Project number IST-25582

CGL
Computational Geometric Learning

Variable Metric Random Pursuit: Experimental Validation

STREP

Information Society Technologies

Period covered:	November 1, 2012–October 31, 2013
Date of preparation:	November 11, 2013
Date of revision:	November 13, 2013
Start date of project:	November 1, 2010
Duration:	3 years
Project coordinator name:	Joachim Giesen (FSU)
Project coordinator organisation:	Friedrich-Schiller-Universität Jena Jena, Germany

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

Variable Metric Random Pursuit: Experimental Validation

S. U. Stich · C. L. Müller · B. Gärtner

Received: date / Accepted: date

Abstract We consider unconstrained randomized optimization of smooth convex objective functions in the gradient-free setting. We analyze Random Pursuit (RP) algorithms with fixed (F-RP) and variable metric (V-RP). The algorithms only use zeroth-order information about the objective function and compute an approximate solution by repeated optimization over randomly chosen one-dimensional subspaces. The distribution of search directions is dictated by the chosen metric. Variable Metric RP uses novel variants of a randomized zeroth-order Hessian approximation scheme recently introduced by Leventhal and Lewis (D. Leventhal and A. S. Lewis., Optimization 60(3), 329–245, 2011). We here present (i) a refined analysis of the expected single step progress of RP algorithms and their global convergence on (strictly) convex functions and (ii) novel convergence bounds for V-RP on quadratic functions. We also quantify how well the employed metric needs to match the local geometry of the function in order for the RP algorithms to converge with the best possible rate on strongly convex functions. Our theoretical results are accompanied by extensive numerical experiments, comparing V-RP with the derivative-free schemes CMA-ES, Implicit Filtering, Nelder-Mead, NEWUOA, Pattern-Search and Nesterov's gradient-free algorithms.

Keywords gradient-free optimization · convex optimization · variable metric · line search

1 Introduction

Since its inception by Davidon in the late 1950's [4] variable metric methods have become a cornerstone in first-order (non-)convex continuous optimization. Among the many instances of variable metric schemes Quasi-Newton methods such as the BFGS scheme [3,5,6,26] are ubiquitous in all areas of science and engineering. In zeroth-order (or gradient-free) optimization, the idea of using a variable metric guiding the search for local or global optima has surprisingly been used to a far less extent. Although "directional adaptation" has been conjectured to be useful for randomized gradient-free schemes in the late 1960's [25] the literature on this topic is scarce and scattered across different communities ranging from electrical

The project CG Learning acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 255827.

S. U. Stich, C. L. Müller, B. Gärtner
Institute of Theoretical Computer Science, ETH Zürich,
E-mail: {stich,christian.mueller,gaertner}@inf.ethz.ch

engineering, optimal control, bio-inspired optimization to mathematical programming. Important examples include the Gaussian Adaptation algorithm developed by Kjellström and Taxen [14,17] in the context of analog circuit design, Marti's controlled random search schemes using concepts from optimal control [16], and the arguably most popular scheme, Hansen's Evolution Strategy with Covariance Matrix Adaptation (CMA-ES) [7] that emerged in the bio-inspired optimization community.

Despite their great appeal in practice many randomized gradient-free variable metric schemes lack a thorough theoretical convergence analysis. A marked exception is Leventhal and Lewis' recent work on Randomized Hessian approximation [15]. We here adopt some of their ideas and extend our framework of Random Pursuit (RP) [29], eventually leading to Variable Metric Random Pursuit (V-RP) schemes. We solely consider optimization problems of the kind:

$$\min f(\mathbf{x}) \quad \text{subject to} \quad \mathbf{x} \in \mathbb{R}^n\,, \tag{1}$$

where f is a smooth convex function. We assume that there is a global minimum and that the curvature of the function f is bounded from above. Moreover, we assume that we have only access to function values of f. No analytic gradient or higher order information about f is available.

To motivate Variable Metric Random Pursuit, let us first sketch the working mechanism of standard Random Pursuit on an illustrative example. Each iteration of standard Random Pursuit consists of two steps: (i) a random direction is sampled from an isotropic probability distribution; (ii) the next iterate is chosen such as to (approximately) minimize the objective function along this direction (using an approximate line search procedure). In [29] we have shown that the expected error in function value decreases by a factor of

$$\left(1 - \frac{m}{n\ell}\right)$$

in every step, if $m > 0$ and $\ell > 0$ are parameters of quadratic functions that bound the difference between f and any of its linear approximations from below and above[1]; more precisely,

$$\frac{m}{2}\left\|\mathbf{y}-\mathbf{x}\right\|^2 \leq \ell_{\mathbf{x}}(\mathbf{y}) := f(\mathbf{y}) - f(\mathbf{x}) - \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle \leq \frac{\ell}{2}\left\|\mathbf{y}-\mathbf{x}\right\|^2 \tag{2}$$

is assumed to hold for all $\mathbf{x}, \mathbf{y}$. For twice differentiable functions this condition is equivalent to an uniform lower and upper bound on the Hessian $H(\mathbf{x})$: $m \leq H(\mathbf{x}) \leq \ell$. As an example, let us consider the function

$$f_0(x_1, x_2) = 100x_1^2 + x_2^2,$$

for which $\ell_{\mathbf{x}}(\mathbf{y}) = 100(x_1 - y_1)^2 + (x_2 - y_2)^2$. This means that $m = 2$ and $\ell = 200$ are the best possible parameters in (2), and the progress rate in every step is no better than $(1 - 1/200)$. This also matches our intuition: every level set of f_0 is a long and skinny ellipse, stretching out along the x_2-axis; if we start from a point close to the x_2 axis, the progress in a step will be small, unless we almost sample in x_2-direction.

For this particular function f_0, it would be better to sample from an anisotropic distribution that favors the x_2-direction. Once we fix such an anisotropic sampling distribution, however, other functions become "bad"; in fact, without prior knowledge about f, anisotropic sampling makes no sense at all. Here is where the "variable metric" approach comes in. The idea is to gradually *adapt* the sampling distribution

[1] These inequalities are usually referred to as strong-convexity (the left one) and Lipschitz continuity of the gradient. See Section 3.

to the function f while we run the algorithm. Suppose that we can somehow estimate the Hessians at the various iterates. Under the assumption that f is wedged between two quadratic functions—whose Hessians are not necessarily multiples of the identity, as in (2)—these estimates will allow us to learn a suitable metric that guides the sampling distribution. In case of f_0, we would start with the isotropic one and then converge to a distribution that indeed favors the x_2-direction with the right proportion.

In this contribution we present a framework for analyzing the convergence behavior of Random Pursuit algorithms on convex functions. In a first step we analyze the Fixed Metric Random Pursuit (F-RP) algorithm, a natural extension of Random Pursuit with an arbitrary but fixed anisotropic sampling distribution. We obtain a progress rate that depends on the chosen distribution, as well as on lower and upper quadratic approximations to f. Our progress rate bound improves over the one we would get from the standard Random Pursuit analysis [29], after applying an affine transformation that maps the sampling distribution back to the isotropic one. The improvement is due to the fact that the new analysis takes the whole spectrum of the quadratic upper bound into account rather than just a scalar value ℓ.

In a second step we equip Random Pursuit with a randomized scheme to update the metric that defines the sampling distribution in every step: the Variable Metric Random Pursuit. We present three novel variants of an update scheme recently proposed by Leventhal and Lewis [15]. These learning schemes are generic in the sense that they work for all convex functions and do not require any prior knowledge of the function's shape. We then concentrate on the class of convex quadratic functions and rigorously prove in this case that the sampling distribution converges to a distribution that yields asymptotically optimal (and function-independent) progress rates. We also provide bounds on the number of iterations that the algorithm requires in order to attain optimal progress rates with high probability. The proposed schemes are easily parallelizable, thus allowing a computational speed-up of the update schemes on multi-core machines.

The remainder of the paper is structured as follows. In Section 2 we give a generic description of the different Random Pursuit algorithms and their essential building blocks. We introduce all relevant mathematical definitions such as matrix upper and lower bounds of convex functions and expressions for certain scalar and matrix expectations in Section 3. We derive the expected single-step progress and global convergence of F-RP in Section 4. Section 5 is dedicated to Variable Metric Random Pursuit. We derive theoretical convergence results and show an illustrative numerical example. In Section 6 we present extensive numerical tests and compare the performance of V-RP with state-of-the-art (randomized and deterministic) search schemes. We discuss the key results of the paper and outline future research goals in Section 7.

2 Fixed and Variable Metric Random Pursuit

All Random Pursuit algorithms are designed for problems as defined in (1) where f is assumed to be a differentiable convex function with the property that it has a minimum along every line in $\mathbb{R}^n$. Before stating the formal definition of the considered RP algorithms we need to define one indispensable primitive.

Definition 1 (Line search oracle) For $\mathbf{x} \in \mathbb{R}^n$, a direction $\mathbf{u} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ (not necessarily of unit length), and a convex function f, a function $\mathtt{LS}_f \colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ with

$$\mathtt{LS}_f(\mathbf{x}, \mathbf{u}) = \arg\min_{h \in \mathbb{R}} f(\mathbf{x} + h\mathbf{u}) \tag{3}$$

is called an *exact line search oracle*.

For simplicity we assume here that we have access to an exact line search oracle. However, approximate line search schemes suffice to establish convergence of the Random Pursuit algorithms (see Section 2.1).

The two RP schemes considered here are summarized in Fig. 1. In Fixed Metric

F-RP$(f, \mathbf{x}_0, \Sigma, N)$

Output: Approximate solution x_N to (1)
1 **for** $k = 1$ *to* N **do**
2 $\quad \mathbf{u}_k \sim \mathcal{N}(\mathbf{0}, \Sigma)$
3 $\quad \mathbf{x}_k \leftarrow \mathtt{LS}_f(\mathbf{x}_{k-1}, \mathbf{u}_k)$
4 **return** $\mathbf{x}_N$

V-RP$(f, \mathbf{x}_0, B_0, N)$

Output: Approximate solution x_N to (1)
1 **for** $k = 1$ *to* N **do**
2 $\quad B_k \leftarrow \mathtt{updateHess}(f, \mathbf{x}, B_{k-1})$
3 $\quad \mathbf{u}_k \sim \mathcal{N}(\mathbf{0}, B_k^{-1})$
4 $\quad \mathbf{x}_k \leftarrow \mathtt{LS}_f(\mathbf{x}_{k-1}, \mathbf{u}_k)$
5 **return** $\mathbf{x}_N$

Fig. 1 Fixed Metric Random Pursuit (left panel) and the Variable Metric version (right panel). The generic sub-routine `updateHess` on line 2 exemplifies any function that generates the metric B_k in step k. Three specific instantiations are discussed in Sec. 5 (cf. Figs. 3, 4).

Random Pursuit (F-RP) a direction $\mathbf{u} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ is sampled from a multivariate normal distribution with fixed covariance Σ at iteration k of the algorithm. The next iterate $\mathbf{x}_k$ is calculated from the current iterate $\mathbf{x}_{k-1}$ as

$$\mathbf{x}_k := \mathbf{x}_{k-1} + \mathtt{LS}_f(\mathbf{x}_{k-1}, \mathbf{u}) \cdot \mathbf{u}\,. \tag{4}$$

This algorithm only requires function evaluations in addition to the line search oracle. No first or second-order information about the objective is needed. We emphasize that besides the starting point no further input parameters describing function properties (such as curvature constant etc.) are necessary. The actual run time will, however, depend on the specific properties of the objective function and on the choice of the covariance matrix Σ, as detailed in Section 4. Finding a good estimate of the parameter Σ *a priori* might be costly or impossible, for instance, if the optimal choice depends on the current position $\mathbf{x}_k$ and thus changes over time.

Variable Metric Random Pursuit (V-RP) comprises an independent process that gives an approximation of the Hessian at each iteration. The inverse of the Hessian is then used as covariance matrix in the multivariate normal distribution to generate the current search direction. In principle, any deterministic or randomized gradient-free estimator can be used for this purpose. In Section 5 we present three extensions to a Randomized Hessian approximation scheme recently proposed in [15].

2.1 Approximate line search oracles

Access to an exact line search oracle (3) is typically not required to establish convergence of the RP algorithms. This is of importance in practical applications. Analysis in [29] shows that an inexact or *approximate line search oracle* (ALS) is sufficient to establish convergence of a simple instance of F-RP (with $\Sigma = I_n$ the n-dimensional identity matrix). Several accuracy measures can be used to define inexact line search oracles, for instance, using *absolute* or *relative* accuracy measures (cf [29]).

Definition 2 (Line search oracle with relative error) For $0 \leq \mu \leq 1$, $\mathbf{x} \in \mathbb{R}^n$, a direction $\mathbf{u} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ (not necessarily of unit length), and a convex function f, a function $\mathtt{ALS}_f \colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ with

$$f(\mathbf{x} + \mathtt{ALS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u}) \leq f(\mathbf{x}) - \mu(f(\mathbf{x}) - f(\mathbf{x} + \mathtt{LS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u})) \tag{5}$$

is called an approximate line search oracle with *accuracy* μ.

Note that $|\mathtt{ALS}_f(\mathbf{x},\mathbf{u}) - \mathtt{LS}_f(\mathbf{x},\mathbf{u})|$ can be large if $\|\mathbf{x} + \mathtt{LS}_f(\mathbf{x},\mathbf{u})\cdot\mathbf{u}\|$ is large as well. As we measure deviations only on a relative and not on an absolute scale, such an inexact line search oracle can efficiently be implemented with binary search (dichotomy) using function evaluations only.

As we will see in Section 4, the number of iterations of an RP algorithm to reach a target accuracy is proportional to μ^{-1}. This means that for $\mu = \frac{1}{2}$, twice as many iterations are necessary to reach the same accuracy as with the choice $\mu = 1$, respectively. We note that is is straightforward to extend our convergence results to more general settings. For instance, we can vary the accuracy parameter μ in every iteration as long as μ stays positive, or the distribution of μ is independent of $\mathbf{x}$ and $\mathbf{u}$ (see also the concrete implementations in Section 6.2.1).

3 Definitions and Notations

3.1 Quadratic norms

Let PD_n denote the set of symmetric positive definite $n \times n$ matrices. With respect to $A \in \mathrm{PD}_n$, we can define an 'anisotropic' inner product and a corresponding norm by

$$\langle \mathbf{x}, \mathbf{y}\rangle_A := \mathbf{y}^T A\mathbf{x}\,, \qquad \text{and} \qquad \|\mathbf{x}\|_A^2 := \langle \mathbf{x}, \mathbf{x}\rangle_A\,,$$

for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. In statistics the induced metric is also known as the Mahalanobis metric. We observe that

$$\lambda_{\min}(A)\,\|\mathbf{x}\|^2 \le \|\mathbf{x}\|_A^2 \le \lambda_{\max}(A)\,\|\mathbf{x}\|^2\,, \tag{6}$$

due to $\lambda_{\min}(A) = \min\{\mathbf{x}^T A\mathbf{x} : \|x\| = 1\}$ and $\lambda_{\max}(A) = \max\{\mathbf{x}^T A\mathbf{x} : \|x\| = 1\}$. We will frequently need the following lemma.

Lemma 1 *Let $A, B \in \mathrm{PD}_n, \mathbf{x} \in \mathbb{R}^n$ with $\mathbf{x} \neq \mathbf{0}$. Then*

$$\lambda_{\min}(B^{-1}A) \le \frac{\|\mathbf{x}\|_A^2}{\|\mathbf{x}\|_B^2} \le \lambda_{\max}(B^{-1}A)\,.$$

Proof Let C be the matrix such that $B = CC^T$ and set $\mathbf{y} = C^T\mathbf{x}$. Then we have

$$\frac{\|\mathbf{x}\|_A}{\|\mathbf{x}\|_B} = \frac{\|\mathbf{y}\|_{C^{-1}AC^{-T}}}{\|\mathbf{y}\|}\,,$$

where C^{-T} is a shorthand for $(C^T)^{-1}$. Using (6), we can argue that

$$\lambda_{\min}(C^{-1}AC^{-T}) \le \frac{\|\mathbf{y}\|^2_{C^{-1}AC^{-T}}}{\|\mathbf{y}\|^2} \le \lambda_{\max}(C^{-1}AC^{-T})\,.$$

It remains to show that the matrices $C^{-1}AC^{-T}$ and $C^{-T}C^{-1}A = B^{-1}A$ have the same eigenvalues. This follows from the "Rotation" Lemma 2. □

Lemma 2 (Rotation) *Let $L \in \mathbb{R}^{n\times n}$, and let $C \in \mathbb{R}^{n\times n}$ be an invertible matrix. Then the two matrices*

$$P := LCC^T \text{ and } Q := C^T LC$$

have the same eigenvalues.

Proof We show that P and Q have the same characteristic polynomial. For this, we first observe that

$$P - tI_n = C^{-T}(Q - tI_n)C^T.$$

It follows that

$$\begin{aligned}\det(P - tI_n) &= \det(C^{-T}(Q - tI_n)C^T) = \det(C^{-T})\det(C^T)\det(Q - tI_n)\\ &= \det(Q - tI_n)\,. \qquad \square\end{aligned}$$

3.2 Sampling distributions

The RP algorithms discussed in this paper randomly sample direction vectors every iteration. We use a multivariate normal distribution to generate the corresponding samples. One drawback of this procedure is the fact that two samples that share the same direction must not necessarily be of the same length. Whilst this is no issue in practice (due to concentration and the used line search oracle) it turns out to be beneficial for the upcoming convergence proofs if we restrict ourselves to *normalized* samples.

Definition 3 (Multivariate normal distribution) The multivariate normal distribution arises from independent and identically distributed (i.i.d.) standard normals. The vector $\mathbf{v} \in \mathbb{R}^n$ is standard multivariate normally distributed, i.e., $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, I_n)$ if $v_i \sim \mathcal{N}(0,1)$ for $i = 1, \ldots, n$ and I_n the n-dimensional identity matrix. For $\boldsymbol{\mu} \in \mathbb{R}^n, \Sigma = CC^T \in \mathrm{PD}_n$, where PD_n is the set of symmetric positive definite matrices, $\mathbf{u} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ is multivariate normal with mean $\boldsymbol{\mu}$ and covariance Σ if $\mathbf{u} = \boldsymbol{\mu} + C\mathbf{v}$ and $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, I_n)$.

By normalization of the vectors according to the norm induced by the inverse covariance Σ^{-1} we obtain normalized directions. Geometrically speaking, the set of normalized samples lies on the surface of a hyper-ellipsoid which is defined by Σ^{-1}.

Definition 4 (Normalized multivariate normal distribution) Let $\Sigma \in \mathrm{PD}_n$. We denote by $\bar{\mathcal{N}}(0, \Sigma)$ the distribution arising from the image of the normal distribution $\mathcal{N}(0, \Sigma)$ under the mapping $T(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|_{\Sigma^{-1}}$.

For example, $\mathbf{u} \sim \bar{\mathcal{N}}(0, I_n)$ denotes the uniform distribution over all unit length vectors subject to the standard Euclidean norm (the uniform distribution on the unit $(n-1)$-sphere).

3.3 Quadratic bounds

We now introduce some important inequalities that are useful for the subsequent analysis. We always assume that the objective function f is differentiable and convex. The latter property is equivalent to

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \,, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \,. \tag{7}$$

We also require that the curvature of f is bounded. However, we allow for different curvatures depending on the direction. By this we mean that for some fixed symmetric and positive definite matrix $L_1 \in \mathrm{PD}_n$,

$$f(\mathbf{y}) - f(\mathbf{x}) - \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \leq \frac{1}{2} \|\mathbf{x} - \mathbf{y}\|_{L_1}^2 \,, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \,. \tag{8}$$

We will also refer to this inequality as the *(matrix) quadratic upper bound*. It means that the deviation of f from any of its linear approximations can be bounded by a quadratic function. We denote by $C^1_{L_1}$ the class of (once) differentiable convex functions for which the quadratic upper bound holds with parameter L_1.

A differentiable function is *strongly convex* with parameter $M \in \mathrm{PD}_n$ if the *(matrix) quadratic lower bound*

$$f(\mathbf{y}) - f(\mathbf{x}) - \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \geq \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_M^2 \,, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \,, \tag{9}$$

holds. Let $\mathbf{x}^*$ be the unique minimizer of a strongly convex function f with parameter M. Then equation (9) implies this useful relation:

$$\frac{1}{2}\left\|\mathbf{x}-\mathbf{x}^*\right\|_M^2 \leq f(\mathbf{x})-f(\mathbf{x}^*) \leq \frac{1}{2}\|\nabla f(\mathbf{x})\|_{M^{-1}}^2\,, \quad \forall \mathbf{x}\in\mathbb{R}^n\,. \tag{10}$$

The former inequality uses $\nabla f(\mathbf{x}^*)=0$, and the latter one follows from (9) via

$$\begin{aligned} f(\mathbf{x}^*) &\geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{x}^*-\mathbf{x}\rangle + \frac{1}{2}\left\|\mathbf{x}^*-\mathbf{x}\right\|_M^2 \\ &\geq f(\mathbf{x}) + \min_{\mathbf{y}\in\mathbb{R}^n}\left(\langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + \frac{1}{2}\left\|\mathbf{y}-\mathbf{x}\right\|_M^2\right) = f(\mathbf{x}) - \frac{1}{2}\|\nabla f(\mathbf{x})\|_{M^{-1}}^2 \end{aligned}$$

by standard calculus.

3.4 Expectations involving normally distributed random variables

We here review and derive certain facts about the moments of the standard normal distribution for the later convergence analysis.

Lemma 3 *Let $\mathbf{u}\in\mathcal{N}(\mathbf{0},\Sigma)$ be drawn from the multivariate normal distribution over $\mathbb{R}^n$ with covariance $\Sigma\in\mathrm{PD}_n$.*

(i) Then for all indices i,j,k,l,

$$\mathbb{E}[u_i]=0\,, \qquad \mathbb{E}[u_iu_j]=\Sigma_{ij}\,, \qquad \mathbb{E}\left[\mathbf{u}\mathbf{u}^T\right]=\Sigma\,,$$

and

$$\mathbb{E}[u_iu_ju_ku_l]=\Sigma_{ij}\Sigma_{kl}+\Sigma_{ik}\Sigma_{jl}+\Sigma_{il}\Sigma_{jk}\,.$$

(ii) Let $A\in\mathrm{SYM}_n$ be a symmetric $n\times n$ matrix. Then

$$\mathbb{E}\left[\mathbf{u}^TA\mathbf{u}\right]=\mathrm{Tr}[A\Sigma]\,, \qquad \mathbb{E}\left[\mathbf{u}^TA\mathbf{u}\cdot\mathbf{u}\mathbf{u}^T\right]=\mathrm{Tr}[A\Sigma]\Sigma+2\Sigma A\Sigma\,.$$

Proof The first three equalities in part (i) are easy consequences of the above definition. The last one is known as *Isserlis' Theorem* [11].

The matrix equations in (ii) easily follow from (i) via

$$\mathbb{E}\left[\mathbf{u}^TA\mathbf{u}\right]=\sum_{i,j=1}^n\mathbb{E}\left[u_iu_jA_{ij}\right]=\sum_{i,j=1}^n\Sigma_{ij}A_{ij}=\mathrm{Tr}[A^T\Sigma]\,,$$

and

$$\begin{aligned} \left(\mathbb{E}\left[\mathbf{u}^TA\mathbf{u}\cdot\mathbf{u}\mathbf{u}^T\right]\right)_{ij} &= \sum_{k,l=1}^n\mathbb{E}\left[u_iu_ju_ku_lA_{kl}\right] \\ &= \sum_{k,l=1}^n A_{kl}\left(\Sigma_{ij}\Sigma_{kl}+\Sigma_{ik}\Sigma_{jl}+\Sigma_{il}\Sigma_{jk}\right) \\ &= \mathrm{Tr}[A\Sigma]\Sigma_{ij}+\left(\Sigma A\Sigma\right)_{ij}+\left(\Sigma A\Sigma\right)_{ji}\,, \end{aligned}$$

using symmetry of $\Sigma A\Sigma$. □

Lemma 4 *Let $\mathbf{v}\sim\bar{\mathcal{N}}(0,\Sigma)$ normalized with $\Sigma=CC^T\in\mathrm{PD}_n$ and let $A\in\mathrm{SYM}_n$. Then*

$$\textit{(i)}\quad \mathbb{E}\left[\mathbf{v}\mathbf{v}^T\right]=\frac{\Sigma}{n}\,, \quad \textit{(ii)}\quad \mathbb{E}\left[\mathbf{v}^TA\mathbf{v}\right]=\frac{\mathrm{Tr}[A\Sigma]}{n}\,, \quad \textit{and}$$
$$\textit{(iii)}\quad \mathbb{E}\left[\mathbf{v}^TA\mathbf{v}\cdot\mathbf{v}\mathbf{v}^T\right]=\frac{\mathrm{Tr}[A\Sigma]I_n+2\Sigma A\Sigma}{n(n+2)}\,.$$

Proof Let $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \Sigma)$. The random vector $\mathbf{w} = \mathbf{u}/\left\|\mathbf{u}\right\|_{\Sigma^{-1}} = \mathbf{u}/\left\|C^{-1}\mathbf{u}\right\|_2$ has the same distribution as $\mathbf{v}$ by definition. In particular,

$$\mathbb{E}_{\mathbf{v}}\left[\mathbf{v}^T A\mathbf{v}\cdot\mathbf{v}\mathbf{v}^T\right] = \mathbb{E}_{\mathbf{u}}\left[\frac{\mathbf{u}^T A\mathbf{u}\cdot\mathbf{u}\mathbf{u}^T}{\left\|\mathbf{u}\right\|_{\Sigma^{-1}}^4}\right] = \frac{\mathbb{E}_{\mathbf{u}}\left[\mathbf{u}^T A\mathbf{u}\cdot\mathbf{u}\mathbf{u}^T\right]}{\mathbb{E}_{\mathbf{u}}\left[\left\|\mathbf{u}\right\|_{\Sigma^{-1}}^4\right]},$$

where the equality follows from independence of $\frac{\mathbf{u}^T A\mathbf{u}\cdot\mathbf{u}\mathbf{u}^T}{\left\|\mathbf{u}\right\|_{\Sigma^{-1}}^4}$ and $\left\|\mathbf{u}\right\|_{\Sigma^{-1}}^4$ (cf. [8]). By Lemma 3 (ii), the numerator equals $\mathrm{Tr}[A\Sigma]I_n + 2\Sigma A\Sigma$. For the denominator we again use Lemma 3 (ii) with $A = \Sigma^{-1}$ to get

$$\mathbb{E}\left[\left\|\mathbf{u}\right\|_{\Sigma^{-1}}^4\right] = \mathrm{Tr}\left[C^{-T}\mathbb{E}\left[\mathbf{u}^T\Sigma^{-1}\mathbf{u}\cdot\mathbf{u}\mathbf{u}^T\right]C^{-1}\right] = n(n+2),$$

where C^{-T} is used as shorthand for $(C^T)^{-1}$. The proof of the first claim uses similar arguments and the observation that

$$\mathbb{E}\left[\left\|\mathbf{u}\right\|_{\Sigma^{-1}}^2\right] = \mathrm{Tr}\left[\mathbb{E}[\mathbf{u}\Sigma^{-1}\mathbf{u}]\right] = n,$$

again by Lemma 3 (ii). The second claim follows from the equality $\mathbf{v}^T A\mathbf{v} = \mathrm{Tr}[A\cdot\mathbf{v}\mathbf{v}^T]$ and (i). □

We finally need the following lemma on the expectation of certain scalar products:

Lemma 5 *Let $\mathbf{v} \in \bar{\mathcal{N}}(\mathbf{0}, \Sigma)$ with $\Sigma \in \mathrm{PD}_n$, let $A \in \mathrm{PD}_n$ and $\mathbf{x} \in \mathbb{R}^n$. Then*

$$\mathbb{E}\left[\langle\mathbf{x},\mathbf{v}\rangle\,\mathbf{v}\right] = \frac{\Sigma\mathbf{x}}{n}, \quad \textit{and} \quad \mathbb{E}\left[\left\|\langle\mathbf{x},\mathbf{v}\rangle\,\mathbf{v}\right\|_A^2\right] = \frac{\mathrm{Tr}[A\Sigma]\left\|\mathbf{x}\right\|_\Sigma^2 + 2\left\|\mathbf{x}\right\|_{\Sigma A\Sigma}^2}{n(n+2)}.$$

Proof This is an immediate application of Lemma 4. We calculate

$$\mathbb{E}\left[\langle\mathbf{x},\mathbf{v}\rangle\,\mathbf{v}\right] = \mathbb{E}\left[\mathbf{v}\mathbf{v}^T\mathbf{x}\right] = \mathbb{E}\left[\mathbf{v}\mathbf{v}^T\right]\mathbf{x} = \frac{1}{n}\Sigma\mathbf{x},$$

using Lemma 4. For the second moment we get

$$\mathbb{E}\left[\left\|\langle\mathbf{x},\mathbf{v}\rangle\,\mathbf{v}\right\|_A^2\right] = \mathbb{E}[\mathbf{x}^T\mathbf{v}\mathbf{v}^T A\mathbf{v}\mathbf{v}^T\mathbf{x}] = \mathbf{x}^T\,\mathbb{E}[\mathbf{v}\mathbf{v}^T A\mathbf{v}\mathbf{v}^T]\mathbf{x},$$

and the claim follows from Lemma 4 and the observation that $\mathbf{v}\cdot\mathbf{v}^T A\mathbf{v}\cdot\mathbf{v}^T = \mathbf{v}^T A\mathbf{v}\cdot\mathbf{v}\mathbf{v}^T$. □

4 Convergence of Fixed Metric Random Pursuit

To prepare the convergence proof of Algorithm F-RP, we study the expected progress in a single step, which is the quantity

$$f(\mathbf{x}_k) - \mathbb{E}\left[f(\mathbf{x}_{k+1}) \mid \mathbf{x}_k\right].$$

We will now derive the global convergence rates for convex and strongly convex functions.

4.1 Single step progress

Once a search direction is determined, the subsequent iterate is chosen according to Equation (4). As the step size is determined by the line search oracle, we can derive the following lower bound on the single step progress.

Lemma 6 *Let $f \in C^1_{L_1}$, $\mathbf{x} \in \mathbb{R}^n$, covariance $\Sigma \in \mathrm{PD}_n$ direction $\mathbf{u} \sim \bar{\mathcal{N}}(0, \Sigma)$, and* ALS *an inexact line search with accuracy $0 \leq \mu \leq 1$. Then for $\mathbf{x}_+ = \mathbf{x} + \mathtt{ALS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u}$, the next iterate after one step of Algorithm F-RP, is holds*

$$\mathbb{E}_{\mathbf{u}}[f(\mathbf{x}_+) \mid \mathbf{x}] \leq f(\mathbf{x}) - \frac{h\mu}{n} \langle \nabla f(\mathbf{x}), \mathbf{x} - \mathbf{z} \rangle + \frac{h^2 \mu}{2n(n+2)} \left(\mathrm{Tr}[L_1 \Sigma] \, \|\mathbf{z} - \mathbf{x}\|^2_{\Sigma^{-1}} + 2 \, \|\mathbf{z} - \mathbf{x}\|^2_{L_1} \right) , \quad (11)$$

for every $h \geq 0$ and $\mathbf{z} \in \mathbb{R}^n$.

Proof We first study the progress for a *fixed* direction $\mathbf{u}$ and apply the expectation only later. Let $\hat{\mathbf{x}} := \mathbf{x} + \mathtt{LS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u}$ denote the optimum along direction $\mathbf{u}$. As a simple consequence of Definition 1 we have

$$f(\hat{\mathbf{x}}) = f(\mathbf{x} + \mathtt{LS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u}) \leq f(\mathbf{x} + t\mathbf{u}) \, ,$$

for every $t \in \mathbb{R}$. Now we apply the quadratic upper bound (8) with $\mathbf{y} = \mathbf{x} + t\mathbf{u}$ and deduce

$$f(\hat{\mathbf{x}}) \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), t\mathbf{u} \rangle + \frac{1}{2} \left\| t^2 \mathbf{u} \right\|^2_{L_1} \, , \quad (12)$$

for every $t \in \mathbb{R}$. We want to bound $\mathbb{E}_{\mathbf{u}}[f(\hat{\mathbf{x}}) \mid \mathbf{x}]$ from above. For this we choose t such that we are able to compute the expectations of the right hand side. We set $t = h \left\langle \Sigma^{-1}(\mathbf{x} - \mathbf{z}), \mathbf{u} \right\rangle$ where $h > 0$ and $\mathbf{z} \in \mathbb{R}^n$ are the parameters from the lemma. With Lemma 5 we compute

$$\mathbb{E}_{\mathbf{u}}[\left\langle \Sigma^{-1}(\mathbf{x} - \mathbf{z}), \mathbf{u} \right\rangle \mathbf{u} \mid \mathbf{x}] = \mathbf{x} - \mathbf{z} \, ,$$

$$\mathbb{E}_{\mathbf{u}} \left[\left\| \left\langle \Sigma^{-1}(\mathbf{x} - \mathbf{z}), \mathbf{u} \right\rangle \cdot \mathbf{u} \right\|^2_{L_1} \mid \mathbf{x} \right] = \frac{\mathrm{Tr}[L_1 \Sigma] \, \|\mathbf{x} - \mathbf{z}\|^2_{\Sigma^{-1}} + 2 \, \|\mathbf{x} - \mathbf{z}\|^2_{L_1}}{n(n+2)} \, .$$

Together with (12), these two inequalities imply already the lemma for $\hat{\mathbf{x}}$ (or alternatively, for $\mathbf{x}_+$ in case $\mu = 1$). For the general case, we simply recall the definition (5) of the line search with accuracy μ and the lemma follows. □

Now we are ready to eliminate the two free parameters h and $\mathbf{z}$ in our bound (11) for the single step progress. It turns out that letting $\mathbf{z}$ be one of the minimizers of f is advantageous.

Lemma 7 *Let $f \in C^1_{L_1}$, $\mathbf{x} \in \mathbb{R}^n$, covariance $\Sigma \in \mathrm{PD}_n$ direction $\mathbf{u} \sim \bar{\mathcal{N}}(0, \Sigma)$,* ALS *an inexact line search with accuracy $0 < \mu \leq 1$, and $\mathbf{x}_+ = \mathbf{x} + \mathtt{ALS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u}$ the next iterate after one step of Algorithm F-RP. In addition, let $\mathbf{x}^* \in \mathbb{R}^n$ be one of the minimizers of f. Then, for every positive $h \geq 0$ it holds*

$$\mathbb{E}_{\mathbf{u}} \left[f(\mathbf{x}_+) - f(\mathbf{x}^*) \mid \mathbf{x} \right] \leq \left(1 - \frac{h\mu}{n} \right) (f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{h^2 \mu}{2n(n+2)} \left(\frac{\mathrm{Tr}\left[L_1 \Sigma \right]}{\lambda_{\min}(\Sigma L_1)} + 2 \right) \|\mathbf{x} - \mathbf{x}^*\|^2_{L_1} \, . \quad (13)$$

Proof We apply the previous Lemma 6 with parameter $\mathbf{z} = \mathbf{x}^*$. Using the definition of convexity (7) we can bound the term $\langle \nabla f(\mathbf{x}), \mathbf{x} - \mathbf{x}^* \rangle$ from below by $f(\mathbf{x}) - f(\mathbf{x}^*)$. By subtracting $f(\mathbf{x}^*)$ on both sides of (11) we thus arrive at

$$\mathbb{E}_{\mathbf{u}} \left[f(\mathbf{x}_+) - f(\mathbf{x}^*) \mid \mathbf{x} \right] \leq \left(1 - \frac{h\mu}{n} \right) \left(f(\mathbf{x}) - f(\mathbf{x}^*) \right) + \frac{h^2 \mu}{2n(n+2)} \left(\operatorname{Tr}\left[L_1 \Sigma \right] \|\mathbf{x} - \mathbf{x}^*\|^2_{\Sigma^{-1}} + 2 \|\mathbf{x} - \mathbf{x}^*\|^2_{L_1} \right) . \quad (14)$$

Finally, we bound the Σ^{-1}-norm from above with Lemma 1. We get

$$\|\mathbf{x} - \mathbf{x}^*\|^2_{\Sigma^{-1}} \leq \lambda_{\max}(L_1^{-1}\Sigma^{-1}) \|\mathbf{x} - \mathbf{x}^*\|_{L_1} , \quad (15)$$

and the lemma follows from $\lambda_{\max}(L_1^{-1}\Sigma^{-1}) = 1/\lambda_{\min}(\Sigma L_1)$. □

We can also use Lemma 6 with different parameters. Here we use $\mathbf{z} = \mathbf{x} - L_1^{-1}\nabla f(\mathbf{x})$ and subsequently determine the last free parameter h to minimize the right hand side.

Lemma 8 *Let $f \in C^1_{L_1}$, $\mathbf{x} \in \mathbb{R}^n$ such that $\nabla f(\mathbf{x}) \neq \mathbf{0}$, covariance $\Sigma \in \mathrm{PD}_n$ direction $\mathbf{u} \sim \mathcal{N}(0, \Sigma)$,* `ALS` *an inexact line search with accuracy $0 < \mu \leq 1$, and $\mathbf{x}_+ = \mathbf{x} + \mathtt{ALS}_f(\mathbf{x}, \mathbf{u}) \cdot \mathbf{u}$ the next iterate after one step of Algorithm F-RP. Then*

$$\mathbb{E}_{\mathbf{u}} \left[f(\mathbf{x}_+) \mid \mathbf{x} \right] \leq f(\mathbf{x}) - \frac{\mu(n+2)}{2n \left(\operatorname{Tr}[L_1 \Sigma] \sigma(\mathbf{x}) + 2 \right)} \|\nabla f(\mathbf{x})\|^2_{L_1^{-1}} . \quad (16)$$

where

$$\sigma(\mathbf{x}) := \frac{\|\nabla f(\mathbf{x})\|^2_{(L_1 \Sigma L_1)^{-1}}}{\|\nabla f(\mathbf{x})\|^2_{L_1^{-1}}} . \quad (17)$$

Proof Lemma 6 with $\mathbf{z} = \mathbf{x} - L_1^{-1}\nabla f(\mathbf{x})$ yields

$$\mathbb{E}_{\mathbf{u}} \left[f(\mathbf{x}_+) \mid \mathbf{x} \right] \leq f(\mathbf{x}) - \frac{h\mu}{n} \|\nabla f(\mathbf{x})\|^2_{L_1^{-1}} + \frac{h^2 \mu}{2n(n+2)} \left(\operatorname{Tr}[L_1 \Sigma] \|\nabla f(\mathbf{x})\|^2_{(L_1 \Sigma L_1)^{-1}} + 2 \|\nabla f(\mathbf{x})\|^2_{L_1^{-1}} \right) . \quad (18)$$

The best choice of the parameter h is obviously the one that minimizes the right hand size of this inequality. Taking the derivative with respect to h, we see that h must satisfy

$$-(n+2) \|\nabla f(\mathbf{x})\|^2_{L_1^{-1}} + h \left(\operatorname{Tr}[L_1 \Sigma] \|\nabla f(\mathbf{x})\|^2_{(L_1 \Sigma L_1)^{-1}} + 2 \|\nabla f(\mathbf{x})\|^2_{L_1^{-1}} \right) = 0 .$$

With this choice of h we obtain our final bound on the single step progress. □

This lemma shows that, on average, there is progress in every single step if $\|\nabla f(\mathbf{x})\|_{L_1^{-1}}$ is bounded away from zero.[2] This can be assured for all strongly convex functions, but not for convex functions in general. This leads us to the next section where we will use the just derived lemmas to prove global convergence.

[2] Here we use that $\operatorname{Tr}[L_1 \Sigma] > 0$; see the remark after Theorem 2.

4.2 Global convergence

We now use the previously derived bounds on the expected single step progress (Lemma 7 and 8) to show convergence of F-RP in expectation. We first show convergence on smooth but not necessarily strongly convex functions.

Theorem 1 *Let $f \in C^1_{L_1}$, let $\mathbf{x}^* \in \mathbb{R}^n$ be a minimizer of f and let the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ be generated by Algorithm F-RP with covariance $\Sigma \in \mathrm{PD}_n$ and line search with accuracy $0 < \mu \leq 1$. Assume there exists $R \in \mathbb{R}$, s.t. $\|\mathbf{y} - \mathbf{x}_0\|_{L_1} \leq R$ for all $\mathbf{y} \in \mathbb{R}^n$ with $f(\mathbf{y}) \leq f(\mathbf{x}_0)$. Then, for any $N \geq 0$, we have*

$$\mathbb{E}\left[f(\mathbf{x}_N) - f(\mathbf{x}^*)\right] \leq \frac{Q}{N+1}\,, \tag{19}$$

where

$$Q := \max\left\{\frac{2\omega(L_1,\Sigma)R^2 n}{\mu(n+2)}, f(\mathbf{x}_0) - f(\mathbf{x}^*)\right\}\,, \quad \omega(L_1,\Sigma) := \left(\frac{\mathrm{Tr}\left[L_1\Sigma\right]}{\lambda_{\min}(\Sigma L_1)} + 2\right)\,.$$

Proof By assumption, $\|\mathbf{x}_k - \mathbf{x}^*\|_{L_1} \leq R$ for all $k = 0, 1, \ldots, N$. With Lemma 7 it follows for any step size $h_k \geq 0$:

$$\mathbb{E}\left[f(\mathbf{x}_{k+1}) - f(\mathbf{x}^*) \mid \mathbf{x}_k\right] \leq \left(1 - \frac{h_k\mu}{n}\right)\left(f(\mathbf{x}_k) - f(\mathbf{x}^*)\right) + h_k^2\frac{R^2\omega(L_1,\Sigma)\mu}{2n(n+2)}\,.$$

Taking expectations over $\mathbf{x}_k$, we obtain

$$\mathbb{E}\left[f(\mathbf{x}_{k+1}) - f(\mathbf{x}^*)\right] \leq \left(1 - \frac{h_k\mu}{n}\right)\mathbb{E}\left[f(\mathbf{x}_k) - f(\mathbf{x}^*)\right] + h_k^2\frac{R^2\omega(L_1,\Sigma)\mu}{2n(n+2)}\,.$$

As in [29, Theorem 5.3], the choice $h_k := \frac{2n}{\mu(k+1)}$ for $k = 0, 1, \ldots, (N-1)$ yields the lemma. □

On strongly convex functions the convergence of F-RP is linear.

Theorem 2 *Let $f \in C^1_{L_1}$ and let f in addition be strongly convex with parameter $M \in \mathrm{PD}_n$. Let $\mathbf{x}^* \in \mathbb{R}^n$ denote the unique minimizer of f, and let the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ be generated by Algorithm F-RP with covariance $\Sigma \in \mathrm{PD}_n$ and line search with accuracy $0 \leq \mu \leq 1$. Then*

$$\mathbb{E}\left[f(\mathbf{x}_N) - f(\mathbf{x}^*)\right] \leq \hat{\varrho}^N \cdot \left(f(\mathbf{x}_0) - f(\mathbf{x}^*)\right)\,, \tag{20}$$

where

$$\hat{\varrho} := 1 - \frac{\lambda_{\min}(ML_1^{-1})\mu(n+2)}{n\left(\mathrm{Tr}[L_1\Sigma]\lambda_{\max}(\Sigma^{-1}L_1^{-1}) + 2\right)} \tag{21}$$

is the convergence factor *or* convergence rate*.*

Proof We use Lemma 1 to establish

$$\|\nabla f(\mathbf{x}_k)\|^2_{L_1^{-1}} \geq \lambda_{\min}(ML_1^{-1})\,\|\nabla f(\mathbf{x}_k)\|^2_{M^{-1}}\,.$$

Applying the quadratic lower bound (10) to further bound the latter term from below yields

$$\|\nabla f(\mathbf{x}_k)\|^2_{L_1^{-1}} \geq 2\lambda_{\min}(ML_1^{-1})\left(f(\mathbf{x}_k) - f(\mathbf{x}^*)\right)\,.$$

Now we can combine this bound with Lemma 8 and get

$$\mathbb{E}_{\mathbf{u}}\left[f(\mathbf{x}_k) - f(\mathbf{x}^*) \mid \mathbf{x}_k\right] \leq \varrho(\mathbf{x}_k) \cdot \left(f(\mathbf{x}_k) - f(\mathbf{x}^*)\right)\,, \tag{22}$$

where

$$\varrho(\mathbf{x}_k) := 1 - \frac{\lambda_{\min}(ML_1^{-1})\mu(n+2)}{n\left(\mathrm{Tr}[L_1\Sigma]\sigma(\mathbf{x}_k)+2\right)}\,, \tag{23}$$

is the exact convergence factor. With Lemma 1 we find a uniform upper bound on $\sigma(\mathbf{x}_k)$:

$$\sigma(\mathbf{x}_k) = \frac{\|\nabla f(\mathbf{x})\|^2_{(L_1\Sigma L_1)^{-1}}}{\|\nabla f(\mathbf{x})\|^2_{L_1^{-1}}} \leq \lambda_{\max}(\Sigma^{-1}L_1^{-1})\,. \tag{24}$$

We can use this to uniformly (over all k's) bound $\varrho(\mathbf{x}_k)$:

$$\varrho(\mathbf{x}_k) \leq 1 - \frac{\lambda_{\min}(ML_1^{-1})\mu(n+2)}{n\left(\mathrm{Tr}[L_1\Sigma]\lambda_{\max}(\Sigma^{-1}L_1^{-1})+2\right)} = \hat{\varrho}\,.$$

The Theorem follows from Equation (22) by taking expectation over $\mathbf{x}_k$. □

We remark that the progress is strict: It follows from the decomposition $\Sigma = CC^T$, "Rotation" Lemma 2 in Section 3, and Sylvester's law of inertia that all three terms $\lambda_{\min}(ML_1^{-1})$,$\mathrm{Tr}[L_1\Sigma]$ and $\lambda_{\max}(L_1\Sigma)$ are positive.

It is not necessary that the function f is strongly convex everywhere for linear convergence to hold. Theorem 3 below shows that convergence (at about a quarter of the rate of the one in Theorem 2) can be proven assuming only a weaker condition. Let us recall that strong convexity with parameter M implies that

$$f(\mathbf{x}) - f(\mathbf{x}^*) \geq \frac{1}{2}\left\|\mathbf{x}-\mathbf{x}^*\right\|^2_M\,, \forall \mathbf{x}\in\mathbb{R}^n\,. \tag{25}$$

It turns out that, instead of strong convexity (9), the weaker condition (25) is enough for linear convergence.

Theorem 3 *Let $f \in C^1_{L_1}$ and let f in addition have a unique minimizer $\mathbf{x}^* \in \mathbb{R}^n$ satisfying (25) with $M \in \mathrm{PD}_n$. Let the sequence $\{\mathbf{x}_k\}_{k\geq 0}$ be generated by Algorithm F-RP with covariance $\Sigma \in \mathrm{PD}_n$ and line search with accuracy $0 \leq \mu \leq 1$. Then*

$$\mathbb{E}\left[f(\mathbf{x}_N) - f(\mathbf{x}^*)\right] \leq \left(1 - \frac{\mu(n+2)}{4n\theta}\right)^N \cdot \left(f(\mathbf{x}_0) - f(\mathbf{x}^*)\right)\,, \tag{26}$$

where

$$\theta = \left(\frac{\mathrm{Tr}\left[L_1\Sigma\right]}{\lambda_{\min}(M\Sigma)} + \frac{2}{\lambda_{\min}(ML_1^{-1})}\right)\,.$$

Proof To see this, we use (14) to get

$$\begin{aligned}\mathbb{E}\left[f(\mathbf{x}_+) - f(\mathbf{x}^*) \mid \mathbf{x}\right] &\leq \left(1 - \frac{h\mu}{n}\right)\left(f(\mathbf{x}) - f(\mathbf{x}^*)\right) \\ &\quad + \frac{h^2\mu}{2n(n+2)}\left(\mathrm{Tr}\left[L_1\Sigma\right]\left\|\mathbf{x}-\mathbf{x}^*\right\|^2_{\Sigma^{-1}} + 2\left\|\mathbf{x}-\mathbf{x}^*\right\|^2_{L_1}\right) \\ &\leq \left(1 - \frac{h\mu}{n} + \frac{h^2\mu\theta}{n(n+2)}\right)\left(f(\mathbf{x}) - f(\mathbf{x}^*)\right)\,,\end{aligned}$$

where we used Lemma 1 to bound

$$\left\|\mathbf{x}-\mathbf{x}^*\right\|^2_{\Sigma^{-1}} \leq \left\|\mathbf{x}-\mathbf{x}^*\right\|^2_M \cdot \frac{1}{\lambda_{\min}(M\Sigma)}$$

Function	Previous result [29]	New result	Estimation
f_1	$1-\frac{1}{\ell n}$	$1-\frac{(n+2)}{n(n\ell/2+2)}$	$\approx 1-\frac{(n+2)}{n(n\ell/4+n/4+2)}$
f_2	$1-\frac{1}{\ell n}$	$1-\frac{(n+2)}{n(n\ell/2+2)}$	$\approx 1-\frac{(n+2)}{n(\ell/2+n+1/2)}$

Table 1 Theoretical convergence rates of F-RP on functions f_1 and f_2 from previous analysis [29], present analysis, and a heuristic argument (as explained in the main text).

and

$$\|\mathbf{x}-\mathbf{x}^*\|_{L_1}^2 \leq \|\mathbf{x}-\mathbf{x}^*\|_M^2 \cdot \frac{1}{\lambda_{\min}(ML_1^{-1})}\,,$$

followed by (25).

Setting h to $\frac{1}{2\theta}$ the term in the left bracket becomes $\left(1-\frac{\mu(n+2)}{4n\theta}\right)$ and the proof continues as the proof of Theorem 2. □

Remark The presented theoretical results extend our previous work [29] in two ways: (i) the analysis in [29] considered only F-RP with covariance $\Sigma = I_n$ the n-dimensional identity matrix with less expressive quadratic lower and upper bound assumptions; (ii) the lower- and upper bounds introduced in Section 3.3 allow for a more detailed description of the convergence rates because the quadratic model captures the eigenspectra of the functions.

4.3 Numerical illustration

To demonstrate the tightness of the bounds in practice we consider the convergence behavior of F-RP on two quadratic functions with parameter $\ell \geq 1$.

$$f_1(\mathbf{x}) = \frac{1}{2}\left(\sum_{i=1}^{\lceil \frac{n}{2} \rceil} x_i^2 + \ell \sum_{i=\lfloor \frac{n}{2} \rfloor}^{n} x_i^2\right)\,, \qquad f_2(\mathbf{x}) = \frac{1}{2}\left(x_1^2 + \frac{\ell}{2}\sum_{i=2}^{n-1} x_i^2 + \ell x_n^2\right)\,,$$

The Hessian matrices in both functions have the same maximal (ℓ) and minimal (1) eigenvalues. The function f_1 has two different scales that are distributed evenly among the dimensions. The second function f_2 has – for large dimension – one global scale with one small and one large eigenvalue. A previous numerical study [27] suggests that function f_1 is challenging for RP algorithms and f_2 is easy among all convex quadratic functions with the same condition number and trace.

The algorithm F-RP with covariance $\Sigma = I_n$ converges on both functions where the convergence rate is described by the convergence factor (cf. Theorem 2). For f_1 the result established in [29] provides the upper bound $\left(1-\frac{1}{\ell n}\right)$ on the convergence factor. The new results provided in Theorem 2 yields a (better) bound of roughly $\left(1-\frac{2}{\ell n}\right)$ (see Table 4.3 for the exact factor). Unfortunately, the new results from Theorem 2 can still not distinguish between f_1 and f_2 (it provides the same upper bound on the convergence factor). Figure 4.3 shows the numerically observed convergence rates of F-RP (with exact line search) for functions f_1 and f_2 with $\ell = 1000$ in dimension $n = 20$. We observe that (i) F-RP converges faster on f_2 than on f_1 and (ii) the new estimate of the convergence factor is more realistic for f_1 (the challenging quadratic function) than for f_2 (red lines).

It is clear that our worst-case analysis cannot give accurate convergence rates on all convex quadratic functions. We can, nonetheless, explain the observed difference by having a fresh look at the proof of Theorem 2 and using the following heuristic argument. In Equation (24) we used a crude (yet tight in the worst case) estimation of $\sigma(\mathbf{x})$. This estimate is not tight in every step. In order to find a convergence

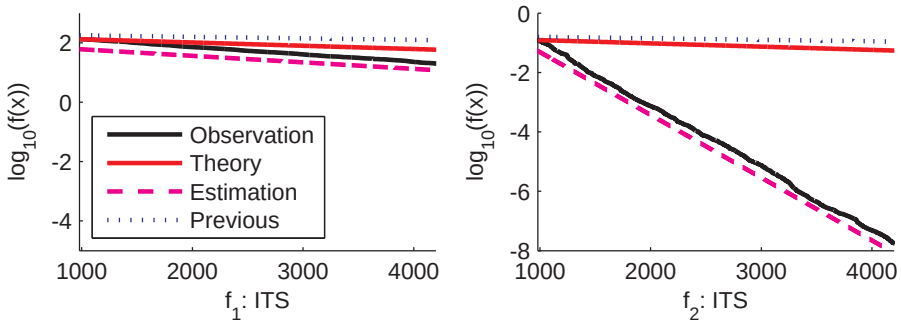

Fig. 2 Comparison of the derived convergence rates (see Table 4.3) with empirical measurements on f_1 (left) and f_2 (right). Logarithm of function value f_i vs. number of iterations (ITS). Observed $F-RP$ convergence (solid/black), convergence rate derived in [29] (dotted), convergence rate derived in this paper (solid/red), and heuristically estimated average convergence rate (dashed).

factor that best matches the observed rates we may rather analyze an *average case* scenario. Although such an analysis is beyond the scope of this manuscript we present the following heuristic argument. Let us assume that for a function f and $L_1 \in \mathrm{PD}_n$, the sequence of gradient directions $\{\frac{\nabla f(\mathbf{x}_k)}{\|\nabla f(\mathbf{x}_k)\|_{L_1^{-1}}}\}$ at the iterates $\{\mathbf{x}_k\}$ of F-RP follows "roughly" a normalized normal distribution with covariance L_1, i.e., its distribution is close to the distribution of $\mathbf{v} \sim \bar{\mathcal{N}}(0, L_1)$. Then

$$\mathbb{E}_{\nabla f(\mathbf{x})}\left[\frac{\|\nabla f(\mathbf{x})\|^2_{(L_1 \Sigma L_1)^{-1}}}{\|\nabla f(\mathbf{x})\|^2_{L_1^{-1}}}\right] = E_{\mathbf{v}}\left[\mathbf{v}^T L_1^{-1} \mathbf{v}\right] = \frac{\mathrm{Tr}[L_1^{-1}]}{n},$$

with Lemma 4. Intuitively, the above assumption is a good guess in case of quadratic functions (as f_1 and f_2 both are). Replacing the upper bound on $\sigma(\mathbf{x}_k)$ in (24) by its heuristic estimation $\mathrm{Tr}[L_1^{-1}]/n$ yields new convergence factors for both functions f_1 and f_2 as tabluated in the last column of Table 4.3. We further see in Figure 4.3 that the new factors using this "average" parameter σ well match the empirically observed results (dashed lines). An in-depth analysis of this striking observation is subject of further research.

5 Metric Learning in Random Pursuit

Our bounds on the progress rate of F-RP with fixed covariance matrix Σ describe the influence of the matrix upper and lower bounds and the chosen covariance matrix on the convergence rate of RP algorithms. For instance, for the special case of quadratic functions of the form $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H \mathbf{x}$ with $H \in \mathrm{PD}_n$ and trivial quadratic upper and lower bound $L_1 = M = H$, we have seen in the previous section that the exact convergence factor ϱ (23) concisely describes the empirically observed convergence rate.

The choice $\Sigma = H^{-1}$ of the covariance matrix leads to the expected progress rate $\hat{\varrho} = \left(1 - \frac{1}{n}\right)$ (cf. Thm 2). This rate is (i) independent of the function f (i.e., the spectrum of H) and (ii) optimal from a theoretical point of view. This follows from the fact that $\left(1 - \frac{1}{n}\right)$ is a lower bound on the convergence rate of the "hit-and-run" search algorithm as shown by Jägersküpper in [12]. The idealized "hit-and-run" scheme analyzed there is identical to the Random Pursuit.

For general strongly convex functions $f\colon \mathbb{R}^n \to \mathbb{R}$, the Hessian $\nabla^2 f(\mathbf{x})$ is not constant for all $\mathbf{x} \in \mathbb{R}^n$. However, if we assume that the Hessian is only mildly changing (for instance, Lipschitz continuous) then for all $\mathbf{x} \in \mathbb{R}^n$ that are close

to the unique minimizer $\mathbf{x}^* \in \mathbb{R}^n$ the corresponding Hessians will also be close, meaning that $\nabla^2 f(\mathbf{x}) \approx \nabla^2 f(\mathbf{x}^*)$ in some norm. The choice $\Sigma^{-1} = \nabla^2 f(\mathbf{x}^*)$ is thus likely to also yield a good convergence rate on this function class.

We are now left with the challenge of how to efficiently learn a suitable covariance matrix (that induces the right metric) on smooth convex functions in the present gradient-free setting. Iterative stochastic covariance matrix adaptation schemes are well-established in gradient-free continuous optimization [7,14,17] but notoriously difficult to study theoretically. A welcome alternative has recently been introduced by Leventhal and Lewis [15] in form of a Randomized Hessian approximation scheme. We here review and extend their scheme which we refer to as Variable Metric (VM) update. For twice differentiable functions $f\colon \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^n$ and initial iterate $B_0 \in \mathrm{PD}_n$, Leventhal and Lewis already showed that the VM scheme generates a random sequence $\{B_k\}_{k\in\mathbb{N}}$ of iterates that converge to $\nabla^2 f(\mathbf{x})$. For quadratic functions, their analysis also reveals exact rates for the convergence in expectation of the sequence of estimates $\{B_k\}_{k\in\mathbb{N}}$ to the true Hessian H. We are, however, interested in the inverses of these matrices, since we want to understand convergence of the covariances $\Sigma = B_k^{-1}, k \in \mathbb{N}$ to the optimal covariance H^{-1}. We know that for a sufficiently large number K of steps, $B_K^{-1} \approx H^{-1}$ almost surely, but explicit bounds for K do not directly follow from existing results.

We here address this question and provide, in addition, three novel, theoretically sound, and easy-to-implement VM updates. For many practical situations, function evaluations are the most costly operations, and the goal is to keep the number of evaluations as low as possible. The third proposed scheme allows – at the expense of slightly more memory storage and computational cost – to significantly boost the performance of the VM update in this scenario. We illustrate the numerical performance of these schemes on a model quadratic function (see Section 5.5). For simplicity, we analyze VM updates and their interplay with Random Pursuit solely on quadratic functions. The general convex case is subject of future research.

5.1 Variable Metric update schemes

The generic VM update [15] comprises direct updates of a Hessian estimate. Given a symmetric matrix $B \in \mathrm{PD}_n$ as current Hessian estimate, a direction $\mathbf{u}$ is selected uniformly at random from the $(n-1)$-dimensional hypersphere S^{n-1} (i.e., $\mathbf{u} \sim \bar{\mathcal{N}}(0, I_n)$). The next iterate B_+ is determined according to:

$$B_+ = B + \mathbf{u}^T (H - B)\, \mathbf{u} \cdot \mathbf{u}\mathbf{u}^T . \tag{27}$$

This formula requires the evaluation of $\mathbf{u}^T H \mathbf{u}$ with unknown H. For twice differentiable functions f the second derivative of f at $\mathbf{x}$ in direction $\mathbf{u}$ can be well approximated by finite differences[3]:

$$\mathbf{u}^T H \mathbf{u} \approx \frac{f(\mathbf{x} + \epsilon\mathbf{u}) - 2f(\mathbf{x}) + f(\mathbf{x} - \epsilon\mathbf{u})}{\epsilon^2} \tag{28}$$

for some small $\epsilon > 0$. In the convex quadratic case, the above formula is exact for arbitrary $\epsilon > 0$. For general functions that do not meet the above requirements, the approximation of $\mathbf{u}^T H \mathbf{u}$ with formula (28) may not be accurate, thus leading to a failure of the update.

Note that formula (28) only requires two additional function evaluations at $\mathbf{x} \pm \epsilon\mathbf{u}$. In addition, the formula implies that the estimate B_+ behaves at $\mathbf{x}$ like the unknown Hessian along direction $\mathbf{u}$, that is, $\mathbf{u}^T B_+ \mathbf{u} = \mathbf{u}^T H \mathbf{u}$. This can be seen

[3] The two points $\mathbf{x} \pm \epsilon\mathbf{u}$ are not required to be at the same distance to $\mathbf{x}$. For points $\mathbf{x} - \epsilon_1\mathbf{u}$, $\mathbf{x} + \epsilon_2\mathbf{u}$ the curvature can be estimated by quadratic interpolation with slight adaptation of formula (28).

directly from (27) by noting that $\mathbf{u}^T\mathbf{u}\mathbf{u}^T\mathbf{u} = 1$. Unfortunately, the update does not guarantee that the matrix B_+ stays positive definite. An standard result in Wedderburn [30, pg. 69] states that for $B \in \mathrm{PD}_n$, $\mathbf{u} \in \mathbb{R}^n$ with $\|\mathbf{u}\| = 1$, the matrix $B + t\mathbf{u}\mathbf{u}^T$ is positive definite if $t^{-1} < \mathbf{u}^T B^{-1}\mathbf{u}$.

In order for the matrix B_+ to be useful in Variable Metric Random Pursuit, an additional correction step is required if B_+ is not positive definite. Leventhal and Lewis suggest an *ad hoc* projection of B_+ onto the cone of PD_n matrices. They numerically show that this yields a practicable algorithm [15]. We here introduce two alternatives, `updateHess` and `updateHessCorr`, as outlined in Fig. 3.

`updateHess`$(f, \mathbf{x}, B, \epsilon)$

Requires: Global variable T, initialized at first invocation to $T = B$
Output : Hessian estimate $B_+ \in \mathrm{PD}_n$
1 $\mathbf{u} \sim \mathcal{N}(0, I_n)$
2 $\Delta_\mathbf{u} \leftarrow \frac{f(\mathbf{x}+\epsilon\mathbf{u})-2f(\mathbf{x})+f(\mathbf{x}-\epsilon\mathbf{u})}{\epsilon^2} - \mathbf{u}^T B\mathbf{u}$
3 **if** $T \leftarrow T + \Delta_\mathbf{u} \cdot \mathbf{u}\mathbf{u}^T \in \mathrm{PD}_n$ **then**
4 $\quad B_+ \leftarrow T$
else
5 $\quad B_+ \leftarrow B$
6 **return** B_+

`updateHessCorr`$(f, \mathbf{x}, B, \epsilon)$

Output: Hessian estimate $B_+ \in \mathrm{PD}_n$
1 $\mathbf{u} \sim \mathcal{N}(0, I_n)$
2 $\Delta_\mathbf{u} \leftarrow \frac{f(\mathbf{x}+\epsilon\mathbf{u})-2f(\mathbf{x})+f(\mathbf{x}-\epsilon\mathbf{u})}{\epsilon^2} - \mathbf{u}^T B\mathbf{u}$
3 **if** $T \leftarrow B + \Delta_\mathbf{u} \cdot \mathbf{u}\mathbf{u}^T \in \mathrm{PD}_n$ **then**
4 $\quad B_+ \leftarrow T$
else
5 $\quad \mathbf{v} \leftarrow$ `smallestEVec`(T)
6 $\quad \Delta_\mathbf{v} \leftarrow \frac{f(\mathbf{x}+\epsilon\mathbf{v})-2f(\mathbf{x})+f(\mathbf{x}-\epsilon\mathbf{v})}{\epsilon^2} - \mathbf{v}^T T\mathbf{v}$
7 $\quad B_+ \leftarrow \left(B + \Delta_\mathbf{v} \cdot \mathbf{v}\mathbf{v}^T\right) + \Delta_\mathbf{u} \cdot \mathbf{u}\mathbf{u}^T$
8 **return** B_+

Fig. 3 Two implementations of the VM update scheme (27). Left panel: The update (27) is applied to a temporary matrix T and the matrix B is only updated if T is positive definite. Right panel: The matrix B is updated in every step. Positive definiteness is established by an additional correction step (see main text for further information).

In sub-routine `updateHess` the current matrix T is returned if it is positive definite. Otherwise, the *last known* positive definite matrix is used. As long as the iterates are positive definite, no additional computational effort is needed. The update can be implemented in $O(n^2)$ by using a rank-one update on the Cholesky decomposition of T. However, if T is not positive semidefinite this approach fails, and the condition on line 3 of the algorithm must be checked by computation of the smallest eigenvalue.

In sub-routine `updateHessCorr` we ensure that the generated iterates are always positive definite. In case B_+ is not positive definite (checked by Wedderburn's formula), we apply a second VM update step in direction $\mathbf{v}$, where $\mathbf{v}$ is an eigenvector of B_+ corresponding to the smallest (hence negative) eigenvalue of B_+. By standard matrix perturbation theory, as detailed in Lemma 9 below, the twice updated matrix will be positive definite again (as H is). This scheme comes at the expense of two additional function evaluations at $\mathbf{x} \pm \epsilon\mathbf{v}$. This version of the VM update has already been successfully used in a recent numerical study [27].

Lemma 9 *Let $A \in \mathrm{PD}_n$, $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z}_1 \in \mathbb{R}^n$ an eigenvector corresponding to the smallest eigenvalue of $(A - \mathbf{x}\mathbf{x}^T)$. Then*

$$B := A - \mathbf{x}\mathbf{x}^T + \left|\lambda_{\min}(A - \mathbf{x}\mathbf{x}^T)\right| \mathbf{z}_1\mathbf{z}_1^T \in \mathrm{PD}_n\,.$$

Proof The matrix $(A - \mathbf{x}\mathbf{x}^T)$ is symmetric. Let $(A - \mathbf{x}\mathbf{x}^T) = \sum_{i=1}^n \lambda_i \mathbf{z}_i\mathbf{z}_i^T$ denote its spectral decomposition with $\lambda_1 \le \lambda_2 \le \ldots \lambda_n$ in increasing order. If $\lambda_1 \ge 0$, then there is nothing to show. Otherwise, we observe that by a variant of Weyl's theorem (cf. [9, Theorem 4.3.4]), $0 \le \lambda_i(A) \le \lambda_{i+1}(A - \mathbf{x}\mathbf{x}^T) = \lambda_{i+1}$ for $i = 1, \ldots, n-1$. Thus at most λ_1 can be negative. We conclude

$$\mathbf{y}^T B\mathbf{y} = \mathbf{y}^T \left(\sum_{i=1}^n \lambda_i \mathbf{z}_i\mathbf{z}_i^T + |\lambda_1|\, \mathbf{z}_1\mathbf{z}_1^T\right) \mathbf{y} \ge \mathbf{y}^T \left(\lambda_1 \mathbf{z}_1\mathbf{z}_1^T + |\lambda_1|\, \mathbf{z}_1\mathbf{z}_1^T\right) \mathbf{y} \ge 0\,,$$

for all $\mathbf{y} \in \mathbb{R}^n$. □

The present VM update schemes can be used in several ways in combination with Random Pursuit: (i) One can use the VM scheme at the initial iterate $\mathbf{x}_0 \in \mathbb{R}^n$ multiple times in order to get a good approximation B of the Hessian $\nabla^2 f(\mathbf{x}_0)$ and then employ $\Sigma = B^{-1}$ in F-RP; (ii) one could continuously toggle between VM update steps and line searches (as shown in Fig. 1, right panel), yielding a fully adaptive scheme. The analysis of the latter combination is naturally more evolved. On quadratic functions, however, the VM update is independent of the current position $\mathbf{x} \in \mathbb{R}^n$ and thus, V-RP is amenable to a theoretical analysis. Finally, note also that the update schemes are easily parallelizable due to the independence of the directions $\mathbf{u}$ and the calculation of $\mathbf{u}H\mathbf{u}$. This can be used on today's multi-core machines to speed-up the algorithm.

5.2 Accelerating the VM scheme

The VM schemes discussed so far need in every iteration at least two function evaluations to perform the update. In settings where function evaluations are costly or time consuming an alternative to parallelizing the estimation H is to reuse/recycle previously computed function values. We here propose two variants. Firstly, a number of function evaluations are necessary in V-RP to perform the line search (along a line defined by a vector $\mathbf{v}$). If the number of known function values on this line is at least 3, then $\mathbf{v}^T H \mathbf{v}$ can be estimated and an additional VM update (at no extra cost) can be performed. Secondly, we may also reuse previously computed curvature estimates. If we assume that the Hessian $H(\mathbf{x}_k)$ is only mildly changing between successive iterates $\mathbf{x}_k$ of V-RP, then the previously computed values $\mathbf{u}_{k-t}^T H(\mathbf{x}_{k-t})\mathbf{u}_{k-t}$ back to some horizon h, $t = 1, \ldots, h$, might still be accurate estimates of the curvature in direction $\mathbf{u}_{k-t}$ at the current position $\mathbf{x}_k$. Thus, one might apply the update (27) again for directions $\mathbf{u}_{k-t}$ using the approximation $\mathbf{u}_{k-t}^T H(\mathbf{x}_k)\mathbf{u}_{k-t} \approx \mathbf{u}_{k-t}^T H(\mathbf{x}_{k-t})\mathbf{u}_{k-t}$. This requires additional computation time but no additional function evaluations. Note that the length of the horizon is a free parameter that should reflect our believe about how strongly the function's curvature changes.

The two practical improvements of the VM scheme are summarized in Fig. 4 with horizon $h = n^2$. The updates over the whole horizon can be triggered at suitable intervals (e.g., every n-th iteration).

`updateHessStore`$(f, \mathbf{x}, B, \epsilon, \text{reuse})$

Requires: Persistent storage S of size n^2; S stores the n^2 most recently added elements. If the capacity of S is exceeded, the oldest element is deleted.
Output : Hessian estimate $B_+ \in \mathrm{PD}_n$

```
1 B_+ ← updateHess{Corr}(f, x, B, ε)
2 Add (u, u^T H u) to S, possibly also (v, v^T H v)
3 if reuse then
4     repeat m times
5         foreach (s, s) ∈ S do
6             if T ← B_+ + (s − s^T B_+ s) · ss^T ∈ PD_n then
7                 B_+ ← T
8 return B_+
```

Fig. 4 Implementation of the accelerated VM scheme. The pairs $(\mathbf{u}, \mathbf{u}^T H \mathbf{u})$ and $(\mathbf{v}, \mathbf{v}^T H \mathbf{v})$ in line 2 refer to the used quantities in the respective function used in line 1. In line 5 the elements of S can be processed in any order.

5.3 Convergence of the VM update schemes

We now show that the sequential VM updates indeed yields a sequence of matrices that converge to the Hessian. We restrict our theoretical investigation in this and the next section to quadratic functions $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x}$. The results can be extended to settings where the Hessian is mildly changing (say Lipschitz continuous) such than the errors made in the estimation (28) can be controlled.

To simplify the notation, we write the VM update in terms of the *error matrix* $E = (B - H)$.

Definition 5 (VM update scheme) Let $E_0 \in \mathbb{R}^{n\times n}$ be a symmetric matrix and $\{\mathbf{u}_k\}_{k\geq 0}$ a sequence of independent vectors sampled uniformly from $\tilde{\mathcal{N}}(0, I_n)$. Then the sequence $\{E_k\}_{k>0}$ is generated by the VM update if the iterates satisfy

$$E_{k+1} = E_k - \mathbf{u}_k^T E_k \mathbf{u}_k \cdot \mathbf{u}_k \mathbf{u}_k^T \,. \tag{29}$$

We observe that this definition matches with (27) for $E_k = B_k - H$.

The following Lemma summarizes the most important properties of the VM update scheme.

Lemma 10 *Let $\{E_k\}_{k\geq 0}$ be generated by the VM update* (29) *with E_0 symmetric and let $K = n(n+2)\ln(a\sqrt{b})$ for parameters $a \geq 1$, $b > 1$. Then*

(i) $\|E_k\|_F \leq \|E_{k-1}\|_F$, *for* $k \geq 1$,

(ii) $\mathbb{E}\left[\|E_k\|_F^2\right] \leq \left(1 - \frac{2}{n(n+2)}\right)^k \|E_0\|_F^2$ *for* $k \geq 0$, *and*

(iii) $\|E_k\|_F \leq \frac{\|E_0\|_F}{a}$ *for all* $k \geq K$, *with probability at least* $1 - \frac{1}{b}$.

Corollary 1 *Let $\{B'_k\}_{k\geq 0}$ with B'_0 symmetric a sequence of iterates generated either by the VM update as implemented in* `updateHessCorr` *or the sequence of internal matrices T in* `updateHess`. *Then the statement from Lemma 10 also holds for the sequence of error matrices $\{E'_k\}_{k\geq 0} := \{B'_k - H\}_{k\geq 0}$.*

Proof The internal matrices T in `updateHess` are exactly updated according to (29), hence nothing is to show. The iterates of `updateHessCorr` are almost generated according to (29). The additional correction step (if necessary) can be viewed as one step of the VM update (27) in a special (not random) direction. However, by (i) of Lemma 10 the Frobenius norm of the error matrix will not increase by this step. □

Proof (of Lemma 10) We note that (i) and (ii) were already proven by Leventhal and Lewis [15, Theorem 2.1]. It remains to show that (iii) follows from Markov's inequality. Since $\|E_k\|_F$ decreases monotonically, it suffices to prove (iii) for $k = K$. We have

$$\left(1 - \frac{2}{n(n+2)}\right)^K \leq \frac{1}{(a\sqrt{b})^2} \,,$$

hence, by (ii),

$$\mathbb{E}[\|E_K\|_F^2] \leq \frac{1}{b}\frac{\|E_0\|_F^2}{a^2} \,.$$

By the Markov inequality, the probability that $\|E_K\|_F^2$ exceeds its expectation by more than a factor of b is at most $1/b$, and this yields the lemma. □

For the accelerated scheme with `updateHessStore`, we have to show that applying the previous updates *again* is indeed sufficient for convergence. The main difficulty is that the vectors $\mathbf{u}_k$ are no longer normalized normally distributed but samples from a finite set.

Lemma 11 *Let U be a set of $h = \tilde{c}n^2 \log n$ normalized normal vectors $\{\mathbf{u}_i\}_{i=1}^h$, $\mathbf{u}_i \sim \bar{\mathcal{N}}(0, I_n)$ for $i = 1, \ldots, h$. Let $E \in \mathrm{PD}_n$ and let $E_+ = E - \mathbf{u}^T E \mathbf{u} \cdot \mathbf{u}\mathbf{u}^T$, the matrix obtained from one VM update* (27) *with vector $\mathbf{u}$. If $\mathbf{u}$ is sampled uniformly at random from the set U, it holds*

$$\mathbb{E}_{\mathbf{u}} \left[\|E_+\|_F^2 \right] \leq \left(1 - \frac{1}{4n^2}\right) \|E\|_F^2 \,,$$

with probability at least $\frac{1}{2}$. The parameter $\tilde{c}$ follows from Rudelson's Lemma (see below).

This Lemma shows, that the accelerated VM scheme as depicted in Fig. 4 indeed converges as soon as the size of the storage is large enough, $h = \Omega(n^2 \log n)$. For quadratic functions the scheme will converge to the true Hessian H given sufficiently many samples.

Proof (of Lemma 11) First we observe

$$\begin{aligned} \|E_+\|_F^2 &= \|E\|_F^2 - 2(\mathbf{u}_i^T E \mathbf{u}_i)^2 + (\mathbf{u}_i^T E \mathbf{u}_i)^2 \\ &= \|E\|_F^2 - (\mathbf{u}_i^T E \mathbf{u}_i)^2 \,, \end{aligned}$$

for all $i = 1, \ldots, h$, as a direct consequence of (29). Therefore

$$\begin{aligned} \mathbb{E}\left[\|E_+\|_F^2 \right] &= \|E\|_F^2 - \frac{1}{h} \sum_{i=1}^h (\mathbf{u}_i^T E \mathbf{u}_i)^2 \\ &\geq \|E\|_F^2 - \left(\frac{1}{h} \sum_{i=1}^h \mathbf{u}_i^T E \mathbf{u}_i \right)^2 = \|E\|_F^2 - \mathrm{Tr}\left[E \left(\frac{1}{h} \sum_{i=1}^h \mathbf{u}_i \mathbf{u}_i^T \right) \right]^2 \,, \end{aligned}$$

where the inequality is due to Jensen. Now we want to show that the empirical mean $\frac{1}{h} \sum_{i=1}^h \mathbf{u}_i \mathbf{u}_i^T$ is close to its expectation $\frac{1}{n} I_n$ (see Lemma 4). For this we use Rudelson's Lemma from [24, Thm 3.1.]. The statement of the Lemma is as follows: For a suitable constant c it holds

$$\mathbf{Pr}\left[\left\| \frac{1}{h} \sum_{i=1}^h \mathbf{u}_i \mathbf{u}_i^T - \frac{1}{n} I_n \right\|_2 > t \right] \leq 2 \exp\left[-ct^2 \frac{h}{\log h} \right] \,,$$

for every $0 < t < 1$. Therefore for the choice $h = \tilde{c} n^2 \log n$ the probability of failure

$$\mathbf{Pr}\left[\left\| \frac{1}{h} \sum_{i=1}^h \mathbf{u}_i \mathbf{u}_i^T - \frac{1}{n} I_n \right\|_2 > \frac{1}{2n} \right] \leq \frac{1}{2} \,,$$

where $\tilde{c}$ depends on the c from the original Lemma. Therefore, for this choice of h, we have

$$\|E\|_F^2 - \mathrm{Tr}\left[E \left(\frac{1}{h} \sum_{i=1}^h \mathbf{u}_i \mathbf{u}_i^T \right) \right]^2 \geq \|E\|_F^2 - \mathrm{Tr}\left[E \left(\frac{1}{2n} I_d \right) \right]^2 = \left(1 - \frac{1}{4n^2}\right) \|E\|_F^2 \,,$$

with probability at least $\frac{1}{2}$. □

5.4 Convergence of Variable Metric Random Pursuit

We now show that the combination of RP with sequential VM updates indeed yields a convergent algorithm on convex quadratic functions. This follows directly form the convergence results for the VM update.

The convergence rate of V-RP on strongly convex functions is described by the convergence factor $\hat{\varrho}$ from Theorem 2. As soon as the estimated Hessian B_k converges to the true Hessian H, $\hat{\varrho}$ converges to $\left(1-\frac{1}{n}\right)$, the optimal factor. We are interested in how quickly this the factor $\hat{\varrho}$ converges to the optimal one. Let us consider a quadratic function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x}$ with quadratic upper and lower bound $M = L_1 = H$. Then $\hat{\varrho}$ depends only on the spectrum of $H\Sigma$ with $\Sigma = B_k^{-1} = (H+E_k)^{-1}$.

Lemma 12 (VM convergence factor) *Let $H \in \mathrm{PD}_n$, let $\{E_k\}_{k\geq 0}$ with symmetric E_0 be a sequence of iterates generated by the VM update (29) on the function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x}$ and let $c < 1$. If for $k' \geq 0$:*

$$\|E_{k'}\|_2 \leq \frac{c}{\|H^{-1}\|_2}\,, \tag{30}$$

then the convergence factor $\hat{\varrho}$ from Theorem 2 with $\mu = 1$ can be upper bounded by

$$\hat{\varrho} \leq 1 - \frac{(1-c)^2}{n}\,. \tag{31}$$

Proof To show the Lemma, we derive bounds on the smallest and largest eigenvalues of the matrix $H\Sigma$, where $\Sigma = (H+E_{k'})^{-1}$. First we observe that

$$\max\left\{\left|\lambda_{\min}(E_{k'}H^{-1})\right|, \left|\lambda_{\max}(E_{k'}H^{-1})\right|\right\} = \left\|E_{k'}H^{-1}\right\|_2 \leq \|E_{k'}\|_2 \left\|H^{-1}\right\|_2$$

by the definition of the 2-norm and submultiplicativity. With the assumption on the product of the two norms, the right hand side can be upper bounded by c:

$$\|E_{k'}\|_2 \left\|H^{-1}\right\|_2 \leq c < 1\,.$$

Therefore, the largest eigenvalue of $H\Sigma$ is well defined and finite:

$$\begin{aligned}\lambda_{\max}(H\Sigma) &= \frac{1}{\lambda_{\min}((H\Sigma)^{-1})} = \frac{1}{\lambda_{\min}(I_n + E_{k'}H^{-1})} \\ &= \frac{1}{1+\lambda_{\min}(E_{k'}H^{-1})} \leq \frac{1}{1-c}\,.\end{aligned}$$

With a similar argumentation we obtain a lower bound on the smallest eigenvalue of $H\Sigma$:

$$\lambda_{\min}(H\Sigma) = \frac{1}{1+\lambda_{\max}(E_{k'}H^{-1})} \geq 1 - \lambda_{\max}(E_{k'}H^{-1}) \geq 1-c\,,$$

where the first inequality follows from $\frac{1}{1+x} \geq 1-x$ for $x > -1$. These two bounds together with the trivial estimate $\mathrm{Tr}[H\Sigma] \leq n\lambda_{\max}(H\Sigma)$ yield:

$$\begin{aligned}\frac{(n+2)\lambda_{\min}(H\Sigma)}{n(\mathrm{Tr}[H\Sigma] + 2\lambda_{\min}(H\Sigma)} &\geq \frac{(n+2)}{n(n\lambda_{\max}(H\Sigma) + 2\lambda_{\max}(H\Sigma))} \\ &\geq \frac{\lambda_{\min}(H\Sigma)}{n\lambda_{\max}(H\Sigma)} \geq \frac{(1-c)^2}{n}\,,\end{aligned}$$

and the claim follows. □

Now we have an upper bound on the convergence factor $\hat{\varrho}$ as soon as (31) holds, and we know an upper bound on the convergence rate of V-RP. We summarize the derived upper bound on $\hat{\varrho}$ in the following Corollary.

Corollary 2 *Let $H \in \mathrm{PD}_n$, let $\{B_k\}_{k\geq 0}$ with B_0 symmetric be a sequence of iterates generated by the VM update (27) on the function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T H\mathbf{x}$.*

If the VM update is implemented as `updateHess` *or* `updateHessCorr`*, then $\hat{\varrho} < 1$ for all $k \geq 0$ and with probability at least $1 - \frac{1}{b}$ it holds:*

$$\hat{\varrho} \leq 1 - \frac{(1 - \min\{1, c_k\})^2}{n}, \tag{32}$$

where $c_k = \left\|H^{-1}\right\|_2 \|E_0\|_F \sqrt{b} e^{-\frac{k}{n(n+2)}}$. Thus (32) is only non-trivial if $c_k < 1$.

If the update is implemented as `updateHessStore`*, then*

$$\hat{\varrho} \quad \begin{cases} < 1 & \text{for all } k \geq 0 \\ = 1 - \frac{1}{n} & \text{for } k \geq \tilde{c} n^2 \log n \end{cases}$$

where $\tilde{c}$ is such as in Lemma 11.

Proof The second part is due to Lemma 11. The first part we note that $\|E_k\|_2 \leq \|E_k\|_F$. With Lemma 10 (iii) for $a = \left\|H^{-1}\right\|_2 \|E_0\|_F$ we conclude that (31) holds with $c = c_k$ with probability $1 - \frac{1}{b}$. □

5.5 Illustrative numerical example

We now illustrate the typical convergence behavior of Variable Metric Random Pursuit on the challenging convex quadratic function f_1, introduced in Section 4.3. This function has two different scales that need to be learned. We use parameter $\ell = 10^7$. The ratio of largest to smallest eigenvalue of the Hessian (i.e. the condition number) is 10^7, and the global minimum of f_1 is at $\mathbf{x}^* = \mathbf{0}_n$ (where $\mathbf{0}_n$ is the all-zeros vector) with $f_1(\mathbf{x}^*) = 0$. We conduct 51 runs of V-RP in $n = 20$ dimensions. The initial conditions are $\mathbf{x}_0 = (1, \ldots, 1, 1/\sqrt{\ell}, \ldots, 1/\sqrt{\ell})^T$, $B_0 = \frac{1}{2}\ell \cdot I_n$. The two VM update schemes `updateHessCorr` (see Fig. 3) and `updateHessStore` (see Fig. 4) are tested with the setting $\epsilon = 1$ for both schemes. The `updateHessStore` scheme reuses samples from the storage S in every n-th iteration. We here report the evolution of the mean, maximum, and minimum function value vs. number of iterations (#ITS). We also calculate and report the evolution of the derived convergence factor $\hat{\varrho}$ from Thm. 2.

On quadratic functions, a typical V-RP optimization trajectory (see [15,27] for several examples) shows three distinct phases of convergence in function values: (i) a first short phase of rapid improvement, (ii) a metric learning phase with only marginal progress in function decrease, and (iii) a final rapid decrease in function value. In the present experiments we chose the initial iterate $\mathbf{x}_0$ such as to minimize the first phase. This allows a clearer quantification of the length of the adaptation phase. We see that the adaptation phase lasts for roughly $5n^2$ iterations in case of `updateHessStore` and 15-18n^2 iterations for `updateHessCorr`. We also visualize the derived upper bounds on the convergence factor (as summarized in Cor. 2) in the lower panel of Fig. 5. For `updateHessCorr` the curve is plotted using $b = 1$, and for `updateHessStore` using $\tilde{c} = 1.5$. We see that the shape of both curves resembles the observed data. However, in both cases the theoretical bounds overestimate the empirically observed curves ("shifted" to the right). In the upper panel of Fig. 5 we depict the theoretical derived upper bound on the function value (Thm. 2 with factor $\hat{\varrho}$ form Cor. 2). For `updateHessStore` the shape of this curve well matches

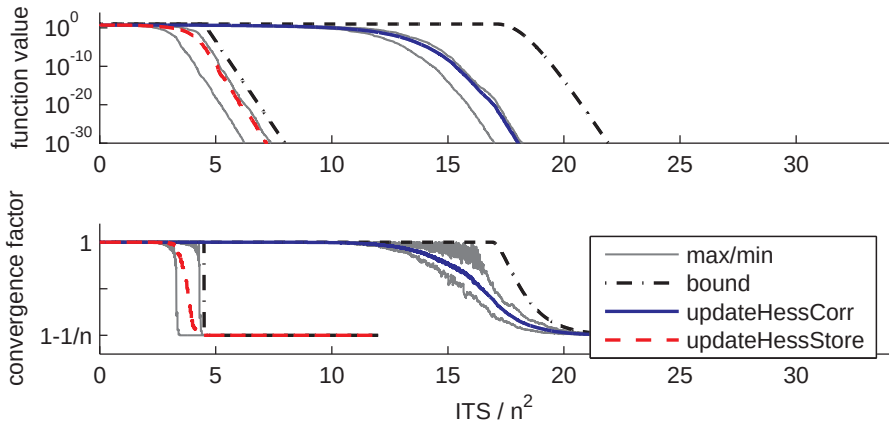

Fig. 5 Convergence of V-RP on f_1 for `updateHessCorr` (blue/solid) and `updateHessStore` (red/dashed). Parameter $\ell = 10^7$ in $n = 20$ dimensions. Upper panel: Mean and max/min (grey) function values vs. #ITS over 51 runs. Lower Panel: Mean and max/min (grey) convergence factor $\hat{\varrho}$ vs. #ITS over 51 runs. Respective upper bounds (black/dash-dotted). See main text for further information.

the observed convergence. For `updateHessCorr` we see that the empirically observed phase transition between phase (ii) and (iii) occurs more smoothly than predicted by the theoretical bound.

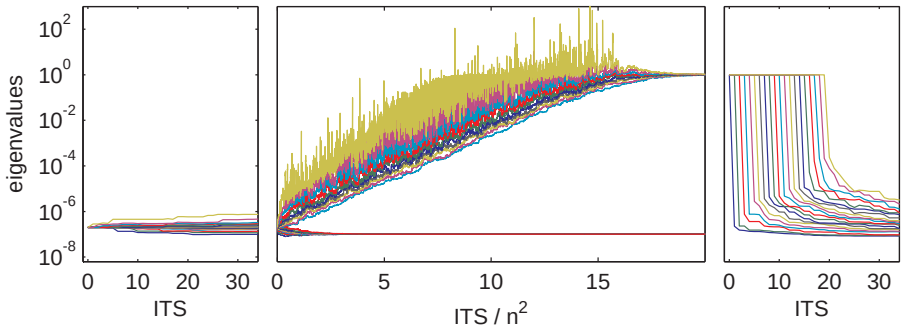

Fig. 6 Evolution of the spectrum of $\Sigma = B_k^{-1}$ for VM scheme `updateHessCorr` on f_1. Eigenvalues vs. # ITS for 1 run. Parameter $\ell = 10^7$ in $n = 20$ dimensions. Left two panels with intial setting $B_0 = \frac{\ell}{2} I_n$, right panel with $B_0 = I_n$. See main text for further information.

We also illustrate the evolution of the spectrum of the estimated inverse Hessian $\Sigma = B_k^{-1}$ in Fig. 6 for one run with update scheme `updateHessCorr`. At the beginning all eigenvalues are close to $\frac{2}{\ell}$, as $B_0^{-1} = \frac{2}{\ell} I_n$ (left panel). Then, about half of the eigenvalues start to increase up to 1, the other half decreases to $\frac{1}{\ell}$. We see that the large eigenvalues of Σ (or correspondingly the small eigenvalues of H) are more difficult to approximate. This takes up to 16-18n^2 iterations. In the right panel we depicted another run with initial matrix $\Sigma = B_0^{-1} = I_n$. At the beginning all eigenvalues are equal to 1. Due to the nature of the VM update scheme (rank one updates), at most one eigenvalue can become different from 1 in every iteration. Thus it takes exactly $n = 20$ iterations until all eigenvalues are between 10^{-7} and 10^{-5}. From this moment, the situation is similar to the experiment in the left two panels with $B_0 = \frac{\ell}{2} I_n$.

6 Computational Experiments

In [27] we have already presented extensive numerical results of V-RP in comparison with other randomized variable metric schemes. There, we analyzed the influence of the Hessian eigenvalue spectrum on the convergence of these schemes. The main result from these tests were that among a parametrized set of Hessian matrices with equal trace and condition number, the matrices with a sigmoidal spectrum are the most difficult to learn for the VM update scheme and matrices with an inverse sigmoidal (almost flat) distribution of eigenvalues are easier to learn. The functions f_1 and f_2 are limit cases of these function classes and can be considered as worst and best cases.

We here compare the performance of V-RP with a number of randomized and *deterministic* derivative-free algorithms. The set of test functions comprises three quadratic functions (including f_1 and f_2) with different spectra and one non-convex function. We first present the definition of the test functions and describe the numerical performance evaluation protocol. We then detail the algorithms and their parametrization.

6.1 Benchmark Functions

The first two functions are f_1 and f_2 (see definition in Section 4.3) with parameter $\ell = 10^7$. The benchmark set comprises a third quadratic function with yet another spectra. For parameter $\ell \geq 1$, we define

$$f_3(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{n} e^{1+(i-1)\frac{\log \ell - 1}{n-1}} x_i^2 \,.$$

In order to test the "valley-following" abilities of the different algorithms we also include the non-convex Rosenbrock [23] function f_4 in the benchmark set:

$$f_4(\mathbf{x}) = \sum_{i=1}^{n-1} \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right) \,.$$

The quadratic functions attain their minimum function value at $\mathbf{x}^* = \mathbf{0}_n$, the all zero vector ($f_1(\mathbf{0}_n) = f_2(\mathbf{0}_n) = f_3(\mathbf{0}_n) = 0$). The Rosenbrock function is minimized at $\mathbf{x}^* = \mathbf{1}_n$, with $f_4(\mathbf{1}_n) = 0$.

In order to prohibit the tested algorithms from making use of the diagonal structure of the Hessian matrices of f_1–f_3 we rotate the function domain by generating random rotation matrices R with $RR^T = I_n$ and a shift parameter $\mathbf{x}_s \sim \mathcal{N}(0, I_n)$, thus leading to function instances of the form

$$f(R(\mathbf{x} - \mathbf{x}_s)) \,.$$

We also apply the same transformation and shift to the initial iterate $\mathbf{x}_0$. This procedure and/or the special structure of $\mathbf{x}^*$. We use as initial iterate $\mathbf{x}_0 = \mathbf{1}_n$ for the quadratic functions and $\mathbf{x}_0 = \mathbf{0}_n$ for f_4.

6.2 Algorithms

6.2.1 V-RP schemes

We implemented F-RP with fixed covariance $\Sigma = I_n$. This scheme is simply referred to as RP. We also implemented two VM schemes presented in this paper,

`updateHessCorr` and `updateHessStore`. The parameter setting for the latter one can be found in Fig. 4 with $m = 10$. The choice of the parameter ϵ does not influence the performance of the schemes on the quadratic functions. We thus set it to $\epsilon = 1$. For f_4 we used $\epsilon = 10^{-6}$. For `updateHessStore` we apply the updates from the storage every n-th iteration in random order, starting after the n^2-th iteration (as soon as enough data is collected). All RP schemes have to be combined with an implementation of the line search oracle. We tested three different implementations and present them here in increasing order of function evaluations they consume.

ES: This scheme is also know as (1+1)-Evolution Strategy (ES). To perform the line search, only one additional point along the chosen line is probed. The scheme accepts the new point if the function value of the new point is lower than the function value of the previous iterate. In order to ensure that a positive fraction p is accepted on average we use an adaptive step size scheme (aSS) as detailed in the procedure `aSS` in Figure 1 from [27] with parameters $p = 0.27$ and $\sigma = 1$.

Exact: By probing two additional points on the given line (three in total), the exact minimizer can be computed if the function is quadratic (f_1–f_3). For f_4 this scheme may fail to report a better value but we observed in our experiments that the quality of the guessed minimizer is sufficient.

Matlab: We use the built-in MATLAB routine `fminunc.m` from the optimization toolbox [22] with `optimset('TolX'=`0.01`)` as approximate line search. In the present gradient-free setting `fminunc.m` uses a mixed cubic/quadratic polynomial line search where the first three points bracketing the minimum are found by bisection [22].

6.2.2 CMA-ES

The Evolution Strategy with Covariance Matrix Adaptation [7] (CMA-ES) is one of the most popular and efficient schemes for derivative free optimization on non-convex and noisy problems. New search points are sampled from a multivariate normal distribution whose parameter are updated in each iteration. The fundamental design principle used here is slightly different than for the V-RP schemes. Instead of performing the updates on an estimation of the Hessian (and then computing its inverse), the updates are performed directly on the inverse directly. The CMA-ES scheme is augmented by an auxiliary variable called evolution path that takes into account the correlation of successive means taken over a finite horizon. This is similar in spirit to Rao-Blackwellization techniques in Marko Chain Monte Carlo methods [1] and Polyak's heavy ball method in first-order optimization [20].

Among the many different instances of CMA-ES, we consider here the one that is the fastest scheme for quadratic functions known today. This scheme is called the (1,4)-CMA-ES with mirrored sampling and sequential selection. We also refer to [2] for a full description of this scheme and all parameter settings used. The scale parameter is set to $\sigma = 1$ for our experiments. The code for the (1,4)-CMA-ES scheme has been retrieved from `http://coco.gforge.inria.fr/doku.php?id=bbob-2010-results`.

6.2.3 Nesterov's Random Gradient schemes

Nesterov [19] introduced a derivative-free optimization scheme that is very similar to RP. Optimization is performed iteratively among randomly chosen lines. The optimal step size is estimated by finite-difference estimation. This scheme is called Random Gradient (RG) method. Its advantage over RP is that the finite-difference calculation needs only one additional function evaluation, and it is guaranteed to

make progress in every iteration (opposed to the ES line search). One disadvantage is that the RG method needs an estimate of the curvature of the function which is not available in practice. For test purposes, we always use the correct curvature of the objective function (parameter ℓ) as input to the RG scheme.

Similar to the accelerated gradient methods for convex optimization, an accelerated version of the RG scheme is available [19]. This scheme also needs only two function evaluation per iteration and shows superior theoretical convergence properties [19].

6.2.4 Pattern Search

Pattern Search is a deterministic scheme that evaluates the objective function in every iteration on $2n$ predefined points on a stencil. We use the built-in MATLAB routine `patternsearch` from the Global Optimization Toolbox [22] with parameters `Cache`=`on`, `InitialMeshSize`=1, `TolMesh`=1e-20, `TolX`=1e-20, `TolFun`=1e-20.

6.2.5 Nelder-Mead

We use the built-in MATLAB routine `fminsearch` from the Optimization Toolbox [22] which implements the classical Nelder-Mead (N-M) Down-Hill Simplex algorithm [18]. We use the algorithm with parameters `TolX`=1e-20, `TolFun`=1e-20.

6.2.6 NEWUOA

NEWUOA [21] is an iterative algorithm that builds a quadratic model of the objective function. Steps are proposed by minimizing this model within a trust region. When the quadratic model is updated, the new model interpolates the objective function in `npt` points, typically `npt`=$2n+1$. We use the C implementation made available by M. Guilbert on `http://www.inrialpes.fr/bipop/ people/guilbert/newuoa/newuoa.html`. This code is based on the original FORTRAN implementation of NEWUOA by Powell. We use the standard setting `npt`=$2n+1$ and $\rho_{\text{beg}} = 1$, $\rho_{\text{end}} = 10^{-14}/\ell$.

6.2.7 Implicit Filtering

Implicit filtering (IMFIL) is a hybrid of a Quasi-Newton and a VM scheme. The gradients and Hessians are approximated by finite differences. We use the MATLAB code by Kelly [13], available on `http://www4.ncsu.edu/~ctk/imfil.html` with the setting `smooth_problem`=1 and `bscales`= $(1, 2^{-1}, \ldots, 2^{-100})$ to avoid premature convergence.

6.3 Convergence on the function pair f_1/f_2

We test the convergence of all algorithms on f_1 and f_2 for dimension $n = 20$. We performed 31 independent trials of the same experiment. We let the algorithms run until either the accuracy 10^{-14} was successfully reached, or a budget of total $200n^2$ function evaluations (FES) was consumed. In addition to the number of FES we also recorded the run time (in seconds) needed to perform a single trial. All algorithms where executed on a single core. We report the number of function evaluations performed by the algorithm to reduce the function value by one order of magnitude.

We first focus on the results for function f_1 as it presents a kind of worst-case scenario for adaptive schemes. The data for the benchmark set is listed in Table 4 in the Appendix. Table 2 shows a subset of these data, neglecting non-adaptive

acc.	NEWUOA	IMFIL	N-M	Pattern	CMA-ES	V-RP ES	V-RP Exact
10^{7}	0.06	1.58	0.74	2.18	0.23	0.68	0.11
10^{6}	0.13	2.32	1.72	6.52	0.57	1.21	0.30
10^{5}	0.17	3.07	2.52	10.63	0.93	1.77	0.53
10^{4}	0.23	3.91	3.19	15.32	1.28	2.29	0.80
10^{3}	0.28	4.68	3.71	19.79	1.62	2.83	1.09
10^{2}	0.34	5.59	4.11	25.18	2.27	3.37	1.40
10^{1}	0.40	-	6.58	-	15.29	6.15	2.97
10^{0}	175.60	-	-	-	30.44	11.45	13.89
10^{-1}	-	-	-	-	45.58	13.75	17.52
10^{-2}	-	-	-	-	55.81	14.74	19.25
10^{-3}	-	-	-	-	62.74	15.68	20.07
10^{-4}	-	-	-	-	66.43	16.61	20.61
10^{-5}	-	-	-	-	69.65	17.49	21.11
10^{-6}	-	-	-	-	70.65	18.49	21.65
10^{-7}	-	-	-	-	71.81	19.42	22.17
10^{-8}	-	-	-	-	72.23	20.37	22.75
sec.	438.34	5711.35	10.70	1904.77	29.63	45.37	35.92

Table 2 Accuracy vs. number of FES/n^2 on f_1 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached within a budget of $200n^2$ FES. V-RP is implemented with update scheme `updateHessStore` for two different implementations of the line search (ES and Exact). Average computation time of a single run on a single core CPU; the time until the budget of $200n^2$ FES is exceeded or the time needed to reach accuracy 10^{-14}.

schemes as well as some combinations of V-RP and line search implementation. The data in Table 2 are graphically depicted in Fig. 7. We observe that among the successful algorithms (CMA-ES, V-RP ES, and V-RP Exact) CMA-ES needed about a factor of 3.4 more FES to reach accuracy 10^{-9} than both V-RP schemes but only needs half the run time. The other four algorithms (NEWUOA, IMFIL, Nelder-Mead, Pattern search) only managed to reach accuracy 10^{0} - 10^{2} with the same budget of FES. With the exception of Nelder-Mead their execution time is exceeding the time of CMA-ES by a factor of 14-190.

We also observe that all seven tested algorithms make rapid progress at the beginning (up to accuracy roughly 10^{2}). They then get either stuck or – after a learning phase – resume fast convergence toward higher accuracy levels (roughly 10^{-2}-10^{-9}). These phases are typical for these kind of algorithms (as discussed in Section 5.5).

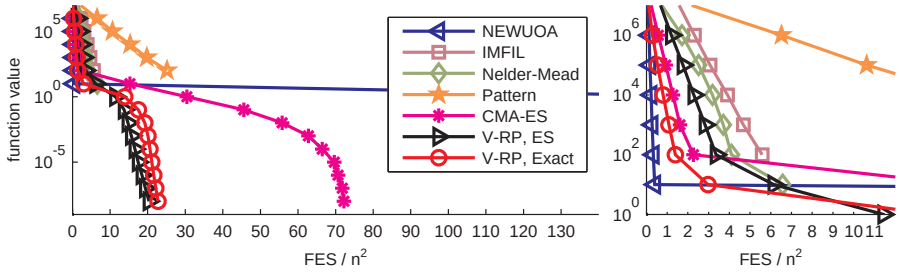

Fig. 7 Reached accuracy vs. number of FES/n^2 on f_1 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). See Tables 2 and 4. V-RP is implemented with update scheme `updateHessStore` for two different implementations of the line search (ES and Exact).

The empirical results on function f_2 reveal several interesting features. NEWUOA, CMA-ES, and V-RP all show faster convergence on this function compared to f_1 (in accordance with previous experiments [27]). These schemes also solve the problem to high accuracy. All other algorithms show, however, reduced performance on f_2

ℓ	NEWUOA	IMFIL	N-M	Pattern	Nesterov Acc.	CMA-ES	V-RP ES
0	0.70	4.08	-	66.35	3.96	3.51	8.57
1	0.81	7.39	-	103.61	9.77	3.83	9.35
2	2.22	8.25	-	-	37.54	6.06	10.61
3	6.74	10.60	-	-	138.25	11.20	11.60
4	20.03	-	-	-	-	18.56	12.39
5	51.14	-	-	-	-	26.45	14.01
6	105.19	-	-	-	-	34.41	16.51
7	-	-	-	-	-	42.82	19.40

Table 3 Number of FES/n^2 to reach accuracy 10^{-9} on f_3 with parameter ℓ, in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. V-RP is implemented with update scheme `updateHessStore` and line search ES.

when compared to f_1 (see data in Tables 4 and 5 in the Appendix). Both Pattern Search and Nesterov's schemes do not reach an accuracy below 10^5. RP ES, IMFIL, and N-M need a considerably higher number of FES to reach an accuracy of 10^3.

In summary, these results show that only adaptive schemes and, to some extent, NEWUOA, are competitive algorithms in the presence of ill-conditioning. The results also suggest that the performance of popular methods such as Implicit Filtering and Nelder-Mead (the standard derivative-free method in MATLAB) are not suitable *even for problems where only a few dimensions are not on the same scale* (such as f_2).

6.4 Evaluating the power of adaptation

Given practical limitations on the available budget of function evaluations it is natural to ask whether function evaluations should be rather spent on estimating the Hessian or for direct function optimization. In order to evaluate the power of adaptation we test the described algorithms on Rosenbrock's function and the following parametric set of functions with increasing curvature. We consider f_3 with parameters $\ell = 10^i$ for $i = 0, \ldots, 7$. For $i = 0$ this function equals the so-called sphere function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{x}$. We report the average number of FES needed to reach accuracy 10^{-9} on each function (for 31 independent trials). The data of the full benchmark set are listed in Tables 7-14 in the appendix. Table 3 shows a subset of algorithms including Nesterov's accelerated RG scheme. A subset of the data from Tables 7-14 is also graphically depicted in Fig. 8 and Fig. 12.

We observe that V-RP-ES (with `updateHessStore` and line search ES) outperforms all algorithms for $\ell > 3$. Nesterov's non-adaptive accelerated RG scheme is superior to V-RP only for the isotropic case $\ell = 0$. IMFIL reaches the target accuracy only for $\ell \leq 3$, but in this regime it is more efficient than V-RP. NEWUOA is superior to V-RP for $\ell \leq 3$. For $\ell \geq 4$ NEWUOA needs a rapidly increasing number of FES and can not reach the target accuracy for $\ell = 7$. CMA-ES is superior for $\ell \leq 3$ and is outperformed by V-RP for $\ell > 3$. Nelder-Mead fails for all settings to reach the target accuracy (its progress can be observed in Fig. 8 and Fig. 12). Pattern search is only successful for $\ell \leq 1$ and needs at least a factor of 10 times more FES than all other algorithms.

Finally, only the V-RP algorithms, CMA-ES, and NEWUOA are able to solve Rosenbrock's function f_4 in $n = 20$ dimensions (see Table 6 for all data). Figure 9 shows the trajectories of NEWUOA, CMA-ES, V-RP (with `updateHessStore` and exact/ES line search), and Pattern search. Surprisingly, the NEWUOA outperforms both CMA-ES and all V-RP variants. None of the non-adaptive algorithms shows competitive performance. Pattern search and standard RP with ES line search reach

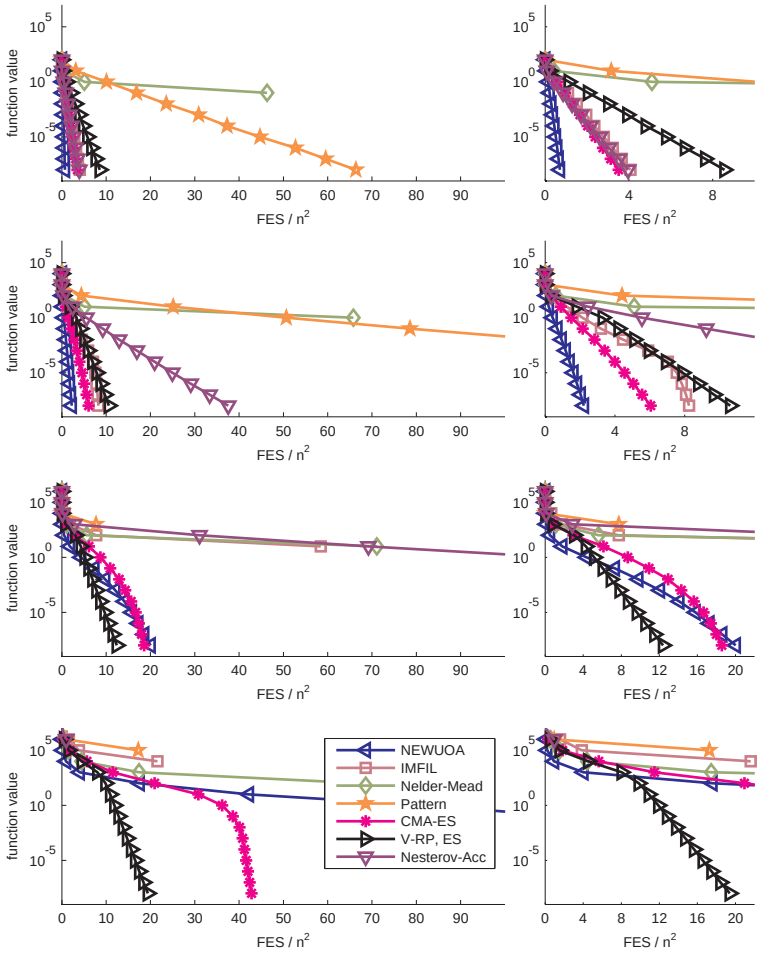

Fig. 8 Reached accuracy vs. number of FES/n^2 on f_3 with parameters $\ell = 10^0$, $\ell = 10^2$, $\ell = 10^4$, $\ell = 10^7$ (from top to bottom) in $n = 20$ dimensions (mean over 31 independent runs). See Fig. 12 for parameters $\ell = 10^1$, $\ell = 10^2$, $\ell = 10^5$, $\ell = 10^6$. V-RP is implemented with update scheme `updateHessStore` and line search ES. See Tables 7, 9, 11 and 14 for the corresponding data and Table 3 for a summary.

an accuracy of 10^1 and 10^0, respectively. Implicit filtering, Nelder-Mead, and Nesterov's schemes even fail to get an accuracy of 10^1.

In summary, the empirical results from the presented benchmark clearly show the superiority of adaptive schemes such as V-RP and CMA-ES.

7 Discussion and Conclusion

In this contribution we have analyzed Random Pursuit algorithms that employ (i) a fixed but arbitrary metric (Fixed Metric Random Pursuit) and (ii) a variable metric learning procedure (Variable Metric Random Pursuit). We have detailed convergence proofs and convergence rates for these Random Pursuit algorithms on convex functions. We have used an improved (matrix) quadratic upper bound technique to show expected single-step progress and global convergence of Fixed Metric Random Pursuit on (strictly) convex functions. We have also shown that Variable Metric Random Pursuit can achieve an optimal convergence rate on convex quadratic functions that, after a finite learning phase of length at most $O(n^2)$, does not depend on the underlying properties of the unknown Hessian of the func-

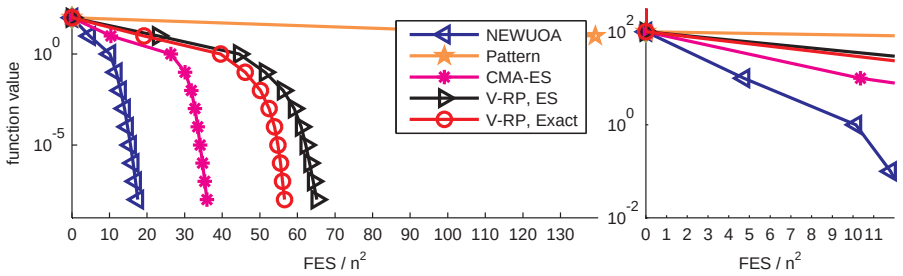

Fig. 9 Reached accuracy vs. number of FES/n^2 on f_4 in $n = 20$ dimensions (mean over 31 independent runs). See Tab. 6.

tion. Compared to standard Random Pursuit [29] we have thus removed one of our previously identified challenges toward the design of competitive gradient-free optimization methods that are easy to implement, possess theoretical convergence guarantees, and are useful in practice.

The numerical experiments show that adaptive schemes are in general (condition number exceeding 10^3) superior to non-adaptive schemes. Both V-RP and CMA-ES outperformed the other tested algorithms in this study, both in terms of FES and time efficiency. It is also noteworthy that V-RP with `updateHessStore` and ES-type line search can outperform the tested CMA-ES (i.e., the (1,4)-CMA-ES with mirrored sampling and sequential selection [2]) which has been reported to be the fasted adaptive scheme on convex quadratic functions known today[2].

V-RP can also be applied to non-convex functions with mildly changing Hessian (such as f_4). However, V-RP schemes are likely to fail if formula (28) returns inaccurate curvature estimates. CMA-type schemes are known to be more robust in these scenarios.

A number of theoretical challenges remain. Firstly, it would be very interesting to give tighter upper and lower bounds on the expected convergence factor for Variable Metric Random Pursuit. This is subject of further research although some initial results have already been obtained in [28]. This paper also suggest an approach to calculate a better bound on the convergence factor $\hat{\varrho}$ by considering the expectation of the exact factor $\varrho(\mathbf{x})$ (cf. Sec. 4.3).

Secondly, it is still an open question how to analyze Random Pursuit schemes for constrained optimization problems of the form

$$\min f(x) \quad \text{subject to} \quad x \in \mathcal{K}\,, \tag{33}$$

where $\mathcal{K} \subset \mathbb{R}^n$ is a convex set. Thirdly, it is an open problem to derive convergence guarantees for Random Pursuit schemes on non-convex functions, such as, e.g., on the class of globally convex (or δ-convex) functions [10] or on noisy functions with certain bounds on the variance of the noise. Finally, convergence on the important class of non-smooth convex functions is another fundamental challenge for gradient-free optimization that, most likely, needs novel tools and techniques to be developed by the mathematical programming community.

Acknowledgements We like to thank the anonymous reviewers whose comments and suggestions very much helped to improve the quality and content of this paper.

References

1. Andrieu, C., Thoms, J.: A tutorial on adaptive MCMC. Statistics and Computing **18**(4), 343–373 (2008). DOI DOI10.1007/s11222-008-9110-y

2. Brockhoff, D., Auger, A., Hansen, N., Arnold, D., Hohm, T.: Mirrored Sampling and Sequential Selection for Evolution Strategies. In: PPSN XI, *LNCS*, vol. 6238, pp. 11–21. Springer (2011)
3. Broyden, C.G.: The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. IMA Journal of Applied Mathematics **6**(1), 76–90 (1970). DOI 10.1093/imamat/6.1.76. URL `http://imamat.oxfordjournals.org/content/6/1/76.abstract`
4. Davidon, W.C.: Variable Metric Method for Minimization. SIAM Journal on Optimization **1**(1), 1–17 (1991). DOI 10.1137/0801001. URL `http://link.aip.org/link/?SJE/1/1/1`
5. Fletcher, R.: A new approach to variable metric algorithms. The Computer Journal **13**(3), 317–322 (1970). DOI 10.1093/comjnl/13.3.317. URL `http://comjnl.oxfordjournals.org/content/13/3/317.abstract`
6. Goldfarb, D.: A Family of Variable-Metric Methods Derived by Variational Means. Mathematics of Computation **24**(109), 23–26 (1970). URL `http://www.jstor.org/stable/2004873`
7. Hansen, N., Ostermeier, A.: Completely Derandomized Self-Adaption in Evolution Strategies. Evolutionary Computation **9**(2), 159–195 (2001)
8. Heijmans, R.: When does the expectation of a ratio equal the ratio of expectations? Statist. Papers **40**, 107–115 (1999)
9. Horn, R.A., Johnson, C.R.: Matrix analysis, reprint 1990 edn. Cambridge University Press (1985)
10. Hu, T.C., Klee, V., Larman, D.: Optimization of globally convex functions. SIAM Journal on Control and Optimization **27**(5), 1026–1047 (1989). DOI 10.1137/0327055. URL `http://link.aip.org/link/?SJC/27/1026/1`
11. Isserlis, L.: On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. Biometrika **12**, 134–139 (1918)
12. Jägersküpper, J.: Lower bounds for hit-and-run direct search. In: J. Hromkovic, R. Královic, M. Nunkesser, P. Widmayer (eds.) Stochastic Algorithms: Foundations and Applications, *Lecture Notes in Comput. Sci.*, vol. 4665, pp. 118–129. Springer Berlin (2007)
13. Kelley, C.T.: Implicit Filtering. SIAM (2011)
14. Kjellström, G., Taxen, L.: Stochastic Optimization in System Design. IEEE Trans. Circuits Systems **28**(7) (1981)
15. Leventhal, D., Lewis, A.S.: Randomized Hessian estimation and directional search. Optimization **60**(3), 329–345 (2011). DOI 10.1080/02331930903100141. URL `http://www.tandfonline.com/doi/abs/10.1080/02331930903100141`
16. Marti, K.: Controlled random search procedures for global optimization. In: V. Arkin, A. Shiraev, R. Wets (eds.) Stochastic Optimization, *Lecture Notes in Control and Information Sciences*, vol. 81, pp. 457–474. Springer (1986)
17. Müller, C.L., Sbalzarini, I.F.: Gaussian adaptation revisited - an entropic view on covariance matrix adaptation. In: C. Di Chio et al. (ed.) EvoApplications, no. 6024 in Lecture Notes in Comput. Sci., pp. 432–441. Springer, Berlin (2010)
18. Nelder, J.A., Mead, R.: A Simplex Method for Function Minimization. The Computer Journal **7**(4), 308–313 (1965). DOI 10.1093/comjnl/7.4.308. URL `http://comjnl.oxfordjournals.org/content/7/4/308.abstract`
19. Nesterov, Y.: Random Gradient-Free Minimization of Convex Functions. Tech. rep., ECORE (2011)
20. Polyak, B.: Introduction to Optimization. Optimization Software - Inc, Publications Division, New York (1987)
21. Powell, M.: The newuoa software for unconstrained optimization without derivatives. In: G. Pillo, M. Roma (eds.) Large-Scale Nonlinear Optimization, *Nonconvex Optimization and Its Applications*, vol. 83, pp. 255–297. Springer US (2006). DOI 10.1007/0-387-30065-1_16. URL `http://dx.doi.org/10.1007/0-387-30065-1\_16`
22. R2012a, M.: http://www.mathworks.ch/help/toolbox/optim/ug/fminunc.html
23. Rosenbrock, H.H.: An automatic method for finding the greatest or least value of a function. The Computer Journal **3**(3), 175–184 (1960). DOI 10.1093/comjnl/3.3.175. URL `http://comjnl.oxfordjournals.org/content/3/3/175.abstract`
24. Rudelson, M., Vershynin, R.: Sampling from large matrices: An approach through geometric functional analysis. Journal of the ACM (JACM) **54**(4), 21 (2007)
25. Schumer, M., Steiglitz, K.: Adaptive step size random search. Automatic Control, IEEE Transactions on **13**(3), 270–276 (1968). DOI 10.1109/TAC.1968.1098903
26. Shanno, D.F.: Conditioning of Quasi-Newton Methods for Function Minimization. Mathematics of Computation **24**(111), 647–656 (1970). URL `http://www.jstor.org/stable/2004840`
27. Stich, S.U., Müller, C.L.: On spectral invariance of randomized hessian and covariance matrix adaptation schemes. In: C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, M. Pavone (eds.) Parallel Problem Solving from Nature - PPSN XII, *Lecture Notes in Computer Science*, vol. 7491, pp. 448–457. Springer Berlin Heidelberg (2012)
28. Stich, S.U., Müller, C.L., Gärtner, B.: Matrix-valued Iterative Random Projections. Tech. rep., ETH Zürich (2013). URL `http://people.inf.ethz.ch/sstich/mrp.pdf`. Technical Report CGL-TR-xx
29. Stich, S.U., Müller, C.L., Gärtner, B.: Optimization of convex functions with Random Pursuit. SIAM Journal on Optimization **23**(2), 1284–1309 (2013)
30. Wedderburn, J.H.M.: Lectures on Matrices (Colloquium Publications). AMS, New York (1938)

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{7}	0.10	0.06	1.58	0.74	2.18	0.36	0.27	0.23	0.72	0.19	0.35	0.68	0.11	0.22
10^{6}	0.25	0.13	2.32	1.72	6.52	0.90	0.92	0.57	1.27	0.61	1.28	1.21	0.30	0.74
10^{5}	0.40	0.17	3.07	2.52	10.63	1.47	2.54	0.93	1.80	0.93	2.08	1.77	0.53	1.37
10^{4}	0.55	0.23	3.91	3.19	15.32	2.04	3.91	1.28	2.30	1.24	2.88	2.29	0.80	1.97
10^{3}	0.71	0.28	4.68	3.71	19.79	2.60	5.05	1.62	2.80	1.55	3.59	2.83	1.09	2.64
10^{2}	0.88	0.34	5.59	4.11	25.18	3.18	-	2.27	3.37	1.84	4.51	3.37	1.40	3.49
10^{1}	-	0.40	-	6.58	-	3.92	-	15.29	17.93	11.47	32.47	6.15	2.97	12.42
10^{0}	-	175.60	-	-	-	-	-	30.44	34.11	44.37	86.42	11.45	13.89	30.24
10^{-1}	-	-	-	-	-	-	-	45.58	40.75	52.29	101.82	13.75	17.52	36.26
10^{-2}	-	-	-	-	-	-	-	55.81	45.26	59.92	113.60	14.74	19.25	39.83
10^{-3}	-	-	-	-	-	-	-	62.74	48.28	65.66	121.34	15.68	20.07	41.49
10^{-4}	-	-	-	-	-	-	-	66.43	50.45	69.75	127.00	16.61	20.61	42.54
10^{-5}	-	-	-	-	-	-	-	69.65	52.16	72.68	131.63	17.49	21.11	43.62
10^{-6}	-	-	-	-	-	-	-	70.65	53.72	74.87	135.02	18.49	21.65	44.86
10^{-7}	-	-	-	-	-	-	-	71.81	54.99	76.78	137.87	19.42	22.17	46.00
10^{-8}	-	-	-	-	-	-	-	72.23	56.15	78.39	140.34	20.37	22.75	47.10
sec.	30.77	438.34	5711.35	10.70	1904.77	18.26	22.68	29.63	16.83	18.65	85.60	45.37	35.92	521.82

Table 4 Reached accuracy vs. number of FES/n^2 on f_1 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

A Appendix

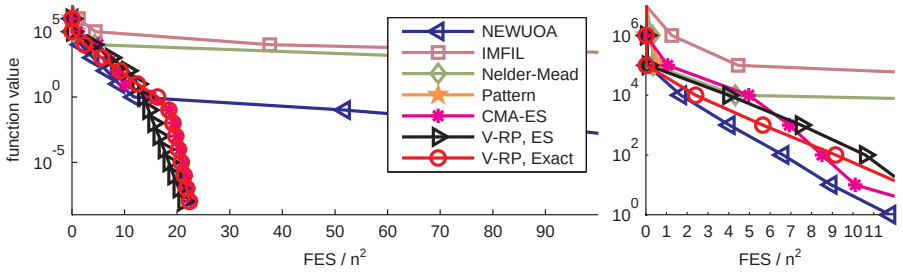

Fig. 10 Reached accuracy vs. number of FES/n^2 on f_2 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). See Tab. 5.

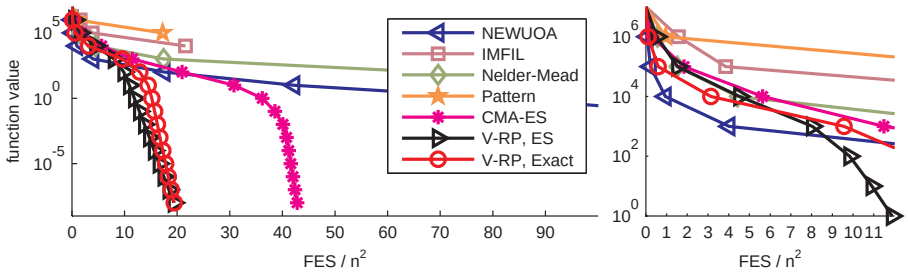

Fig. 11 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). See Tab. 14.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{6}	0.01	0.05	1.26	0.29	0.01	0.35	0.25	0.01	0.02	0.01	0.02	0.02	0.01	0.02
10^{5}	0.05	0.08	4.45	0.35	0.71	0.94	0.42	1.06	0.10	0.10	0.08	0.11	0.03	0.20
10^{4}	8.54	1.69	37.59	4.33	-	-	-	4.97	3.79	2.48	5.08	3.99	2.41	4.88
10^{3}	19.20	4.03	144.48	72.65	-	-	-	6.92	7.40	5.82	11.90	7.49	5.65	12.21
10^{2}	29.92	6.63	-	-	-	-	-	8.50	10.20	9.08	18.22	10.59	9.16	20.04
10^{1}	41.29	8.95	-	-	-	-	-	10.11	12.93	12.54	24.20	12.52	12.43	26.54
10^{0}	-	11.78	-	-	-	-	-	14.99	21.38	31.35	45.85	13.64	16.29	33.48
10^{-1}	-	51.82	-	-	-	-	-	18.59	32.83	43.32	72.59	14.59	18.43	37.34
10^{-2}	-	82.25	-	-	-	-	-	19.48	36.66	51.46	89.86	15.52	19.19	38.63
10^{-3}	-	104.93	-	-	-	-	-	20.03	38.28	54.49	97.54	16.53	19.72	39.72
10^{-4}	-	113.03	-	-	-	-	-	20.56	39.45	55.86	100.07	17.49	20.25	40.86
10^{-5}	-	117.40	-	-	-	-	-	21.10	40.56	56.95	102.38	18.43	20.80	41.98
10^{-6}	-	120.75	-	-	-	-	-	21.61	41.53	57.76	104.36	19.35	21.30	43.15
10^{-7}	-	123.84	-	-	-	-	-	22.13	42.52	58.49	105.94	20.34	21.82	44.24
10^{-8}	-	126.26	-	-	-	-	-	22.60	43.55	59.24	107.37	21.25	22.36	45.36
sec.	30.76	274.79	7976.66	17.38	1926.26	17.36	22.94	8.90	13.13	13.86	85.21	46.66	37.94	528.10

Table 5 Reached accuracy vs. number of FES/n^2 on f_2 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{1}	57.21	4.73	-	-	139.26	-	-	10.36	24.83	25.20	53.62	22.95	19.20	44.56
10^{0}	186.64	10.14	-	-	-	-	-	26.30	49.79	50.42	105.63	44.52	39.54	89.59
10^{-1}	-	11.81	-	-	-	-	-	30.14	58.47	59.58	123.52	51.43	46.12	103.31
10^{-2}	-	12.94	-	-	-	-	-	31.76	64.52	65.67	134.91	56.18	50.18	110.95
10^{-3}	-	13.98	-	-	-	-	-	32.71	68.02	70.16	141.93	59.14	52.45	114.90
10^{-4}	-	14.81	-	-	-	-	-	33.47	70.26	72.96	146.84	61.02	53.90	117.40
10^{-5}	-	15.60	-	-	-	-	-	34.12	72.00	74.94	150.17	62.24	54.82	119.00
10^{-6}	-	16.29	-	-	-	-	-	34.76	73.31	76.42	152.90	63.23	55.49	120.30
10^{-7}	-	16.93	-	-	-	-	-	35.35	74.48	77.67	154.85	64.20	56.07	121.48
10^{-8}	-	17.55	-	-	-	-	-	35.96	75.56	78.85	156.44	65.16	56.60	122.61
sec.	32.90	6.55	8588.06	14.38	1979.45	19.20	24.77	13.96	11.62	10.94	78.11	140.89	117.74	373.41

Table 6 Reached accuracy vs. number of FES/n^2 on f_4 in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{1}	0.07	0.06	0.23	0.42	3.16	0.08	0.07	0.24	0.19	0.11	0.16	0.21	0.10	0.18
10^{0}	0.35	0.15	0.66	5.10	10.06	0.45	0.44	0.59	1.13	0.64	1.14	1.14	0.57	1.02
10^{-1}	0.67	0.22	1.09	46.34	16.86	0.87	0.86	0.95	2.04	1.19	2.28	2.06	1.10	2.09
10^{-2}	0.99	0.30	1.51	-	23.58	1.35	1.29	1.30	2.98	1.73	3.38	3.02	1.64	3.14
10^{-3}	1.32	0.37	1.96	-	30.87	1.85	1.72	1.66	3.90	2.31	4.44	3.93	2.19	4.16
10^{-4}	1.64	0.44	2.35	-	37.27	2.35	2.17	2.03	4.88	2.90	5.61	4.81	2.71	5.26
10^{-5}	1.97	0.51	2.80	-	44.71	2.86	2.62	2.39	5.80	3.46	6.80	5.78	3.24	6.42
10^{-6}	2.32	0.58	3.23	-	52.74	3.37	3.05	2.76	6.75	3.99	7.94	6.69	3.79	7.64
10^{-7}	2.66	0.64	3.66	-	59.56	3.88	3.53	3.13	7.67	4.54	9.08	7.65	4.30	8.73
10^{-8}	3.02	0.70	4.08	-	66.35	4.40	3.96	3.51	8.63	5.08	10.28	8.57	4.87	9.89
sec.	1.15	0.26	28.37	16.94	1292.33	17.88	NaN	1.50	2.25	1.36	44.50	21.90	2.23	135.19

Table 7 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^0$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{1}	0.23	0.08	0.42	1.42	12.38	0.73	0.49	0.43	0.73	0.43	0.72	0.75	0.30	0.49
10^{0}	0.54	0.20	0.98	22.41	20.49	2.00	1.20	0.79	1.79	1.14	2.10	1.86	0.90	1.49
10^{-1}	0.91	0.27	1.57	-	30.73	3.53	2.26	1.17	2.78	1.91	3.51	2.89	1.51	2.66
10^{-2}	1.27	0.36	2.17	-	41.19	5.26	3.42	1.54	3.79	2.60	4.88	3.83	2.08	3.77
10^{-3}	1.61	0.43	2.97	-	51.40	7.01	4.38	1.92	4.70	3.21	6.19	4.79	2.60	4.84
10^{-4}	1.97	0.51	3.93	-	61.55	8.84	5.39	2.30	5.68	3.83	7.47	5.71	3.12	5.98
10^{-5}	2.31	0.59	4.93	-	72.39	10.65	6.53	2.69	6.63	4.45	8.63	6.62	3.65	7.12
10^{-6}	2.63	0.67	5.91	-	82.77	12.55	7.58	3.07	7.59	5.01	9.78	7.56	4.18	8.19
10^{-7}	2.98	0.74	6.75	-	93.21	14.48	8.64	3.45	8.60	5.59	10.99	8.47	4.69	9.39
10^{-8}	3.35	0.81	7.39	-	103.61	16.42	9.77	3.83	9.53	6.14	12.15	9.35	5.20	10.48
sec.	1.09	0.27	42.87	16.93	1652.45	18.72	22.82	1.64	2.26	1.47	32.35	23.23	2.60	131.40

Table 8 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^1$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

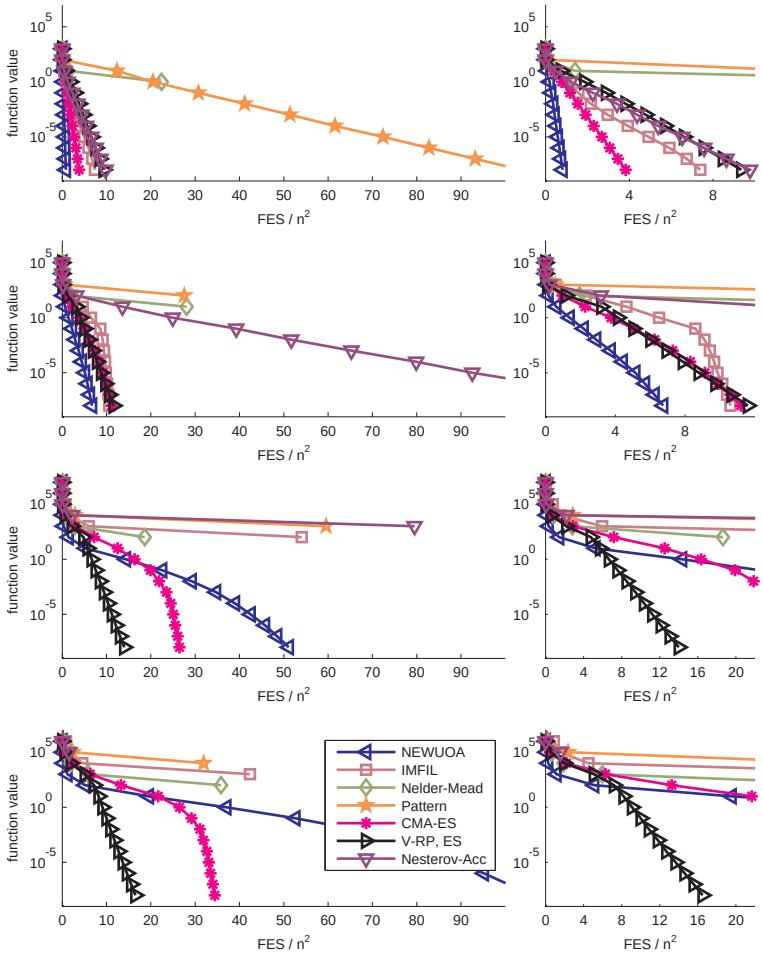

Fig. 12 Reached accuracy vs. number of FES/n^2 on f_3 with parameters $\ell = 10^1$, $\ell = 10^2$, $\ell = 10^5$, $\ell = 10^6$ (from top to bottom) in $n = 20$ dimensions (mean over 31 independent runs). See Fig. 8 for parameters $\ell = 10^0$, $\ell = 10^2$, $\ell = 10^4$, $\ell = 10^7$. V-RP is implemented with update scheme `updateHessStore` and line search ES. See Tables 8, 9, 12 and 13 for the corresponding data and Table 3 for a summary.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^2	0.10	0.06	0.53	0.61	4.41	0.55	0.35	0.35	0.35	0.13	0.28	0.33	0.12	0.19
10^1	0.54	0.21	1.25	5.11	25.12	4.13	2.46	0.90	1.82	1.18	2.24	1.83	0.83	1.42
10^0	1.42	0.44	2.14	65.80	50.68	14.13	5.55	1.51	3.40	2.53	4.77	3.31	1.85	3.53
10^{-1}	2.50	0.70	3.20	-	78.55	28.72	9.23	2.15	4.75	3.98	7.39	4.21	2.93	5.55
10^{-2}	3.68	0.94	4.47	-	108.86	45.79	12.95	2.83	5.88	5.17	9.69	5.12	3.91	7.26
10^{-3}	4.91	1.17	5.91	-	-	64.04	16.96	3.41	6.98	6.26	11.70	5.95	4.68	8.67
10^{-4}	6.17	1.40	7.03	-	-	83.27	20.92	3.99	8.07	7.22	13.33	6.94	5.37	10.00
10^{-5}	7.42	1.62	7.55	-	-	102.76	24.94	4.55	8.99	8.05	14.77	7.89	5.99	11.19
10^{-6}	8.72	1.83	7.83	-	-	122.68	29.09	5.05	9.96	8.80	16.18	8.78	6.56	12.37
10^{-7}	10.10	2.04	8.07	-	-	142.71	33.35	5.56	10.88	9.48	17.55	9.68	7.14	13.54
10^{-8}	11.50	2.22	8.25	-	-	162.88	37.54	6.06	11.80	10.12	18.85	10.61	7.70	14.62
sec.	3.37	0.50	45.49	16.30	1863.95	17.80	23.71	2.30	2.67	2.02	43.35	28.89	10.28	383.55

Table 9 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^2$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{3}	0.03	0.06	0.56	0.33	0.56	0.27	0.22	0.14	0.12	0.05	0.08	0.10	0.05	0.07
10^{2}	0.38	0.17	2.30	1.95	27.54	3.63	3.17	0.97	1.27	0.81	1.52	1.17	0.58	0.97
10^{1}	2.12	0.61	4.64	28.07	-	27.79	13.60	2.27	3.81	3.15	4.99	3.20	2.26	4.06
10^{0}	6.65	1.38	6.51	-	-	115.70	24.95	3.76	6.45	6.36	10.71	4.15	4.80	8.52
10^{-1}	13.08	2.18	8.59	-	-	-	39.27	5.06	8.38	8.97	15.30	5.05	5.77	10.64
10^{-2}	20.44	2.90	9.11	-	-	-	51.70	6.25	9.92	10.84	18.93	6.02	6.32	11.60
10^{-3}	28.33	3.73	9.42	-	-	-	65.28	7.29	11.22	12.38	21.99	6.94	6.86	12.61
10^{-4}	36.81	4.42	9.66	-	-	-	79.90	8.30	12.31	13.62	24.44	7.87	7.36	13.78
10^{-5}	45.38	5.14	9.85	-	-	-	92.56	9.13	13.33	14.74	26.33	8.78	7.88	14.89
10^{-6}	54.13	5.70	10.13	-	-	-	107.68	9.89	14.24	15.71	28.09	9.74	8.36	15.99
10^{-7}	63.07	6.24	10.41	-	-	-	121.35	10.59	15.23	16.61	29.74	10.67	8.86	17.06
10^{-8}	72.09	6.74	10.60	-	-	-	138.25	11.20	16.15	17.42	31.22	11.60	9.38	18.21
sec.	23.01	1.61	65.92	18.51	1843.56	19.05	21.96	4.28	4.02	3.09	48.35	31.62	19.01	596.88

Table 10 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^3$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{4}	0.01	0.03	0.64	0.23	0.01	0.14	0.11	0.02	0.04	0.02	0.03	0.04	0.02	0.04
10^{3}	0.23	0.12	1.81	1.13	7.74	2.64	2.84	0.93	0.84	0.49	0.67	0.81	0.30	0.61
10^{2}	2.00	0.50	7.75	5.61	-	26.29	31.08	2.97	2.95	2.32	3.83	3.06	1.66	3.36
10^{1}	11.98	1.80	58.45	71.09	-	-	69.18	6.13	7.71	7.08	12.08	4.28	5.48	9.98
10^{0}	38.13	4.36	-	-	-	-	111.78	8.69	11.54	12.30	23.04	5.15	6.22	11.36
10^{-1}	79.47	7.32	-	-	-	-	160.50	10.91	14.12	16.49	30.71	6.06	6.70	12.36
10^{-2}	127.32	9.78	-	-	-	-	-	12.92	15.83	19.27	36.45	6.92	7.21	13.37
10^{-3}	-	12.06	-	-	-	-	-	14.31	17.16	21.34	39.90	7.83	7.74	14.39
10^{-4}	-	14.10	-	-	-	-	-	15.63	18.35	23.05	42.88	8.72	8.23	15.54
10^{-5}	-	15.80	-	-	-	-	-	16.66	19.40	24.38	45.28	9.65	8.76	16.72
10^{-6}	-	17.25	-	-	-	-	-	17.42	20.40	25.50	47.26	10.54	9.30	17.83
10^{-7}	-	18.60	-	-	-	-	-	18.01	21.36	26.51	48.93	11.49	9.81	18.92
10^{-8}	-	20.03	-	-	-	-	-	18.56	22.31	27.41	50.54	12.39	10.32	20.06
sec.	35.37	5.24	8588.05	17.96	1853.12	17.40	21.82	7.07	5.21	5.08	58.95	29.18	23.94	624.25

Table 11 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^4$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{5}	0.01	0.01	0.76	0.14	0.01	0.06	0.06	0.01	0.02	0.02	0.02	0.02	0.01	0.02
10^{4}	0.16	0.09	1.59	0.86	2.87	2.01	2.21	0.71	0.62	0.27	0.44	0.60	0.20	0.33
10^{3}	1.19	0.33	5.95	2.83	59.55	20.91	79.53	2.87	2.38	1.59	2.79	2.19	1.30	2.58
10^{2}	11.48	1.44	54.07	18.63	-	-	-	7.18	7.16	6.79	11.99	4.77	5.56	11.23
10^{1}	88.03	5.25	-	-	-	-	-	12.53	14.00	15.21	29.17	5.74	7.58	14.94
10^{0}	-	14.35	-	-	-	-	-	16.35	18.34	22.97	44.82	6.66	8.16	16.07
10^{-1}	-	22.55	-	-	-	-	-	19.91	21.17	27.41	53.49	7.57	8.68	17.21
10^{-2}	-	29.21	-	-	-	-	-	21.87	22.87	30.26	58.76	8.52	9.20	18.31
10^{-3}	-	34.72	-	-	-	-	-	23.54	24.24	32.46	62.10	9.45	9.75	19.39
10^{-4}	-	38.96	-	-	-	-	-	24.50	25.42	34.22	64.89	10.34	10.25	20.53
10^{-5}	-	42.51	-	-	-	-	-	25.06	26.47	35.58	67.16	11.22	10.75	21.68
10^{-6}	-	45.76	-	-	-	-	-	25.52	27.47	36.76	69.11	12.11	11.26	22.88
10^{-7}	-	48.49	-	-	-	-	-	26.03	28.42	37.84	70.90	13.05	11.76	24.03
10^{-8}	-	51.14	-	-	-	-	-	26.45	29.37	38.76	72.66	14.01	12.33	25.13
sec.	34.46	50.67	8273.24	16.28	1863.09	19.17	21.96	10.08	6.69	7.15	66.13	36.11	26.35	569.45

Table 12 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^5$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{6}	0.01	0.01	0.87	0.04	0.01	0.01	0.01	0.01	0.01	0.01	0.02	0.02	0.01	0.02
10^{5}	0.10	0.08	1.57	0.65	2.37	1.76	1.57	0.45	0.60	0.19	0.26	0.56	0.11	0.23
10^{4}	0.77	0.27	4.51	1.89	31.90	17.38	-	2.20	1.93	1.07	2.26	1.98	0.80	1.79
10^{3}	8.45	1.15	42.37	5.98	-	170.48	-	6.29	6.40	4.41	10.98	5.08	4.26	10.04
10^{2}	79.40	5.11	-	35.85	-	-	-	13.25	13.16	12.54	33.67	7.20	9.16	20.72
10^{1}	-	19.53	-	-	-	-	-	21.68	20.56	24.92	58.73	8.21	10.67	24.05
10^{0}	-	37.01	-	-	-	-	-	26.47	25.09	33.63	74.31	9.09	11.38	25.24
10^{-1}	-	52.61	-	-	-	-	-	29.18	27.85	38.78	83.25	9.99	11.92	26.35
10^{-2}	-	65.85	-	-	-	-	-	31.09	29.72	41.81	88.14	10.87	12.41	27.41
10^{-3}	-	74.92	-	-	-	-	-	31.90	31.25	43.96	91.83	11.80	12.95	28.41
10^{-4}	-	82.32	-	-	-	-	-	32.64	32.40	45.62	94.67	12.70	13.45	29.53
10^{-5}	-	89.27	-	-	-	-	-	33.08	33.45	46.93	97.09	13.67	13.97	30.68
10^{-6}	-	95.14	-	-	-	-	-	33.48	34.41	48.12	99.09	14.60	14.49	31.81
10^{-7}	-	100.65	-	-	-	-	-	34.01	35.32	49.17	100.86	15.57	14.99	32.99
10^{-8}	-	105.19	-	-	-	-	-	34.41	36.34	50.05	102.39	16.51	15.52	34.21
sec.	30.90	194.63	7987.02	16.29	1888.71	17.42	22.54	12.99	8.47	9.21	76.31	40.34	29.91	547.15

Table 13 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^6$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.

acc.	RP ES	NEWUOA	IMFIL	N-M	Pattern	Nesterov RG	Nesterov Acc.	CMA-ES	V-RP (corr) ES	V-RP (corr) Exact	V-RP (corr) matlab	V-RP (store) ES	V-RP (store) Exact	V-RP (store) matlab
10^{6}	0.08	0.07	1.58	0.56	0.95	1.44	0.96	0.30	0.48	0.11	0.13	0.52	0.12	0.15
10^{5}	0.54	0.23	3.85	1.52	17.25	14.66	-	1.84	1.48	0.81	1.53	1.53	0.59	1.36
10^{4}	6.50	0.94	21.60	4.43	-	142.72	-	5.64	4.94	3.52	6.83	4.53	3.14	7.41
10^{3}	66.49	4.03	-	17.45	-	-	-	11.49	12.13	12.30	28.55	8.04	9.57	22.97
10^{2}	-	17.57	-	67.45	-	-	-	20.94	19.79	24.87	59.06	9.84	12.91	32.27
10^{1}	-	42.19	-	-	-	-	-	30.81	26.41	37.43	91.47	10.91	14.50	36.40
10^{0}	-	82.48	-	-	-	-	-	36.20	31.75	46.10	110.00	11.89	15.19	37.98
10^{-1}	-	113.01	-	-	-	-	-	38.59	35.23	50.36	120.59	12.81	15.72	39.09
10^{-2}	-	133.34	-	-	-	-	-	40.11	37.23	53.69	126.24	13.78	16.25	40.10
10^{-3}	-	149.80	-	-	-	-	-	40.80	38.63	56.18	129.87	14.74	16.73	41.09
10^{-4}	-	165.02	-	-	-	-	-	41.22	39.90	57.91	132.38	15.65	17.27	42.16
10^{-5}	-	177.24	-	-	-	-	-	41.62	40.92	59.22	134.52	16.56	17.81	43.24
10^{-6}	-	-	-	-	-	-	-	42.01	41.92	60.42	136.33	17.49	18.31	44.39
10^{-7}	-	-	-	-	-	-	-	42.41	42.92	61.56	137.85	18.43	18.88	45.66
10^{-8}	-	-	-	-	-	-	-	42.82	43.94	62.44	139.45	19.40	19.44	46.85
sec.	30.98	447.57	7705.72	17.72	1861.84	18.08	22.77	16.30	10.45	11.96	74.15	45.21	34.37	519.52

Table 14 Reached accuracy vs. number of FES/n^2 on f_3 with parameter $\ell = 10^7$ in $n = 20$ dimensions (mean over 31 independent runs). A dash '-' indicates that accuracy could not be reached withing a budget of $200n^2$ FES. See main text for further information.