n^{\texttt{MLS}}_{\texttt{succ}} := 0$; ▷ the number of successful iterations
set $z_{\rm best} := y^{t-1}$ and $\tilde f(z_{\rm best}) := \tilde f(y^{t-1})$;
for $r = 1, \cdots, R_m$ **do**
  compute the scaled random direction p^r and set `good` $:= 0$;
  while true **do**
    compute $z^r := z_{\rm best} + \alpha_r p^r$ and $\tilde f(z^r)$;
    set $n_{\rm f} := n_{\rm f} + 1$; ▷ counting the number of function evaluations
    if $n_{\rm f}$ reaches `nfmax` **then** `VRDFON-basic` terminates; **end if**
    if $\tilde f(z_{\rm best}) - \tilde f(z^r) > \gamma\alpha_r^2$ **then** ▷ $\gamma\alpha_r^2$-sufficient gain along p^r found
      set `good` $:= 1$, $\tilde f^r_e := \tilde f(z^r)$, and $\alpha_r := \gamma_e\alpha_r$;
    else if $-p^r$ has not been tried already **then**
      set $p^r := -p^r$; ▷ opposite direction is tried
    else, **break**; ▷ no $\gamma\alpha_r^2$-sufficient gain along $\pm p^r$
    end if
  end while
  if `good` **then** ▷ the rth iteration of `MLS-basic` is *successful*
    set $\alpha_{r+1} := \alpha_r/\gamma_e$; $z_{\rm best} := z_{\rm best} + \alpha_{r+1}p^r$;
    update $\tilde f(z_{\rm best}) := \tilde f^r_e$ and $n^{\texttt{MLS}}_{\texttt{succ}} := n^{\texttt{MLS}}_{\texttt{succ}} + 1$;
  else ▷ the rth iteration of `MLS-basic` is *unsuccessful*
    reduce the step size to $\alpha_{r+1} := \alpha_r/\gamma_e$;
  end if
end for
set $y^t := z_{\rm best}$ and $\tilde f(y^t) := \tilde f(z_{\rm best})$;

5 Complexity bounds for `VRDFON-basic`

In addition to (A1)-(A3) to obtain our complexity results we assume the following assumption:
(A4) Given a minimum threshold $0 < \alpha_{\min} < \infty$ for step sizes in `MLS-basic` and the tuning parameter $\gamma_e > 1$ to update step sizes, the condition $\gamma_e^{1-R_m}\delta_k \geq \alpha_{\min}$ holds.

Under the assumptions (A1)-(A4), this section discusses how `VRDFON-basic` is terminated after at most

- $\mathcal{O}(R_m T_0 \omega^{-1})$ function evaluations in the general case,
- $\mathcal{O}(\sqrt{n} R_m T_0 \omega^{-1/2})$ function evaluations in the convex case,
- $\mathcal{O}(n R_m T_0 \log(\omega^{-1}))$ function evaluations in the strongly convex case

with an approximate point $\tilde{x}$, with a given probability arbitrarily close to 1, satisfying

$$f(\tilde{x}) \le \sup\{f(x) \mid x \in \mathbb{R}^n, \quad f(x) \le f(x^0), \quad \text{and} \quad \|g(x)\|_* = \mathcal{O}(\sqrt{n\omega})\}.$$

As explained in the introduction, it is not clear to us which point, since the gradients and Lipschitz constants are unknown.

If $R_m = 2$ and $T_0 = 1$ are chosen according to their definitions in line 1 of Algorithm 1, the term $R_m T_0$ asymptotically vanishes from the factors of our bounds (i.e., $R_m T_0 \sim 1$). In this case, our bounds are better by a factor n than in the deterministic case (see BANDEIRA et al. [6] and KIMIAEI & NEUMAIER [33]), but numerically these factors should be large to increase the efficiency and robustness of our algorithm. In practice, our algorithm has better numerical performance if one of $R_m = n$ and $T_0 = n$ is chosen such that $R_m T_0 \sim n$, as in the deterministic case.

In contrast to the method of BERAHAS et al. [8], which uses the norm condition, our line search does not use the term $c_1 \alpha p^T \tilde{g}(x)$ of the condition (7), since the estimate of the gradient may be inaccurate in the presence of high noise, leading to the failure of the line search algorithm, but $\gamma\alpha^2$. Therefore, we are not interested in obtaining our complexity bound under the norm condition (8). We will only use the estimated gradient to generate some heuristic directions in Section 1 of `suppMat.pdf`. In all cases, the order of ω in our bounds is the same as in [8], although the nature of the line search algorithms is different. On the other hand, our bounds are obtained with high probability. Therefore, high probability results are more likely to be appropriate and differ from the results of BERAHAS et al. [8], which are valid only in expectation.

The following result generalizes Proposition 1 in [33]. It is shown that if the rth iteration of `MLS-basic` is unsuccessful, a useful bound for the directional derivative can be found. In this case, no $\gamma\alpha_r^2$-sufficient gain ($r \in \{1, 2, \cdots, R_m\}$) is found along the search directions $\pm p^r$ (`good` is false when these directions are tried).

Proposition 2 *Let $\{z^r\}$ ($r = 1, 2, \cdots, R_m$) be the sequence generated by* `MLS-basic` *in the $(k+1)$th iteration of* `VRDFON-basic`*. Moreover, suppose that (A1)–(A3) hold and $0 < \gamma < 1$. Then, for all $z^r, p^r \in \mathbb{R}^n$, at least one of the following holds:*
(i) $\tilde{f}(z_{\rm best} + \alpha_r p^r) < \tilde{f}(z^r) - \gamma\alpha_r^2$,
(ii) $\tilde{f}(z_{\rm best} + \alpha_r p^r) > \tilde{f}(z^r) + \gamma\alpha_r^2$ *and* $\tilde{f}(z_{\rm best} - \alpha_r p^r) < \tilde{f}(z^r) - \gamma\alpha_r^2$,
(iii) $|g(z^r)^T p^r| \le \gamma\alpha_r + 2\omega/\alpha_r + \dfrac{1}{2} L\alpha_r \|p^r\|^2$.
Here, in the kth iteration of `VRDFON-basic`*, $z_{\rm best}$ is the best point found by* `MLS-basic`*.*

Proof (A1) results in

$$\alpha_r g(z^r)^T p^r - \frac{1}{2} L\alpha_r^2 \|p^r\|^2 \le f(z_{\text{best}} + \alpha_r p^r) - f(z^r) \le \alpha_r g(z^r)^T p^r + \frac{1}{2} L\alpha_r^2 \|p^r\|^2. \tag{11}$$

We assume that (iii) is violated, so that

$$|g(z^r)^T p^r| > \gamma\alpha_r + 2\omega/\alpha_r + \frac{1}{2} L\alpha_r \|p^r\|^2. \tag{12}$$

We consider the proof in the two cases:
CASE 1. If $g(z^r)^T p^r \le 0$, then from (2) and (12) we get

$$\begin{aligned} \tilde{f}(z_{\text{best}} + \alpha_r p^r) - \tilde{f}(z^r) &\le f(z_{\text{best}} + \alpha_r p^r) - f(z^r) + 2\omega \\ &\le \alpha_r g(z^r)^T p^r + \frac{1}{2} L\alpha_r^2 \|p^r\|^2 + 2\omega \\ &= -\alpha_r |g(z^r)^T p^r| + \frac{1}{2} L\alpha_r^2 \|p^r\|^2 + 2\omega < -\gamma\alpha_r^2, \end{aligned} \tag{13}$$

meaning that `good` is true if p^r was tried, hence (13) holds.
CASE 2. If $g(z^r)^T p^r \ge 0$, then from (2) and (12) we get

$$\begin{aligned} \tilde{f}(z_{\text{best}} - \alpha_r p^r) - \tilde{f}(z^r) &\le f(z_{\text{best}} - \alpha_r p^r) - f(z^r) + 2\omega \\ &\le g(z^r)^T (-\alpha_r p^r) + \frac{1}{2} L\alpha_r^2 \|p^r\|^2 + 2\omega \\ &= -\alpha_r |g(z^r)^T p^r| + \frac{1}{2} L\alpha_r^2 \|p^r\|^2 + 2\omega < -\gamma\alpha_r^2, \end{aligned} \tag{14}$$

meaning that `good` is true if $-p^r$ was tried. Therefore, the second inequality in (ii) holds. By (2), (11), and (12) the first half

$$\begin{aligned} \tilde{f}(z_{\text{best}} + \alpha_r p^r) - \tilde{f}(z^r) &\ge f(z_{\text{best}} + \alpha_r p^r) - f(z^r) - 2\omega \\ &\ge \alpha_r g(z^r)^T p^r - \frac{1}{2} L\alpha_r^2 \|p^r\|^2 - 2\omega > \gamma\alpha_r^2 \end{aligned}$$

is obtained, meaning that `good` is false if p^r was tried. Hence the first inequality in (ii) holds. □

As discussed earlier, `VRDFON-basic` has $1 \le K < \infty$ calls to `DS-basic` and `DS-basic` has $1 \le T_0 < \infty$ calls to `MLS-basic`. Hence, `VRDFON-basic` has KT_0 calls to `MLS-basic`. As defined in `VRDFON-basic`, $R_m = \lceil \log_2 \eta^{-1} \rceil$ for a given $0 < \eta < \frac{1}{2}$ is the number of random directions used by `MLS-basic`. For given $1 \le T_0 < \infty$ and $1 \le K < \infty$, defined by

$$R_d := T_0 R_m = T_0 \lceil \log_2 \eta^{-1} \rceil \ge \log_2 \eta^{-T_0} \tag{15}$$

is the number of random directions used by `DS-basic` and defined by

$$R_v := KR_d = KT_0R_m = KT_0\lceil \log_2 \eta^{-1}\rceil \geq \log_2 \eta^{-KT_0} \tag{16}$$

is the number of random directions used by `VRDFON-basic`.

Assuming (A1)-(A4), we find
• for `MLS-basic` at least one point whose unknown gradient norm is below a constant bound (see (17)) with probability $\geq 1-\eta > \frac{1}{2}$ (see Theorem 1);
• for `DS-basic` at least one point whose unknown gradient norm is below a constant bound (see (21)) with probability

$$\geq 1 - 2^{-R_d} = 1 - 2^{-T_0R_m} \geq 1 - \eta^{T_0} \geq 1 - \eta > \frac{1}{2},$$

(see Theorem 2);
• for `VRDFON-basic` at least one point whose unknown gradient norm is below a constant bound (see (26)) with probability

$$\geq 1 - 2^{-R_v} = 1 - 2^{-KT_0R_m} \geq 1 - \eta^{KT_0} \geq 1 - \eta > \frac{1}{2}$$

(see Theorem 3).

The following result is a generalization of Theorem 1 in [33] for `MLS-basic` to the noisy case in the $(k+1)$th iteration of `VRDFON-basic`. As discussed earlier, `MLS-basic` uses R_m scaled random directions p^r $(r = 1, 2, \cdots, R_m)$. If $\alpha_{\min} > 0$, in the worst case, it is proved that one of the following holds:
(i) If at least on iteration ($r' \in \{1, 2, \cdots, R_m\}$) of `MLS-basic` is successful, meaning that `good` is true when $-p^{r'}$ is tried, a minimum reduction in the inexact function value is found (note that if `good` is true when $p^{r'}$ is attempted, then $p^{r'}$ is not attempted and this case is not the worst case, so it is a real case).
(ii) An upper bound on the unknown gradient norm of at least one $z^{r''}$ ($r'' \in \{1, 2, \cdots, R_m\}$) of the points generated by the unsuccessful iterations of `MLS-basic` is found with a given probability arbitrarily close to one. In fact, it is not clear to us which point, since the gradients and Lipschitz constants are not available.

Theorem 1 *Assume that (A1)-(A4) hold,* `nfmax` *is sufficiently large,* $0 < \eta < \frac{1}{2}$, $0 < \gamma_{\rm rd} < 1$, $\gamma_e > 1$, *and* $0 < \gamma < 1$. *Moreover, define* $\overline{L} := 2\gamma/\gamma_{\rm rd} + L\gamma_{\rm rd}$ *and let* $\{z^r\}$ $(r = 1, 2, \cdots, R_m)$ *be the sequence generated by* `MLS-basic` *in the* $(k+1)$*th iteration of* `VRDFON-basic`*. Then one of the following happens:*
(i) If at least `MLS-basic` *has a successful iteration,* $r' \in \{1, 2, \cdots, R_m\}$, *then it decreases the inexact function value by at least* $\gamma\alpha_{r'}^2$.
(ii) If `MLS-basic` *has no successful iteration, then at least one* $z^{r''}$ $(1 \leq r'' \leq$

R_m) of the points evaluated by the unsuccessful iterations of `MLS-basic`*, with the probability at least $1-\eta$, has an unknown gradient $g(z^{r''})$ satisfying*

$$\|g(z^{r''})\|_* \leq \sqrt{cn}\varGamma(\delta_k) \quad with\ \varGamma(\delta_k) := \overline{L}\delta_k + \gamma_e^{R_m-1}\frac{4\omega}{\gamma_{\mathrm{rd}}\delta_k} \tag{17}$$

for a given $0 < \eta < \frac{1}{2}$. Here c comes from Proposition 1, δ_k is fixed in `MLS-basic`*, independent of r, and is updated outside* `MLS-basic`*.*

Proof Let $\mathcal{R} := \{1, \dots, R_m\}$. We denote by p^r the rth scaled random search direction, by z^r the rth point, and by $\alpha_r = \gamma_e^{1-r}\delta_k \geq \alpha_{\min}$ the rth step size from (A4).

(i) Let $r' \in \{1, 2, \cdots, R_m\}$. The worst case requires $2R_m + 1$ function evaluations and assumes that the r'th iteration of `MLS-basic` is successful and the other iterations are unsuccessful. In the unsuccessful iterations, two function values are computed along the directions $\pm p^r$ $(r \in \mathcal{R} \setminus \{r'\})$, but in the r'th iteration which is successful, `good` is false when $p^{r'}$ is attempted and true when $-p^{r'}$ is attempted (as discussed above if `good` is true when $p^{r'}$ is attempted, then $p^{r'}$ is not attempted and this case is not the worst case, so it is a real case). Therefore, an extrapolation step along $-p^{r'}$ is performed with at most two additional function evaluations and the $\gamma\alpha_{r'}^2$-sufficient gain. Consequently, (i) is verified.

(ii) Suppose that $\tilde{f}(z^r)$ does not decrease by more than $\gamma\alpha_r^2$ for all $r \in \mathcal{R}$; all iterations are unsuccessful. Then we define $\varGamma_0(\alpha_r) := \overline{L}\alpha_r + \dfrac{4\omega}{\gamma_{\mathrm{rd}}\alpha_r}$. Since $\varGamma_0(\alpha_r)$ for $\alpha_r > 0$ is a convex function, we obtain for $r \in \mathcal{R}_m$

$$\begin{aligned}\varGamma_0(\alpha_r) &\leq \max\{\varGamma_0(\alpha_1), \varGamma_0(\alpha_R)\} < \overline{L}\alpha_1 + \frac{4\omega}{\gamma_{\mathrm{rd}}\alpha_R} \\ &= \varGamma(\delta_k) = \overline{L}\delta_k + \gamma_e^{R_m-1}\frac{4\omega}{\gamma_{\mathrm{rd}}\delta_k},\end{aligned} \tag{18}$$

where $\alpha_1 := \max\limits_{r\in\mathcal{R}}\{\alpha_r\} = \delta_k > \alpha_R := \min\limits_{r\in\mathcal{R}}\{\alpha_r\} = \gamma_e^{1-R_m}\delta_k$ since $\gamma_e > 1$ and $R_m \geq 2$. Then we obtain from Proposition 2 and since $\|p^r\| = \gamma_{\mathrm{rd}}$ (due to the definition of the scaled random direction in Section 3), for all $r \in \mathcal{R}$,

$$|g(z^r)^T p^r| \leq \gamma\alpha_r + 2\omega/\alpha_r + \frac{L}{2}\alpha_r\|p^r\|^2 = \gamma\alpha_r + 2\omega/\alpha_r + \frac{L}{2}\gamma_{\mathrm{rd}}^2\alpha_r,$$

so that for all $r \in \mathcal{R}$ and from (18), the inequality

$$\begin{aligned}\|g(z^r)\|_* &= \|g(z^r)\|_*\|p^r\|/\gamma_{\mathrm{rd}} \leq 2\sqrt{cn}|g(z^r)^T p^r|/\gamma_{\mathrm{rd}} \\ &\leq \sqrt{cn}\left(\Big(\frac{2\gamma}{\gamma_{\mathrm{rd}}} + L\gamma_{\mathrm{rd}}\Big)\alpha_r + \frac{4\omega}{\gamma_{\mathrm{rd}}\alpha_r}\right) = \sqrt{cn}\varGamma_0(\alpha_r) < \sqrt{cn}\varGamma(\delta_k)\end{aligned}$$

holds with probability $\frac{1}{2}$ or more according to Proposition 1. In other words,

$$\Pr\Big(\|g(z^r)\|_* > \sqrt{cn}\Gamma(\delta_k)\Big) < \frac{1}{2}, \quad \text{for any fixed } r \in \mathcal{R}.$$

Therefore, we find at least one of the gradients $g = g(z^{r''})$ ($r'' \in \mathcal{R}$) such that (17) holds, that is,

$$\begin{aligned}\Pr\Big(\|g\|_* \le \sqrt{cn}\Gamma(\delta_k)\Big) &= 1 - \prod_{r=1}^{R_m} \Pr\Big(\|g(z^r)\|_* > \sqrt{cn}\Gamma(\delta_k)\Big)\\ &\ge 1 - 2^{-R_m} \ge 1 - \eta,\end{aligned}$$

for a given $0 < \eta < \frac{1}{2}$. □

The following result discusses the complexity bound for `DS-basic` in the $(k+1)$th iteration of `VRDFON-basic`. It is proved that either an upper bound on the number of function evaluations is found or an upper bound on the unknown gradient norm of at least one of the points generated by the unsuccessful iterations of `DS-basic` is found with a given probability arbitrarily close to one in the presence of noise; in fact, it is not clear to us which point, since the gradients and Lipschitz constants are not available.

Theorem 2 *Suppose that (A1)–(A4) hold and let $f(x^0)$ be the initial value of f. Moreover, let $\{y^t\}$ ($t = 1, 2, \cdots, T_0$) be the sequence generated by* `DS-basic` *in the $(k+1)$th iteration of* `VRDFON-basic` *and let $0 < \eta < \frac{1}{2}$, $0 < \gamma_{\rm rd} < 1$, $\gamma_e > 1$ and $0 < \gamma < 1$. Then:*
(i) The number of successful iterations of `DS-basic` *is bounded by*

$$\overline{\gamma}^{-1}\delta_k^{-2}\Big(f(x^0) - \widehat{f} + 2\omega\Big), \tag{19}$$

where $\overline{\gamma} := \gamma_e^{2(2-R_m)}\gamma > 0$, $\widehat{f}$ is finite by (A1) and (A2) discussed in Section 1.2, and the step size δ_k is fixed, independent of t, and updated outside `DS-basic`*. Moreover, the number of function evaluations of* `DS-basic` *is bounded by*

$$2R_mT_0 + (2R_m+1)T_0\overline{\gamma}^{-1}\delta_k^{-2}\Big(f(x^0) - \widehat{f} + 2\omega\Big). \tag{20}$$

(ii) Unsuccessful iterations of `DS-basic` *have at least one point $y^{t'}$ ($1 \le t' \le T_0$), with probability at least $\ge 1 - 2^{-R_d} \ge 1 - \eta > \frac{1}{2}$ for a given $0 < \eta < \frac{1}{2}$, satisfying*

$$\|g(y^{t'})\|_* \le \sqrt{cn}\Gamma(\delta_k), \tag{21}$$

where c and $\Gamma(\delta_k)$ come from Proposition 1 and Theorem 1. Here R_d comes from (15).

Proof (i) S denotes the index set of successful iterations of `DS-basic`, where each successful iteration is a result of at least one successful iteration of `MLS-basic`. Let $r' \in \{1, 2, \cdots, R_m\}$. As discussed in the proof of Theorem 1(i), in the worst case at least the r'th iteration of `MLS-basic` is successful that a result of an extrapolation along $-p^{r'}$. We do not know how many times we can extrapolate $\alpha_{r'}$ to γ_e along the fixed direction $-p^{r'}$, but at least once $\alpha_{r'}$ is expanded by γ_e in an extrapolation and therefore at most $R_m - 1$ times $\alpha_1 = \delta_k$ is reduced by γ_e if we cannot extrapolate along the other scaled random directions and their opposite directions. Therefore, for each $t \in S$, according to the role of updating α_r in lines 19, 36, and 39 of Algorithm 1,

$$\alpha_{r'} \geq \gamma_e \delta_k / \gamma_e^{R_m - 1} = \gamma_e^{2-R_m} \delta_k$$

in the $(k+1)$th iteration of `VRDFON-basic`. Put $\overline{\gamma} := \gamma_e^{2(2-R_m)} \gamma > 0$. We now find an upper bound on the number of successful iterations and the corresponding function evaluations of `DS-basic`. For all $t \in S$ in the $(k+1)$th iteration of `VRDFON-basic`, we have

$$\tilde{f}(y^{t+1}) - \tilde{f}(y^t) = \tilde{f}(z^{\hat{r}}) - \tilde{f}(z_{\text{best}}) \leq -\gamma \alpha_{r'}^2 \leq -\overline{\gamma} \delta_k^2,$$

recursively resulting in $\tilde{f}(y^{t+1}) \leq \tilde{f}(x^0) - \overline{\gamma}\delta_k^2 \sum_{t \in S} 1 = \tilde{f}(x^0) - \overline{\gamma}\delta_k^2 |S|$. From (2) we conclude that

$$|S| \leq \overline{\gamma}^{-1} \delta_k^{-2} \Big(\tilde{f}(x^0) - \tilde{f}(y^{t+1}) \Big) \leq \overline{\gamma}^{-1} \delta_k^{-2} \Big(f(x^0) - \widehat{f} + 2\omega \Big).$$

Therefore, (19) is valid. The step size δ_k is fixed, independent of t, and updated outside `DS-basic`. As mentioned earlier, `MLS-basic` requires at most $2R_m + 1$ function evaluations in the worst case (using R_m scaled random directions and R_m corresponding opposite directions, all iterations of `MLS-basic` are unsuccessful; however, if the r'th iteration is successful, then `good` is false when $p^{r'}$ is attempted and true when $-p^{r'}$ is attempted; a sufficient gain along the last opposite direction $-p^{r'}$ is found. Therefore, an extrapolation with at most two function evaluations is attempted. Therefore, the successful iterations of `DS-basic` use at most

$$(2R_m + 1)\overline{\gamma}^{-1} \delta_k^{-2} \Big(f(x^0) - \widehat{f} + 2\omega \Big)$$

function evaluations.

U denotes the index set of unsuccessful iterations of `DS-basic`. Since $T_0 = |U| + |S|$ and `MLS-basic` uses $2R_m$ function evaluations for each unsuccessful iteration, we conclude that the unsuccessful iterations of `DS-basic` use at most $2R_m|U| \leq 2R_m T_0$ function evaluations. Consequently, the number of function evaluations of `DS-basic` is bounded by (20).

(ii) In this case, the unsuccessful iterations of `DS-basic` generate the sequence y^t $(t = 1, \cdots, T_0)$, resulting in, that for at least one $y^{t'}$ $(1 \le t' \le T_0)$ of the evaluated points, with probability $\ge 1 - 2^{-R_d} = 1 - 2^{-T_0 R_m} \ge 1 - \eta^{T_0} \ge 1 - \eta$, $\|g(y^{t'})\|_* \le \sqrt{cn}\Gamma(\delta_k)$ holds for a given $0 < \eta < \frac{1}{2}$. As mentioned in (i), the step size δ_k is fixed, independent of t; hence the bound $\sqrt{cn}\Gamma(\delta_k)$ is fixed for `DS-basic`. □

The objective function f is convex ($\sigma = 0$) if the condition

$$f(y) \ge f(x) + g(x)^T(y - x) + \frac{1}{2}\sigma\|y - x\| \quad \text{for } x, y \in \mathbb{R}^n \tag{22}$$

holds and is strongly convex ($\sigma > 0$) if (22) holds.

It is proved that an upper bound for the unknown gradient norm of at least one of points generated by the unsuccessful iterations of `VRDFON-basic` is found for all cases with a given probability arbitrarily close to one in the presence of noise.

Theorem 3 *Assume that (A1)–(A4) hold and* $\delta_{\max} > 0$, $Q > 1$, $0 < \gamma_{\rm rd} < 1$, $\gamma_e > 1$, $0 < \gamma < 1$,

$$\delta_{\min} := \Theta(\sqrt{\omega}) \tag{23}$$

and `nfmax` *is sufficiently large. Let* $\{x^k\}$ *(*$k = 1, 2, \cdots$*) be the sequence generated by* `VRDFON-basic`*. Then*

$$\delta_\ell = Q^{1-\ell}\delta_{\max} \quad \textit{for } \ell \ge 1 \tag{24}$$

and `VRDFON-basic` *terminates after at most*

$$K := 1 + \left\lfloor \frac{\log(\delta_{\max}/\delta_{\min})}{\log Q} \right\rfloor = \mathcal{O}(\log \omega^{-1/2}) \tag{25}$$

unsuccessful iterations. Then, for a given $0 < \eta < \frac{1}{2}$, `VRDFON-basic` *finds at least one point* $x^{\ell'}$ *with probability at least* $1 - 2^{-R_v} \ge 1 - \eta$ *satisfying*
(i) in the nonconvex case the condition

$$\|g(x^{\ell'})\|_* = \mathcal{O}(\sqrt{n\omega}); \tag{26}$$

(ii) in the convex case the condition (26) *and*

$$f(x^{\ell'}) - \widehat{f} = \mathcal{O}(r_0\sqrt{n\omega}), \tag{27}$$

where r_0 *is given by*

$$r_0 := \sup\left\{\|x - \widehat{x}\| \mid x \in \mathbb{R}^n, \quad f(x) \le f(x^0)\right\} < \infty; \tag{28}$$

(iii) in the strongly convex case the condition (26)*,*

$$f(x^{\ell'}) - \widehat{f} = \frac{\mathcal{O}(n\omega)}{2\sigma}, \quad \textit{and} \quad \|x^{\ell'} - \widehat{x}\| = \frac{\mathcal{O}(\sqrt{n\omega})}{\sigma}. \tag{29}$$

Here $\widehat{f}$ is finite by (A1) and (A2) discussed in Section 1.2 and R_v comes from (16)*.*

Proof (i) Since `VRDFON-basic` has K unsuccessful iterations, from calls to `DS-basic` and Theorem 2(ii), the condition

$$\|g(x^{\ell'})\|_* \le \sqrt{cn} \min_{\ell=0:K} \Gamma(\delta_\ell), \tag{30}$$

holds for at least one $x^{\ell'}$ of the evaluated points with probability

$$\ge 1 - 2^{-R_v} = 1 - 2^{-T_0 K R_m} \ge 1 - \eta^{T_0 K} \ge 1 - \eta > \frac{1}{2}$$

for a given $0 < \eta < \frac{1}{2}$. By (25), we have $\delta_K = Q^{1-K}\delta_{\max} \le \delta_{\min}$. Then (23)–(25) yield

$$\begin{aligned}\Gamma(\delta_K) &= \overline{L}\delta_K + \gamma_e^{R_m-1}\frac{4\omega}{\gamma_{\rm rd}\delta_K}\\ &= \overline{L}Q^{1-K}\delta_{\max} + \gamma_e^{R_m-1}Q^{K-1}\frac{4\omega}{\gamma_{\rm rd}\delta_{\max}} = \mathcal{O}(\sqrt{\omega}),\end{aligned}$$

whose application in (30) leads to (26). Here $\overline{L}$ comes from Theorem 1.
(ii) The convexity of f leads to

$$\widehat{f} \ge f_\ell + g(x^\ell)^T(\widehat{x} - x^\ell) \quad \text{for all } \ell \ge 0.$$

(i) leads to the fact that for at least one $x^{\ell'}$ of the evaluated points with probability $\ge 1 - 2^{-R_v} > \frac{1}{2}$ the condition

$$f_{\ell'} - \widehat{f} \le g(x^{\ell'})^T(x^{\ell'} - \widehat{x}) \le \|g(x^{\ell'})\|_*\|x^{\ell'} - \widehat{x}\| = \mathcal{O}(r_0\sqrt{n\omega})$$

holds for a given $0 < \eta < \frac{1}{2}$.
(iii) If x is assumed to be fixed, the right-hand side of (22) is a convex quadratic function with respect to y whose gradient in the components vanishes at $y_i = x_i - s_i\sigma^{-1}g_i(x)$ for $i = 1, \cdots, n$, leading to $f(y) \ge f(x) - \dfrac{1}{2\sigma}\|g(x)\|_*^2$. As mentioned earlier, $s \in \mathbb{R}^n$ is a scaling vector here. By applying (26) in this inequality, we obtain at least for one $x^{\ell'}$ of the evaluated points with probability $\ge 1 - 2^{-R_v} > \frac{1}{2}$

$$f_{\ell'} - \widehat{f} \le \frac{1}{2\sigma}\|g(x^{\ell'})\|_*^2 = \frac{\mathcal{O}(n\omega)}{2\sigma} \quad \text{for } \ell' \ge 0 \text{ and a given } 0 < \eta < \tfrac{1}{2}.$$

Substituting x for $\widehat{x}$ and y for $x^{\ell'}$ into (22), we get $f_{\ell'} \geq f(\widehat{x}) + \frac{\sigma}{2}\|x^{\ell'} - \widehat{x}\|^2$ such that (i) leads to the fact that for at least one $x^{\ell'}$ of the evaluated points with probability $\geq 1 - 2^{-R_v} > \frac{1}{2}$

$$\|x^{\ell'} - \widehat{x}\|^2 \leq \frac{2}{\sigma}(f_{\ell'} - \widehat{f}) \leq \frac{1}{\sigma^2}\|g(x^{\ell'})\|_*^2 = \frac{\mathcal{O}(n\omega)}{\sigma^2}$$

holds for a given $0 < \eta < \frac{1}{2}$. □

Compared to the results discussed in Section 1.2, the order of ω in the bound (26) is the same as that in (5) and (6). The conditions (27) and (29) are the same as those of Berahas et al. [8], except that they are satisfied with high probability.

The following result discusses the complexity bound for `VRDFON-basic` for all cases. It is proved that an upper bound on the number of function evaluations used by `VRDFON-basic` is found with a given probability arbitrarily close to one in the presence of noise.

Theorem 4 *Let $\{x^k\}$ $(k = 1, 2, \cdots)$ be the sequence generated by* `VRDFON-basic`*. Under the assumptions of Theorem 3,* `VRDFON-basic` *terminates after at most*
(i) $\mathcal{O}(R_m T_0 \omega^{-1})$ function evaluations in the nonconvex case,
(ii) $\mathcal{O}(\sqrt{n} R_m T_0 \omega^{-1/2})$ function evaluations in the convex case,
(iii) $\mathcal{O}(n R_m T_0 \log \omega^{-1})$ function evaluations in the strongly convex case.

Proof Denote by N_K the number of function evaluations for the termination of `VRDFON-basic`, put $N_0 := 1$, and denote $f_\ell = f(x^\ell)$. Here K comes from (25). In worst case, we terminate `VRDFON-basic` after at most K unsuccessful iterations from calls to `DS-basic`, with K points satisfying (21), and at least one point satisfying (26). Since the gradient and Lipschitz constants are unknown, these points are unknown. As a consequence of this termination, we have $\delta_\ell = Q^{1-\ell}\delta_{\max} \leq \delta_{\min}$ for $\ell \geq K$ and

$$\delta_\ell \geq \delta_{\min} \quad \text{for } \ell \in \mathcal{B} := \{1, \cdots, K\}. \tag{31}$$

The condition (31) is used in the proof of (ii) and (iii).
(i) We conclude from (20) and (24)–(25) that

$$\begin{aligned} N_K &\leq 1 + \sum_{\ell=1}^{K} \Big(2T_0 + (2R_m + 1)T_0\overline{\gamma}^{-1}\delta_\ell^{-2}(f(x^0) - \widehat{f} + 2\omega)\Big) \\ &= 1 + 2R_m T_0 K + (2R_m + 1)T_0\overline{\gamma}^{-1}\Big(f(x^0) - \widehat{f} + 2\omega\Big)\sum_{\ell=1}^{K}\delta_\ell^{-2} \end{aligned}$$

$$= 1 + 2R_mT_0K + (2R_m+1)T_0\overline{\gamma}^{-1}\delta_{\max}^{-2}\Big(f(x^0) - \widehat{f} + 2\omega\Big)\sum_{\ell=1}^{K} Q^{2\ell-2}$$

$$= 1 + 2R_mT_0K + (2R_m+1)T_0\overline{\gamma}^{-1}\delta_{\max}^{-2}\Big(f(x^0) - \widehat{f} + 2\omega\Big)\frac{Q^{2K}-1}{Q^2-1}.$$

Here $\overline{\gamma}$ comes from Theorem 2 and for a given $0 < \eta < \frac{1}{2}$ $R_m = \lceil \log_2 \eta^{-1} \rceil \geq 2$ comes from Algorithm 1. In this case, R_mQ^{2K} dominates the other terms (R_mK, $R_m\omega Q^{2K}$, Q^{2K}, ωQ^{2K}), resulting in

$$N_K = \mathcal{O}(R_mT_0\omega^{-1}).$$

(ii) From (A1) and (A2), r_0 is finite. The convexity of f results in

$$\widehat{f} \geq f_\ell + g(x^\ell)^T(\widehat{x} - x^\ell) \quad \text{for all } \ell \geq 0.$$

By Theorem 3, for a given $0 < \eta < \frac{1}{2}$, we get with probability $\geq 1 - 2^{-R_v} \geq 1 - \eta > \frac{1}{2}$

$$f_\ell - f_{\ell+1} \leq f_\ell - \widehat{f} \leq g(x^\ell)^T(x^\ell - \widehat{x}) \leq \|g(x^\ell)\|_* \|x^\ell - \widehat{x}\| \tag{32}$$

$$\leq r_0\sqrt{cn}\Big(\overline{L}\delta_\ell + \gamma_e^{R_m-1}\frac{4\omega}{\gamma_{\rm rd}\delta_\ell}\Big) \quad \text{for } \ell \in \mathcal{B}. \tag{33}$$

Here $\overline{L}$ comes from Theorem 1. We consider the following two cases:
Case 1. The first term $\overline{L}\delta_\ell$ in (33) dominates the second term. Then we have

$$f_\ell - f_{\ell+1} = \mathcal{O}(\sqrt{n}\delta_\ell) \quad \text{for } \ell \in \mathcal{B}\ . \tag{34}$$

Then we define $\mathcal{B}_1 := \{\ell \in \mathcal{B} \mid (34) \text{ holds}\}$.
Case 2. The second term $4\gamma_e^{R_m-1}\omega/(\gamma_{\rm rd}\delta_\ell)$ in (33) dominates the first term. Then we conclude from (31) that

$$f_\ell - f_{\ell+1} = \mathcal{O}(\sqrt{n}(\omega/\delta_\ell)) = \mathcal{O}(\sqrt{n}(\omega/\delta_{\min})) = \mathcal{O}(\sqrt{n\omega}) \quad \text{for } \ell \in \mathcal{B}. \tag{35}$$

Then we define $\mathcal{B}_2 = \{\ell \in \mathcal{B} \mid (35) \text{ holds}\}$.
Then we conclude from (24), (23), and (25) that with probability $\geq 1-2^{-R_v} > \frac{1}{2}$

$$\sum_{\ell\in\mathcal{B}}\frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\delta_\ell^2} = \sum_{\ell\in\mathcal{B}_1}\frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\delta_\ell^2} + \sum_{\ell\in\mathcal{B}_2}\frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\gamma\delta_\ell^2}$$

$$\leq \sum_{\ell\in\mathcal{B}_1}\frac{f_\ell - f_{\ell+1} + 2\omega}{\delta_\ell^2} + \sum_{\ell\in\mathcal{B}_2}\frac{f_\ell - f_{\ell+1} + 2\omega}{\delta_\ell^2}$$

$$\leq \sum_{\ell\in\mathcal{B}_1}\frac{\mathcal{O}(\sqrt{n}\delta_\ell) + 2\omega}{\delta_\ell^2} + \sum_{\ell\in\mathcal{B}_2}\frac{\mathcal{O}(\sqrt{n\omega}) + 2\omega}{\delta_\ell^2}$$

$$
\begin{aligned}
&\le \sum_{\ell\in\mathcal{B}} \frac{\mathcal{O}(\sqrt{n}\delta_\ell)+2\omega}{\delta_\ell^2} + \sum_{\ell\in\mathcal{B}} \frac{\mathcal{O}(\sqrt{n\omega})+2\omega}{\delta_\ell^2}\\
&= \mathcal{O}(\sqrt{n})\sum_{\ell\in\mathcal{B}}\delta_\ell^{-1} + \mathcal{O}(\sqrt{n\omega})\sum_{\ell\in\mathcal{B}}\delta_\ell^{-2} + 4\omega\sum_{\ell\in\mathcal{B}}\delta_\ell^{-2}\\
&= \mathcal{O}(\sqrt{n})\sum_{\ell\in\mathcal{B}}Q^{\ell-1} + \mathcal{O}(\sqrt{n\omega})\sum_{\ell\in\mathcal{B}}Q^{2\ell-2} + 4\omega\sum_{\ell\in\mathcal{B}}Q^{2\ell-2}\\
&= \mathcal{O}(\sqrt{n}Q^K) + \mathcal{O}(\sqrt{n\omega}Q^{2K}) + \omega\mathcal{O}(Q^{2K})\\
&= \mathcal{O}(\sqrt{n}\omega^{-1/2}) + \mathcal{O}(\sqrt{n\omega}\omega^{-1}) + \omega\mathcal{O}(\omega^{-1})\\
&= \mathcal{O}(\sqrt{n}\omega^{-1/2}) + \mathcal{O}(\sqrt{n}\omega^{-1/2}) + \mathcal{O}(n) = \mathcal{O}(\sqrt{n}\omega^{-1/2}),
\end{aligned}
$$

holds for a given $0<\eta<\frac{1}{2}$, so that by (i) and $R_m = \lceil \log_2 \eta^{-1}\rceil \ge 2$

$$
N_K \le 1 + (2R_m+1)T_0 \sum_{\ell\in\mathcal{B}} \frac{\tilde f_\ell - \tilde f_{\ell+1}}{\gamma\delta_\ell^2} = \mathcal{O}\left(\sqrt{n}R_m T_0 \omega^{-1/2}\right)
$$

holds with probability $\ge 1-2^{-R_v} \ge 1-\eta > \frac{1}{2}$ for a given $0<\eta<\frac{1}{2}$.
(iii) When x is assumed to be fixed, the right hand side of (22) is a convex quadratic function in terms of y whose gradient in the components vanishes at $y_i = x_i - s_i\sigma^{-1}g_i(x)$ for $i=1,\cdots,n$, resulting in $f(y) \ge f(x) - \frac{1}{2\sigma}\|g(x)\|_*^2$. Here as mentioned earlier $s\in\mathbb{R}^n$ is a scaling vector. By applying (26) in this inequality, for a given $0<\eta<\frac{1}{2}$, we get with probability $\ge 1-2^{-R_v} \ge 1-\eta > \frac{1}{2}$

$$
f_\ell - f_{\ell+1} \le f_\ell - \widehat{f} \le \frac{1}{2\sigma}\|g(x^\ell)\|_*^2 \le \frac{cn}{2\sigma}\left(\overline{L}\delta_\ell + \gamma_e^{R_m-1}\frac{4\omega}{\gamma_{\rm rd}\delta_\ell}\right)^2 \quad \text{for } \ell\in\mathcal{B}. \tag{36}
$$

Here $\overline{L}$ comes from Theorem 1. We consider the following two cases:
Case 1. The first term $\overline{L}\delta_\ell$ in (36) dominates the second term. Then we have

$$
f_\ell - f_{\ell+1} = \mathcal{O}(n\delta_\ell^2) \quad \text{for } \ell\in\mathcal{B} \tag{37}
$$

and denote $\mathcal{B}_1 := \{\ell\in\mathcal{B} \mid (37) \text{ holds}\}$.
Case 2. The second term $4\gamma_e^{R_m-1}\omega/(\gamma_{\rm rd}\delta_\ell)$ in (36) dominates the first term. Then we conclude from (31) that

$$
f_\ell - f_{\ell+1} = \mathcal{O}\Big(n(\omega/\delta_\ell)^2\Big) = \mathcal{O}\Big(n(\omega/\delta_{\min})^2\Big) = \mathcal{O}(n\omega) \quad \text{for } \ell\in\mathcal{B} \tag{38}
$$

and denote $\mathcal{B}_2 = \{\ell\in\mathcal{B} \mid (38) \text{ holds}\}$.
Then for a given $0<\eta<\frac{1}{2}$ we conclude from (23)–(25) that with probability

$\geq 1 - 2^{-R_v} \geq 1 - \eta > \frac{1}{2}$

$$\begin{aligned}
\sum_{\ell \in \mathcal{B}} \frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\delta_\ell^2} &= \sum_{\ell \in \mathcal{B}_1} \frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\delta_\ell^2} + \sum_{\ell \in \mathcal{B}_2} \frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\delta_\ell^2} \\
&\leq \sum_{\ell \in \mathcal{B}_1} \frac{f_\ell - f_{\ell+1} + 2\omega}{\delta_\ell^2} + \sum_{\ell \in \mathcal{B}_2} \frac{f_\ell - f_{\ell+1} + 2\omega}{\delta_\ell^2} \\
&\leq \sum_{\ell \in \mathcal{B}_1} \frac{\mathcal{O}(n\delta_\ell^2) + 2\omega}{\delta_\ell^2} + \sum_{\ell \in \mathcal{B}_2} \frac{\mathcal{O}(n\omega) + 2\omega}{\delta_\ell^2} \\
&\leq \sum_{\ell \in \mathcal{B}} \frac{\mathcal{O}(n\delta_\ell^2) + 2\omega}{\delta_\ell^2} + \sum_{\ell \in \mathcal{B}} \frac{\mathcal{O}(n\omega) + 2\omega}{\delta_\ell^2} \\
&= \mathcal{O}(n)K + \mathcal{O}(n\omega) \sum_{\ell \in \mathcal{B}} \delta_\ell^{-2} + 4\omega \sum_{\ell \in \mathcal{B}} \delta_\ell^{-2} \\
&= \mathcal{O}(n)K + \mathcal{O}(n\omega) \sum_{\ell \in \mathcal{B}} Q^{2\ell-2} + 4\omega \sum_{\ell \in \mathcal{B}} Q^{2\ell-2} \\
&= \mathcal{O}(n)K + \mathcal{O}(n\omega)\mathcal{O}(Q^{2K}) + \omega\mathcal{O}(Q^{2K}) \\
&= \mathcal{O}(n \log \omega^{-1}) + \mathcal{O}(n\omega\omega^{-1}) + \omega\mathcal{O}(\omega^{-1}) = \mathcal{O}(n \log \omega^{-1})
\end{aligned}$$

so that by (i) and since $R_m = \lceil \log_2 \eta^{-1} \rceil \geq 2$,

$$N_K \leq 1 + (2R_m + 1)T_0 \sum_{\ell \in \mathcal{B}} \frac{\tilde{f}_\ell - \tilde{f}_{\ell+1}}{\gamma\delta_\ell^2} = \mathcal{O}(nR_mT_0 \log(\omega^{-1}))$$

holds with probability $\geq 1 - 2^{-R_v} \geq 1 - \eta > \frac{1}{2}$ for a given $0 < \eta < \frac{1}{2}$. □

Compared to the results discussed in Section 1.2, the order ω of our complexity bounds is the same as that of Berahas et al. [8] which is valid in expectation.

If $R_mT_0 \sim n$, then the factors of our bounds in the nonconvex, convex, and strongly convex regions will be large by a factor n, as in the deterministic case. We have shown numerically in Section 3 of `suppMat.pdf` that the efficiency and robustness of an improved version of `VRDFON-basic` with $R_mT_0 \sim 1$ are reduced, although our factors are better by a factor n than in the deterministic case (see Bandeira et al. [6] and Kimiaei & Neumaier [33]). When $R_mT_0 \sim n$ is satisfied, the efficiency and robustness of an improved version of `VRDFON-basic` are actually increased.

In the next section, we discuss that the complexity results are not satisfied if scaled random directions are not used and are otherwise valid. In fact, using scaled random directions guarantees our complexity results, and other types of directions (Section 1 of `suppMat.pdf`) can be used to increase the efficiency and robustness of an improved version of `VRDFON-basic`.

6 `VRDFON`, an improved version of `VRDFON-basic`

This section discusses some implementation details, tuning parameters of `VR-BBON-baisc`, and under which conditions the complexity results for `VRDFON` can be guaranteed. As mentioned earlier, `MLS` is an implemented version of `MLS-basic` discussed in Section 1.7 of `suppMat.pdf` and `DS` is an implemented version of `DS-basic` discussed in Section 1.8 of `suppMat.pdf`.

The values of tuning parameters of `VRDFON-baisc` are $Q = 1.5$, $\gamma_{\rm rd} = 10^{-30}$, $\delta_{\min} = 0$, $\delta_{\max} = 1$, $\gamma = 10^{-6}$, $\gamma_e = 3$, $R_m = n$, and $T_0 = 5$. The values of tuning parameters of `VRDFON` are chosen in Table 1 of `suppMat.pdf` after `VRDFON` is ennriched by many practical enhancements discussed in Section 1 of `suppMat.pdf`.

Since the three tuning parameters,
- number $R_m \geq 2$ of scaled random directions in `MLS`,
- number $C \geq 2$ of random approximate coordinate directions in `MLS`,
- number $T_0 \geq 1$ of calls to `MLS` by `DS`,

affect the factors of our complexity bounds, and since the model-based case (`model` $= 1$) is preferable to the model-free case (`model` $= 0$) in the presence of noise, we perform a testing and tuning for these tuning parameters in Section 3 of `suppMat.pdf`. Based on our findings, other tuning parameters were fixed as they did not change the efficiency and robustness of our solver.

The variable `comBound` takes values of $0, 1, 2$ and is a tuning parameter for considering three cases for the complexity results of `VRDFON`. Accordingly, we discuss the conditions under which complexity results can be found for `VRDFON`:
CASE 1 (`comBound` $= 0$). In this case, random approximate coordinate directions are used and random scaled directions are ignored. Other proposed directions and heuristic techniques are also used. In this case, Proposition 1 may not be valid and no complexity result is found for `VRDFON`.
CASE 2 (`comBound` $= 1$). Both random scaled directions and random approximate coordinate directions are used. Other proposed directions and heuristic techniques are also used. In this case, the order of complexity bounds does not change for all cases after heuristic improvements are used, but only their factors become larger
CASE 3 (`comBound` $= 2$). In this case, random scaled directions are used and random approximate coordinate directions are ignored. Other proposed directions and heuristic techniques are also used. The order of complexity bounds does not change for all cases after applying the heuristic improvements, only their factors become larger.

In CASE 2 and CASE 3, Theorem 1, Theorem 2, and Theorem 4 remain valid with the following modifications:
- In Theorem 1(i), number R_m of scaled random steps in each `MLS` must be replaced by number T of all trial steps, and Theorem 1(ii) remains valid with

probability $\geq 1 - 2^{-R_v} \geq 1 - \eta > \frac{1}{2}$ for a given $0 < \eta < \frac{1}{2}$.
• In Theorem 2(i), the number of function evaluations of `DS` is bounded by

$$(2R_m + 2C + 4)T_0 + (2R_m + 2C + 5)T_0\overline{\gamma}^{-1}\delta_k^{-2}\Big(f(x^0) - \widehat{f} + 2\omega\Big)$$

and

$$(2R_m + 4)T_0 + (2R_m + 5)T_0\overline{\gamma}^{-1}\delta_k^{-2}\Big(f(x^0) - \widehat{f} + 2\omega\Big)$$

in Case 2 and Case 3, respectively. As mentioned earlier, the step size δ_k is fixed, independent of t, and updated outside `DS`. We have described at the end of Section 1.8 of `suppMat.pdf` how to obtain the factors of the above two bounds. Theorem 3(ii) remains valid.
• In the proof of Theorem 4, $2R_mT_0$ and $(2R_m + 1)T_0$ must be replaced by $(2R_m + 2C + 4)T_0$ and $(2R_m + 2C + 5)T_0$, respectively, in Case 2 and by $(2R_m + 4)T_0$ and $(2R_m + 5)T_0$, respectively, in Case 3.

7 Numerical results

In this section, we describe how the test problems are selected. Performance measures are then defined to determine which solvers are robust and efficient for small to large scale problems. We then compare our solver with the state-of-the-art solvers for low to high dimensional problems (*results are averaged over five runs*). Finally, we make a recommendation as to which solvers are robust and efficient based on dimension and noise level.

Details of the solvers compared can be found in Section 2 of `suppMat.pdf`. However, a list of solvers compared with `VRDFON` are `VRBBO` by Kimiaei & Neumaier [33], `SNOBFIT` by Huyer & Neumaier [29], `GRID` by Elster & Neumaier [20], `NOMAD` by Abramson et al. [1], `UOBYQA` and `NEWUOA` by Powell [41,42], `BFO` by Porcelli & Toint [40], `DSPFD` by Gratton et al. [22], `MCS` by Huyer & Neumaier [28], `BCDFO` by Gratton et al. [24], `SDBOX` by Lucidi & Sciandrone [36], `CMAES` by Auger & Hansen [4], `LMMAES` by Loshchilov et al. [35], `fMAES` by Beyer [10], and `BiPopMAES` by Beyer & Sendhoff [11]. Moreover, `subUOBYQA`, `subNEWUOA`, and `subNMSMAX` are respectively `UOBYQA`, `NEWUOA`, and `NMSMAX` in a random subspace to handle problems in medium and high dimensions. We denote by `NOMAD1` the model-based version of `NOMAD` and by `NOMAD2` the model-free version of it.

7.1 Test problems

For our numerical results, we used 549 unconstrained `CUTEst` test problems from the collection of Gould et al. [21]. To prepare these results, the test environment of Kimiaei & Neumaier [32] was used.

The starting point. As in [33], we choose the starting point $x^0 := 0$ and shift the arguments by

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

to avoid a solver guessing the solution of toy problems with a simple solution (e.g., all zeros or all ones) – there are quite a few of these in the `CUTEst` library. That is, the initial point is chosen by $x^0 := \xi$ and the initial inexact function value is $\tilde f_0 := \tilde f(x^0)$ while the other inexact function values are computed by $\tilde f_\ell := \tilde f(x^\ell + \xi)$ for all $\ell \geq 0$. In fact, this choice increases the difficulty of the problems, see Figures 3–4 in Section 7.3.1 (the efficiency and robustness of our solver are higher when the standard initial points are used than when the shifted initial points are used).

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.

7.2 Performance measures

Two important tools for figuring out which solver is *robust* and *efficient* are the data profile of Moré & Wild [38] and the performance profile of Dolan & Moré [19], 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.,

$$\delta_s(\kappa) := \frac{1}{|\mathcal{P}|}\Big|\Big\{p \in \mathcal{P} \;\Big|\; cr_{p,s} := \frac{c_{p,s}}{n_p+1} \leq \kappa\Big\}\Big|. \tag{39}$$

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.,

$$\rho_s(\tau) := \frac{1}{|\mathcal{P}|}\Big|\Big\{p \in \mathcal{P} \;\Big|\; pr_{p,s} := \frac{c_{p,s}}{\min(c_{p,\overline{s}} \mid \overline{s} \in S)} \leq \tau\Big\}\Big|. \tag{40}$$

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. The other two plots are* *based on the noise levels. These four plots are used to identify the behaviour of the compared solvers with respect to problem scales and noise levels.*

Efficiency. The *efficiency* $e_{p,s}$ of the solver s to solve the problem p is the inverse of the performance ratio $pr_{p,s}$. Efficiency measures the ability of a solver $s \in \mathcal{S}$ relative to an ideal solver. The number 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.

Other plots based on the noise level. To see the behaviour of the compared solvers in the presence of low to high noise, we plot the number of problems solved and the efficiency versus the noise level.

Measure for the convergence speed. The quotients

$$q_s := (f_s - f_{\text{opt}})/(f_0 - f_{\text{opt}}) \quad \text{for } s \in \mathcal{S} \tag{41}$$

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 [33].

Maximum budgets and stopping tests.

We consider a problem *solved* by the solver s if $q_s \leq \varepsilon$ and neither the maximum number `nfmax` of function evaluations nor the maximum allowed time `secmax` in seconds was reached, and *unsolved* otherwise. ε, `secmax` and `nfmax` are chosen so that the best solver can solve at least half of the problems. They depend on the dimension and the noise level because increasing the noise level and dimension extremely increases the difficulty of the problems. Therefore, ε is chosen slightly larger for problems in medium and high dimensions than for problems in low dimensions. The following choices were found valuable:

$$\texttt{secmax} = \begin{cases} 180 & \text{if } 1 \leq n \leq 300, \\ 420 & \text{if } 301 \leq n \leq 5000, \end{cases}$$

$$\texttt{nfmax} = \begin{cases} 2n^2 + 1000n + 5000 & \text{if } 1 \leq n \leq 300, \\ 500n & \text{if } 301 \leq n \leq 5000 \end{cases}$$

and

$$\varepsilon := \begin{cases} 10^{-3} & \text{if } \omega \in \{10^{-4}, 10^{-3}\} \text{ and } n \in [1, 30], \\ 10^{-2} & \text{if } \omega \in \{0.1, 0.9\} \text{ and } n \in [1, 30], \\ 10^{-3} & \text{if } \omega = 10^{-4} \text{ and } n \in [31, 300], \\ 0.05 & \text{if } \omega \in \{0.1, 0.01, 0.001\} \text{ and } n \in [31, 300], \\ 0.05 & \text{if } \omega \in \{10^{-5}, 10^{-4}, 10^{-3}\} \text{ and } n \in [301, 5000]. \end{cases}$$

7.3 Comparison with the other solvers

This section compares the default version of `VRDFON` with the other solvers for problems in low to high dimensions. In each figure, we display only the best five solvers, but the best four solvers in high dimensions.

7.3.1 Small scale: $1 < n \leq 30$

For the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-1}, 0.9\}$ and small scale problems ($1 < n \leq 30$), this section contains a comparison between `VRDFON` and the well-known model-based solvers (`NOMAD1`, `UOBYQA`, `NEWUOA`, `BCDFO`, `GRID`, and `SNOBFIT`), the well-known direct search solvers (`NOMAD2`, `NMSMAX`, `BFO`, `MCS`, `DSPFD`), the well-known line search solvers (`VRBBO`, `SDBOX`, `FMINUNC`), and the well-known matrix adaptation evolution solvers (`CMAES`, `fMAES`, `BiPopMAES`, `LMMAES`). Moreover, a comparison is made between `VRDFON` and the five most efficient and robust solvers with both standard and shifted initial points.

To compare our algorithm with the well-known model-based and direct search solvers, Figure 1 shows the cumulative (over all noise levels used) performance and data profiles in its subfigures in terms of the number of function evaluations and the other two plots show (the `nf` efficiency versus the noise level ω and the number of solved problems versus the noise level ω). From the subfigures of Figure 1,

• `NEWUOA` is more efficient than the well-known model-based solvers and `VRDFON`, while `UOBYQA` is more robust than others and `VRDFON`, the third robust solver, is more efficient than `SNOBFIT` and `GRID` and less efficient than others. Moreover, `NOMAD1` is the second robust solver compared to the well-known model-based solvers;

• `VRDFON` is the second efficient solver and the first robust solver compared to the well-known direct search solvers, while `NMSMAX` is the first efficient solver and `NOMAD2` is the second robust solver.

To compare our algorithm with the well-known line search and matrix adaptation evolution solvers, Figure 2 shows in its subfigures, the cumulative (over all noise levels used) performance and data profiles in terms of the number of function evaluations and the other two plots show in terms of the noise levels

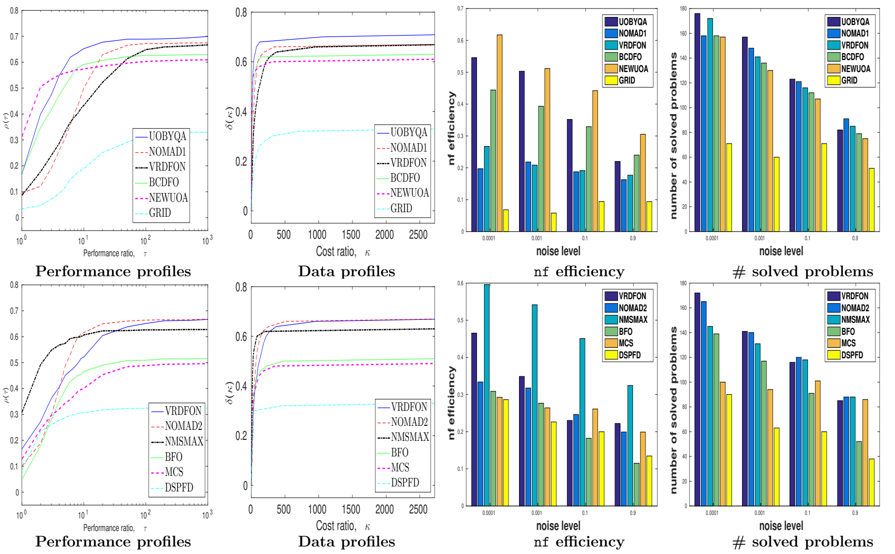


Fig. 1: Comparison between **VRDFON** and model-based solvers (first row) and direct search solvers (second row) for the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-1}, 0.9\}$ and small dimensions $1 < n \leq 30$. Data profile $\delta(\kappa)$ in dependence of a bound κ on the cost ratio, see (39) while performance profile $\rho(\tau)$ in dependence of a bound τ on the performance ratio, see (40). Problems solved by no solver are ignored. Here '# solved problems' counts the number of solved problems.

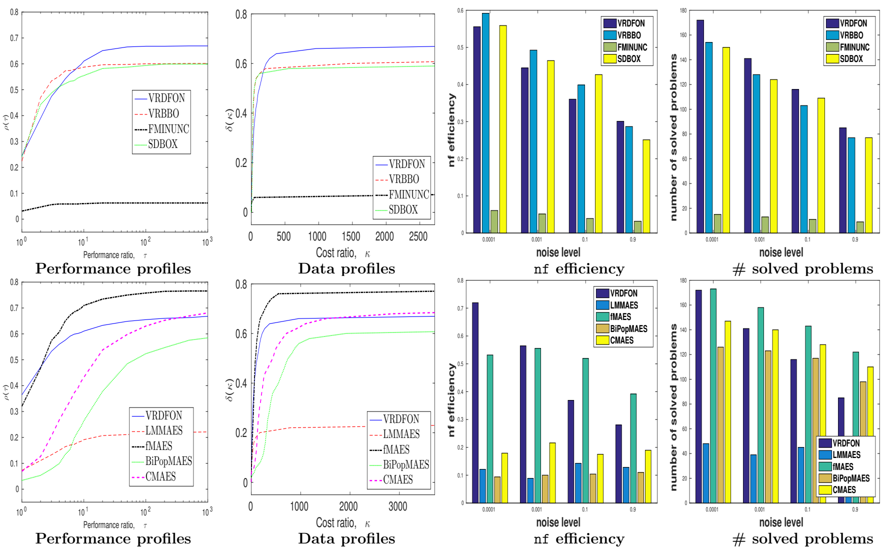
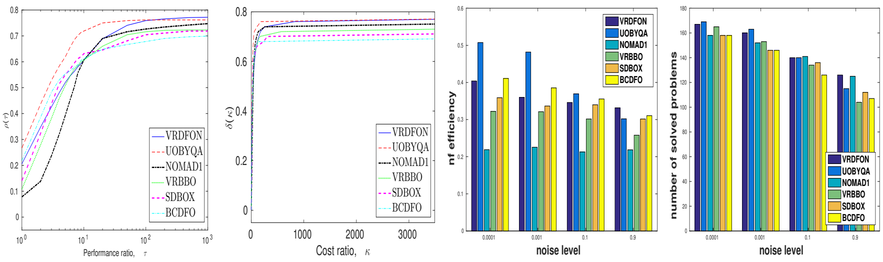


Fig. 2: Comparison between **VRDFON** and line search solvers (first row) and matrix adaptation evolution solvers solvers (second row). Details as in Figure 1.

(the nf efficiency versus the noise level ω and the number of solved problems versus the noise level ω). From the subfigures of Figure 2, we conclude that VRDFON is more robust and efficient than the known line search solvers, while VRDFON is more robust and efficient than the well-known matrix adaptation evolution solvers at low noise and fMAES is more robust and efficient than others at high noise. At low to high noise, VRDFON is slightly more efficient than the well-known matrix adaptation evolution solvers, while fMAES is more robust than others.

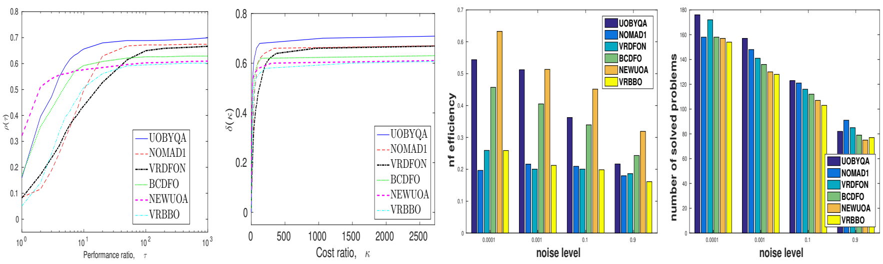
Since our solver is model-based and line search-based, we now provides two comparisons between VRDFON and the best model-based (NOMAD1, UOBYQA, NEWUOA, and BCDFO) and line search solvers (SDBOX and VRBBO) with the shifted and initial starting points for small scale problems.

If the default starting points are used, we conclude from Figure 3 that VRDFON is comparable to NEWUOA in terms of efficiency and is more robust than others. It is also more robust than others. In fact, the efficiency and robustness of VRDFON are higher when the standard initial points are used than when the shifted initial points are used, see Figure 4. This is reason why we used the shifted points for all test problems in our comparisons.



Performance profiles Data profiles nf efficiency # solved problems

Fig. 3: For the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-1}, 0.9\}$ and small dimensions $1 < n \leq 30$. The standard initial points are used. Details as in Figure 1.



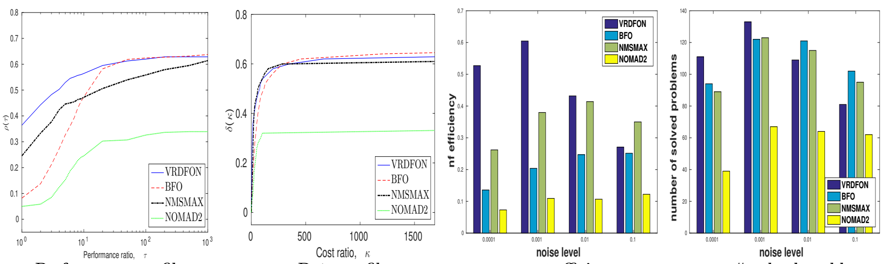
Performance profiles Data profiles nf efficiency # solved problems

Fig. 4: For the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-1}, 0.9\}$ and small dimensions $1 < n \leq 30$. The shifted initial points are used. Details as in Figure 1.

7.3.2 Medium scale: $30 < n \leq 300$

For the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and medium scale problems ($30 < n \leq 300$), this section contains a comparison between `VRDFON` and the well-known direct search solvers, line search solvers, and matrix adaptation evolution solvers.

Figures 5 and 6 show in their subfigures, the cumulative (over all noise levels used) performance and data profiles in terms of the number of function evaluations and the other two plots show (the `nf` efficiency versus the noise level ω and the number of solved problems versus the noise level ω). From the subfigures of Figure 5, we conclude that `VRDFON` is more robust and efficient than the well-known direct search solvers. From the subfigures of Figure 6, we can also conclude that `VRDFON` is more robust and efficient than the well-known line search solvers and matrix adaptation evolution solvers at low and medium noise, while `LMMAES` is more robust than `VRDFON` and the other matrix adaptation evolution solvers at high noise.



Performance profiles **Data profiles** **`nf` efficiency** **# solved problems**

Fig. 5: Comparison between `VRDFON` and direct search solvers for the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and medium dimensions $30 < n \leq 300$. Other details as in Figure 1.

Since the model-based solvers cannot handle problems in high dimensions, as described earlier, we denote `UOBYQA` in the random subspace by `subUOBYQA` and `NEWUOA` in the random subspace by `subNEWUOA`. Moreover, we denote `NMSMAX` in the random subspace by `subNMSMAX`.

The subfigures of Figure 7 show that `VRDFON` are much more efficient than others at low to high noise, but `VRDFON` are more robust than others at low noise, while `subNEWUOA`, `subUOBYQA`, and `NMSMAX` are slightly more robust than `VRDFON` only at high noise.

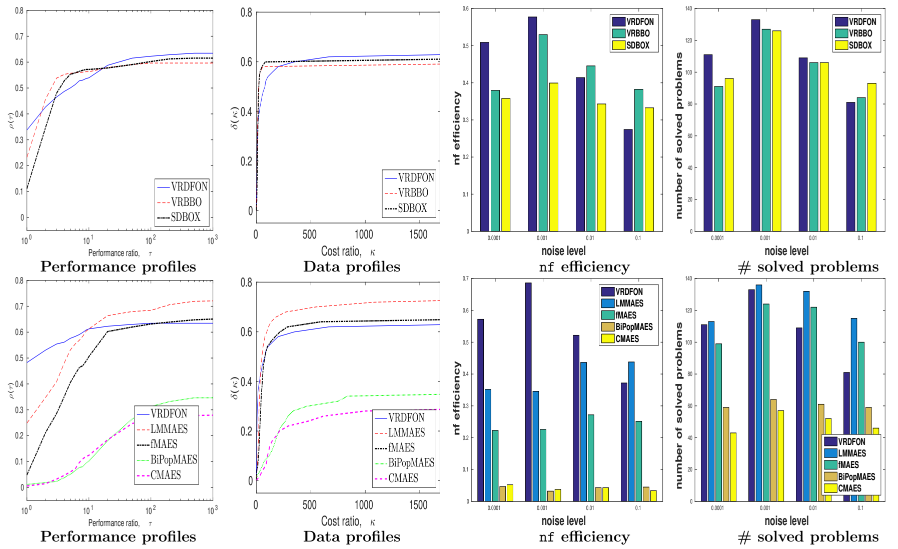


Fig. 6: Comparison between `VRDFON` and line search solvers (first row) and matrix adaptation evolution solvers solvers (second row) for the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and medium dimensions $30 < n \leq 300$. Other details as in Figure 1.

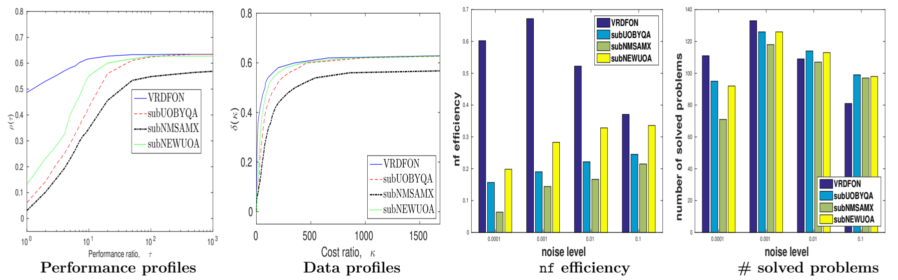
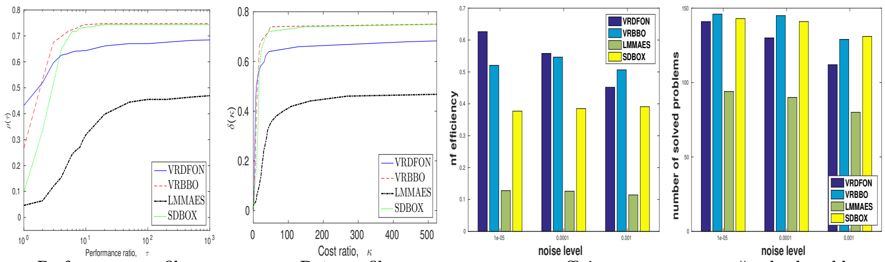


Fig. 7: Comparison between `VRDFON` and model-based solvers in a random subspace for the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and medium dimensions $30 < n \leq 300$. Other details as in Figure 1.

7.3.3 Large scale: $300 < n \leq 5000$

For the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$ and large dimensions $300 < n \leq 5000$, this section contains a comparison between `VRDFON` and the three effective solvers (`VRBBO`, `LMMAES`, and `SDBOX`) for problems in medium dimensions.

Figure 8 shows in its subfigures, the cumulative (over all noise levels used) performance and data profiles in terms of the number of function evaluations for the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$ and the other two plots show (the `nf` efficiency versus the noise level ω and the number of solved problems versus the noise level ω). We conclude from these subfigures that `VRDFON` is slightly more efficient than others, while `VRBBO` and `SDBOX` are slightly more robust than `VRDFON`.



Performance profiles **Data profiles** **nf efficiency** **# solved problems**

Fig. 8: Comparison between `VRDFON` and the effective solvers on the medium scale problems for the noise levels $\omega \in \{10^{-5}, 10^{-4}, 10^{-3}\}$ and large dimensions $300 < n \leq 5000$. Details as in Figure 1.

7.4 Recommendation

From Figures 8–10, we can easily choose which solvers are robust and which are efficient, depending on the noise level and dimension. For small and medium scale problems, Figures 9 and 10 are two comparisons between six more robust and efficient solvers, while Figure 8 is a comparison between four more robust and efficient solvers for large scale problems. In summary, we conclude that `VRDFON` is one of the four more robust and efficient solvers for problems in small dimensions in most cases, and one of the two more robust and efficient solvers for problems in medium and high dimensions in most cases. Therefore, this solver is highly recommended for unconstrained noisy DFO problems in low to high dimensions.

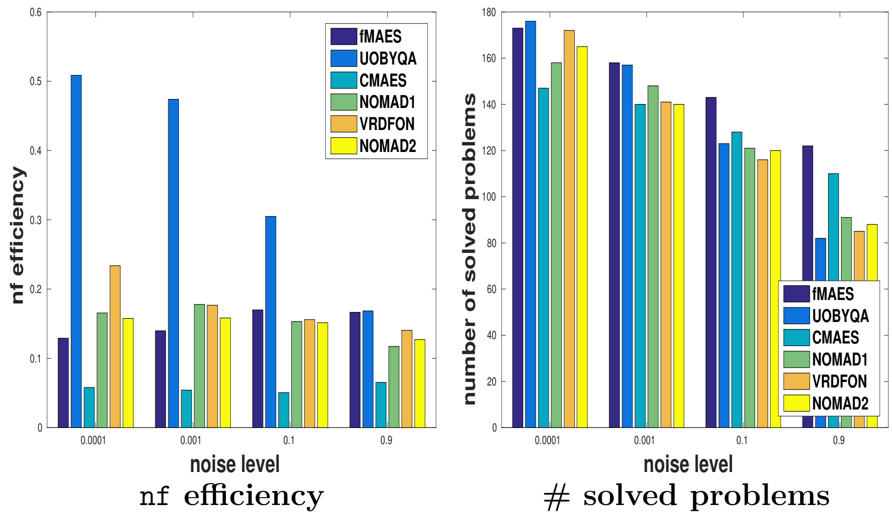


Fig. 9: Comparison between six more robust and efficient solvers for the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-1}, 0.9\}$ and small dimensions $1 < n \le 30$. Other details as in Figure 1.

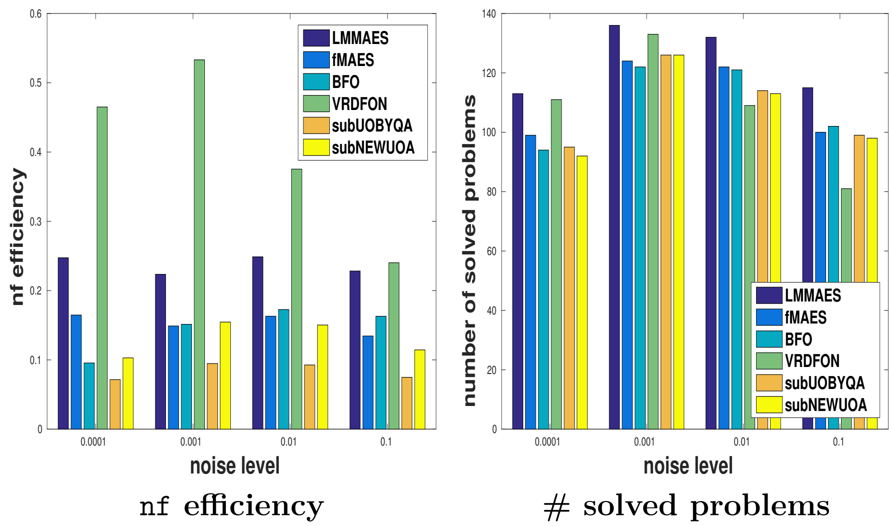


Fig. 10: Comparison between six more robust and efficient solvers for the noise levels $\omega \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and medium dimensions $30 < n \le 300$. Other details as in Figure 1.

References

1. M. A. Abramson, C. Audet, G. Couture, J. E. Dennis, Jr., S. Le Digabel and C. Tribes. The NOMAD project. Software available at https://www.gerad.ca/nomad/.
2. M. A. Abramson, C. Audet, J. E. Dennis, Jr., and S. Le Digabel. OrthoMADS: A Deterministic MADS Instance with Orthogonal Directions. *SIAM J. Optim* **20** (Jan 2009), 948–966.
3. C. Audet and J. E. Dennis. Mesh adaptive direct search algorithms for constrained optimization. *SIAM J. Optim.* **17** (January 2006), 188–217.
4. A. Auger and N. Hansen. A restart CMA evolution strategy with increasing population size. In *2005 IEEE Congress on Evolutionary Computation.* IEEE (2005).
5. C. Audet and W. Hare. Springer Series in Operations Research and Financial Engineering. *Derivative-Free and Blackbox Optimization.* 2017.
6. A. S. Bandeira, K. Scheinberg, and L. N. Vicente. Convergence of trust-region methods based on probabilistic models. *SIAM J. Optim* **24** (January 2014), 1238–1264.
7. A. S. Berahas, R. H. Byrd, and J. Nocedal. Derivative-free optimization of noisy functions via quasi-newton methods. *SIAM J. Optim.* **29** (January 2019), 965–993.
8. A. S. Berahas, L. Cao, and K. Scheinberg. Global convergence rate analysis of a generic line search algorithm with noise (2021). https://doi.org/10.48550/arXiv.1910.04055
9. E. H Bergou, E. Gorbunov, and P. Richtárik. Stochastic three points method for unconstrained smooth minimization, *SIAM J. Optim.* **30** (January 2020), 2726–2749.
10. H. G. Beyer. Design principles for matrix adaptation evolution strategies. Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion. https://doi.org/10.1145/3377929.3389870.
11. H. G. Beyer and B. Sendhoff. Simplify your covariance matrix adaptation evolution strategy. *IEEE Trans. Evol. Comput.* **21** (October 2017), 746–759.
12. M. D. Buhmann. Radial basis functions. *Acta Numer.* **9** (January 2000), 1–38.
13. R. Chen. *Stochastic Derivative-Free Optimization of Noisy Functions.* PhD thesis, Lehigh University (2015). Theses and Dissertations. 2548.
14. A. R. Conn and K. Scheinberg and L. N. Vicente. Introduction to Derivative-Free Optimization. *Society for Industrial and Applied Mathematics*, 2009.
15. A. R. Conn and Ph. L. Toint. An algorithm using quadratic interpolation for unconstrained derivative free optimization. In *Nonlinear Optimization and Applications*, pp. 27–47. Springer US (1996).
16. C. Davis. Theory of positive linear dependence. *Amer. J. Math.* **76** (October 1954), 733.
17. P. Deuflhard and G. Heindl. Affine invariant convergence theorems for newton's method and extensions to related methods. *SIAM J. Numer. Anal.* **16** (February 1979), 1–10.
18. M. A. Diniz-Ehrhardt, J.M. Martínez, and M. Raydan. A derivative-free nonmonotone line-search technique for unconstrained optimization. *J. Comput. Appl. Math.* **219** (October 2008), 383–397.
19. E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.* **91** (January 2002), 201–213.
20. C. Elster and A. Neumaier. A grid algorithm for bound constrained optimization of noisy functions. *IMA J. Numer. Anal.* **15** (1995), 585–608.
21. N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Comput. Optim. Appl.* **60** (2015), 545–557.
22. S. Gratton, C. W. Royer, L. N. Vicente, and Z. Zhang. Direct search based on probabilistic descent. *SIAM J. Optim* **25** (January 2015), 1515–1541.
23. S. Gratton, C. W. Royer, L. N. Vicente, and Z. Zhang. Complexity and global rates of trust-region methods based on probabilistic models. *IMA J. Numer. Anal.* **38** (August 2017), 1579–1597.

24. S. Gratton, Ph. L. Toint, and A. Tröltzsch. An active-set trust-region method for derivative-free nonlinear bound-constrained optimization. *Optim. Methods Softw.* **26** (October 2011), 873–894.
25. L. Grippo and F. Rinaldi. A class of derivative-free nonmonotone optimization algorithms employing coordinate rotations and gradient approximations. *Comput. Optim. Appl.* **60** (June 2014), 1–33.
26. L. Grippo and M. Sciandrone. Nonmonotone derivative-free methods for nonlinear equations. *Comput. Optim. Appl.* **37** (March 2007), 297–328.
27. N. J. Higham. Optimization by direct search in matrix computations. *SIAM J. Matrix Anal. Appl.* **14** (April 1993), 317–333.
28. W. Huyer and A. Neumaier. Global optimization by multilevel coordinate search. *J. Glob. Optim.* **14** (1999), 331–355.
29. W. Huyer and A. Neumaier. SNOBFIT – stable noisy optimization by branch and fit. *ACM. Trans. Math. Softw.* **35** (July 2008), 1–25.
30. W. Huyer and A. Neumaier. MINQ8: general definite and bound constrained indefinite quadratic programming. *Comput. Optim. Appl.* **69** (October 2017), 351–381.
31. M. Kimiaei. (2022). The VRDFON solver. Software available at `https://github.com/GS1400/VRDFON`.
32. M. Kimiaei and A. Neumaier. Testing and tuning optimization algorithm. Preprint, Vienna University, Fakultät für Mathematik, Universität Wien, Oskar-Morgenstern-Platz 1, A-1090 Wien, Austria (2019).
33. M. Kimiaei and A. Neumaier. Efficient unconstrained black box optimization. *Math. Program. Comput.* `http:/doi.org/10.1007/s12532-021-00215-9` (February 2022).
34. J. Larson, M. Menickelly, and S. M. Wild. Derivative-free optimization methods. *Acta Numer.* **28** (May 2019), 287–404.
35. I. Loshchilov, T. Glasmachers, and H. G. Beyer. Large scale black-box optimization by limited-memory matrix adaptation. *IEEE Trans. Evol. Comput.* **23** (April 2019), 353–358.
36. S. Lucidi and M. Sciandrone. A derivative-free algorithm for bound constrained optimization. *Comput. Optim. Appl.* **21** (2002), 119–142.
37. L. Lukšan, L and J. Vlcek. Sparse and partially separable test problems for unconstrained and equality constrained optimization. ICS AS CR, 1999. 30 s. Technical Report, V-767.
38. J. J. Moré and S. M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM J. Optim.* **20** (January 2009), 172–191.
39. A. Neumaier, H. Fendl, H. Schilly, and Thomas Leitner. VXQR: derivative-free unconstrained optimization based on QR factorizations. *Soft Comput.* **15** (September 2010), 2287–2298.
40. M. Porcelli and P. Toint. Global and local information in structured derivative free optimization with BFO. *arXiv: Optimization and Control* (2020).
41. M. J. D. Powell. UOBYQA: unconstrained optimization by quadratic approximation. *Math. Program.* **92** (May 2002), 555–582.
42. M. J. D. Powell. Developments of NEWUOA for minimization without derivatives. *IMA. J. Numer. Anal.* **28** (February 2008), 649–664.
43. L. M. Rios and N. V. Sahinidis. Derivative-free optimization: a review of algorithms and comparison of software implementations. *J. Global. Optim.* **56** (July 2012), 1247–1293.
44. V. J. Torczon. *Multidirectional search: A direct search algorithm for parallel machines.* PhD thesis, Diss., Rice University (1989).
45. B. Van Dyke and T. J. Asaki. Using QR decomposition to obtain a new instance of mesh adaptive direct search with uniformly distributed polling directions. *J. Optim. Theory Appl* **159** (June 2013), 805–821.
46. S. M. Wild, R. G. Regis, and C. A. Shoemaker. ORBIT: Optimization by radial basis function interpolation in trust-regions. *SIAM J. Sci. Comput.* **30** (January 2008), 3197–3219.

47. M. H Wright. Direct search methods: Once scorned, now respectable. *Pitman Research Notes in Math. Series* (1996), 191–208.