Principled Design of Continuous Stochastic Search: From Theory to Practice

Nikolaus Hansen, Anne Auger

Contents

Principled Design of Continuous Stochastic Search: From Theory to Practice

Nikolaus Hansen and Anne Auger

Abstract We derive a stochastic search procedure for parameter optimization from two first principles: (1) imposing the least prior assumptions, namely by maximum entropy sampling, unbiasedness and invariance; (2) exploiting all available information under the constraints imposed by (1). We additionally require that two of the most basic functions can be solved reasonably fast. Given these principles, two principal heuristics are used: reinforcing of good solutions and good steps (increasing their likelihood) and rendering successive steps orthogonal. The resulting search algorithm is the *covariance matrix adaptation evolution strategy*, CMA-ES that coincides to a great extend to a natural gradient descent. The invariance properties of the CMA-ES are formalized, as well as its maximum likelihood and stationarity properties. A small parameter study for a specific heuristic—deduced from the principles of reinforcing good steps and exploiting all information—is presented, namely for the cumulation of an evolution or search path. Experiments on two noisy functions are provided.

1 Introduction: Top Down Versus Bottom Up

Let $f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{x} \mapsto f(\boldsymbol{x})$ be an *objective* or *cost* (or fitness) function to be minimized, where, in practice, the typical search space dimension n obeys $3 < n < 300$. When properties of f are unknown a priori, an iterative search algorithm can proceed in evaluating solutions on f and so gather information for finding better solutions over time (black-box search or optimization). Good solutions have, by definition, a small f-value and evaluations of f are considered as the *cost of search* (remark the double entendre of the word cost for f). The objective is, in practice, to find a

Nikolaus Hansen
INRIA Saclay – Île-de-France, Orsay, France. e-mail: forename.surname@inria.fr

Anne Auger
INRIA Saclay – Île-de-France, Orsay, France. e-mail: forename.surname@inria.fr

Given: a cost function f, a parametrized family of distributions $P(\theta)$, and $\lambda \in \mathbb{N}$
Initialize: $k \leftarrow 0$, set θ_k
Repeat while not happy
Sample: $x_1, \ldots, x_\lambda \sim P(\theta_k)$ i.i.d.
Update: $\theta_{k+1} = \mathit{Update}(\theta_k, x_1, \ldots, x_\lambda, f(x_1), \ldots, f(x_\lambda))$
$k \leftarrow k+1$

Fig. 1 Stochastic search template

good solution with the least number of function evaluations and, more rigorously, to generate a sequence $\boldsymbol{x}_k$, $k = 1, 2, 3 \ldots$, such that $f(\boldsymbol{x}_k)$ converges fast to the *essential infimum* of f, denoted f^*. The essential infimum f^* is the largest real number such that the set of better search points $\{\boldsymbol{x} \in \mathbb{R}^n : f(\boldsymbol{x}) < f^*\}$ has zero volume.

In order to search in continuous spaces with even moderate dimension, some structure in the cost function needs to be exploited. For *evolution strategies*, the principle structure is believed to be *neighborhood*. Strong causality [34]—the principle that small actuator changes have generally only small effects—and fitness-distance correlation [32]—a statistical perspective of the same concept—are two ways to describe the structure that evolution strategies are based upon. In contrast to Chapters **??**, **??** and **??** of this volume [**?**], in this chapter we do not introduce an *a priori* assumption on the problem class we want to address, that is, we do not assume any structure in the cost function *a priori*. However, we use two ideas that might imply the exploitation of neighborhood: we assume that the variances of the sample distribution exist, and we encourage consecutive iteration steps to become, under a variable metric, orthogonal (via step-size control). Empirically, the latter rather reduces the locality of the algorithm: the step-sizes that achieve orthogonality are usually large in their stationary condition. We conjecture therefore that either the mere existence of variances and/or the "any-time" approach that aims to improve in each iteration, rather than only in a final step, implies already the exploitation of a neighborhood structure in our context.

In order to solve the above introduced search problem on f, we take a principled *stochastic* (or *randomized*) approach. We first **sample** points from a distribution over the search space with density $p(.|\theta)$, we **evaluate** the points on f and finally **update** the parameters θ of the distribution. This is done iteratively and defines a search procedure on θ as depicted in Fig. 1. Indeed, the update of θ remains the one and only crucial element—besides the choice of p (and λ) in the first place. Consequently, this chapter is entirely devoted to the question of how to update θ.

Before we proceed, we note that under some mild assumptions on p, and for any increasing transformation $g : \mathbb{R} \to \mathbb{R}$ (in particular also for the identity), the minimum of the function

$$\theta \mapsto E(g(f(\boldsymbol{x}))|\theta) \tag{1}$$

coincides with the minimum of f (the expectation E is taken under the sample distribution p, given parameters θ). The **optimal distribution** is entirely concentrated

in the arg min of f. In black-box search, we do not want (and are not able) to impose strong regularity conditions on the unknown function f. However, we have entire control over p. This seems an excellent justification for a *randomized* approach to the original black-box search problem. We sketch two approaches to solve (1)[1].

The Top Down Way

We might chose p being "sufficiently smooth" and conduct a gradient descent,

$$\theta_{k+1} = \theta_k - \eta \nabla_\theta E(f(\boldsymbol{x})|\theta) \qquad \text{with } \eta > 0 \ . \tag{2}$$

We are facing two problems with Equation (2). On the one hand, we need to compute $\nabla_\theta E(f(\boldsymbol{x})|\theta)$. On the other hand, the gradient ∇_θ strongly depends on the specifically chosen parameterization in θ. The unique solution to the second problem is the *natural gradient*. The idea to use the natural gradient in evolution strategies was coined in [41] and elegantly pursued in [11]. The natural gradient is unique, invariant under reparametrization and in accordance with the KL-divergence or relative entropy, the informational difference measure between distributions. We can reformulate (2) using the natural gradient, denoted $\widetilde{\nabla}$, in a unique way as

$$\theta_{k+1} = \theta_k - \eta \widetilde{\nabla} E(f(\boldsymbol{x})|\theta) \ . \tag{3}$$

We can express the natural gradient in terms of the vanilla gradient ∇_θ, using the Fisher information matrix, as $\widetilde{\nabla} = F_\theta^{-1} \nabla_\theta$. Using the log-likelihood trick, $\nabla_\theta p = (p/p)\nabla_\theta p = p \nabla_\theta \log p$ we can finally, under mild assumption on p, re-arrange (3) into

$$\theta_{k+1} = \theta_k - \eta E(\underbrace{f(\boldsymbol{x})}_{\text{expensive}} \overbrace{F_\theta^{-1} \nabla_\theta \log p(\boldsymbol{x}|\theta)}^{\text{"controlled"}}) \ . \tag{4}$$

In practice, the expectation in (4) can be approximated/replaced by taking the average over a (potentially small) number of samples, $\boldsymbol{x}_i$, where computing $f(\boldsymbol{x}_i)$ is assumed to be the costly part. We will also choose p such that we can conveniently sample from the distribution and that the computation (or approximation) of $F_\theta^{-1} \nabla_\theta \log p$ is feasible. The top down way of (3) and (4) is an amazingly clean and principled approach to stochastic black-box optimization.

The Bottom Up Way

In this chapter, we choose a rather orthogonal approach to derive a principled stochastic search algorithm in the $\mathbb{R}^n$. We take a scrutinizing step-by-step road to

[1] That is, to find a sequence θ_k, $k = 1, 2, 3 \ldots$, such that $\lim_{k\to\infty} E(f(\boldsymbol{x}|\theta_k)) = f^*$.

construct the algorithm based on a few fundamental principles—namely maximal entropy, unbiasedness, maintaining invariance, and, under these constraints, exploiting all available information and solving simple functions reasonably fast.

Surprisingly, the resulting algorithm arrives at (3) and (4): Equations (12) and (51) implement (3) in the manifold of multivariate normal distributions under some monotone transformation of f [1, 5] (let $\eta = 1$, $c_1 = c_\varepsilon = 0$, $c_\mu = \sigma_k = 1$). The monotone transformation is driven by an invariance principle. In both ways, top down and the bottom up, the same, well-recognized stochastic search algorithm *covariance matrix adaptation evolution strategy* (CMA-ES) emerges. Our scrutinizing approach however reveals additional aspects that are consistently useful in practice: cumulation via an evolution path, step-size control, and different learning rates η for different parts of θ. These aspect are either well hidden by (4)[2] or can hardly be derived at all (cumulation). On the downside, the bottom up way is clearly less appealing.

The following sections will introduce and motivate the CMA-ES step-by-step. The CMA-ES samples new solutions from a multivariate normal distribution and updates the parameters of the distribution, namely the mean (incumbent solution), the covariance matrix and additionally a step-size in each iteration, utilizing the f-ranking of the sampled solutions. We formalize the different notions of invariance as well as the maximum likelihood and stationarity properties of the algorithm. A condensed final transcription of the algorithm is provided in the appendix. For a discussion under different perspectives, the reader is referred to [12, 15, 25].

2 Sampling with Maximum Entropy

We start by sampling λ (new) candidate solutions $\boldsymbol{x}_i \in \mathbb{R}^n$, obeying a multivariate normal (search) distribution

$$\boldsymbol{x}_i \sim \boldsymbol{m}_k + \sigma_k \times \mathcal{N}_i(\mathbf{0}, \boldsymbol{C}_k) \qquad \text{for } i = 1, \ldots, \lambda, \tag{5}$$

where $k = 0, 1, 2, \ldots$ is the time or iteration index and $\boldsymbol{m}_k \in \mathbb{R}^n$, $\sigma_k > 0$, and $\mathcal{N}(\mathbf{0}, \boldsymbol{C})$ denotes a multivariate normal distribution with zero mean and covariance matrix $\boldsymbol{C}$, $\sim$ denotes equality in distribution. For convenience, we will sometimes omit the iteration index k.

New solutions obey a multivariate normal distribution with expectation $\boldsymbol{m}$ and covariance matrix $\sigma^2 \times \boldsymbol{C}$. Sets of equal density—that is, lines or surfaces in 2 or 3-D respectively—are ellipsoids centered about the mean and modal value $\boldsymbol{m}$. Figure 2 shows 150 sampled points from a standard (2-variate) normal distribution, $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

[2] Different learning rates might be related to some parameters in the distribution being orthogonal.

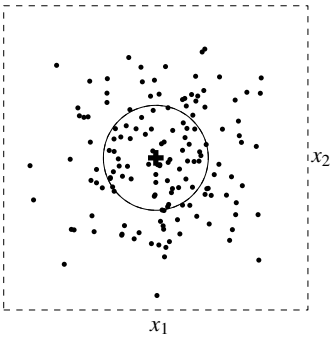

Fig. 2 150 samples from a multivariate (standard) normal distribution in 2-D. Both coordinates are i.i.d. according to a standard normal distribution. The circle depicts the one-σ equal density line, the center of the circle is the mean and modal value at zero. In general, lines of equal density (level sets) are ellipsoids. The probability to sample a point outside the dashed box is close to $1-(1-2\times 0.0015)^2 \approx 1/170$

Given mean, variances and covariances of a distribution, the chosen multivariate normal distribution has **maximum entropy** and—without any further knowledge—suggests itself for randomized search. We explain (5) in more detail.

- The distribution mean value, $\boldsymbol{m}$, is the **incumbent** solution of the algorithm: it is the current estimate for the global optimum provided by the search procedure. The distribution is point symmetrical about the incumbent. The incumbent $\boldsymbol{m}$ is (usually) not evaluated on f. However, it should be evaluated as final solution in the last iteration.
- New solutions are obtained by disturbing $\boldsymbol{m}$ with the *mutation distribution*

$$\mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{C}\right) \equiv \sigma \times \mathcal{N}(\mathbf{0}, \boldsymbol{C}) \ , \tag{6}$$

  where the equivalence holds by definition of $\mathcal{N}(.,.)$. The parameter $\sigma > 0$ is a *step-size* or scale parameter and exists for notational convenience only. The *covariance matrix* $\boldsymbol{C}$ has $\frac{n^2+n}{2}$ degrees of freedom and represents a full quadratic model.
  The covariance matrix determines the *shape* of the distribution, where level-sets of the density are hyper-ellipsoids (confer to [12, 15] for more details). On convex quadratic cost functions, $\boldsymbol{C}$ will closely align with the inverse Hessian of the cost function f (up to a scalar factor). The matrix $\boldsymbol{C}$ defines a *variable neighborhood* metric. The above said suggests that using the maximum entropy distribution with finite variances implies the notion, and underlines the importance of *neighborhood*.

The initial incumbent $\boldsymbol{m}_0$ needs to be provided by the user. The algorithm has no preference for any specific value and its operations are *invariant* to the value of $\boldsymbol{m}_0$ (see translation invariance in Section 4).

Equation (5) implements the principle of **stationarity** or **unbiasedness**, because the expected value of (6) is zero. Improvements are not a priori made *by construction*, but only after sampling *by selection*. In this way, the **least additional assumptions** are built into the search procedure.

The number of candidate solutions sampled in (5) cannot be entirely derived from first principles. For small $\lambda \not\gg n$ the search process will be comparatively local and the algorithm can convergence fast. Only if previously sampled search points are considered, λ could be chosen to its minimal value of one—in particular if the best so-far evaluated candidate solution is always retained. We tend to disregard previous samples entirely (see below). In this case, a selection must take place between $\lambda \geq 2$ new candidate solutions. Because the mutation distribution is unbiased, newly sampled solutions tend to be worse than the previous best solution and in practice $\lambda \geq 5$ is advisable.[3]

On the other hand, for large $\lambda \gg n$, the search becomes more global and the probability to approach the desired, global optimum on multimodal functions is usually larger. On the downside, more function evaluations are necessary to closely approach an optimum even on simple functions.

Consequently, a comparatively successful overall strategy runs the algorithm first with a small population size, e.g. the default $\lambda = 4 + \lfloor 3 \ln n \rfloor$, and afterwards conducts independent restarts with increasing population sizes (IPOP) [6].

After we have established the sampling procedure using a parameterized distribution, we need to determine the *distribution parameters* which are essential to conduct efficient search. All parameters depend explicitly or implicitly on the past and therefore are described in their update equations.

3 Exploiting the Objective Function

The pairs $(\boldsymbol{x}_i, f(\boldsymbol{x}_i))_{i=1,\ldots,\lambda}$, provide the information for choosing a new and better incumbent solution $\boldsymbol{m}_{k+1}$ as well as the new distribution covariance matrix $\sigma^2 \boldsymbol{C}$. Two principles are applied.

1. Old information is disregarded. There are a few reasons to believe that old information can or should be disregarded.
   - The given $(n^2 + 3n)/2$ distribution parameters, $\boldsymbol{m}$ and $\sigma^2 \times \boldsymbol{C}$, should already capture all necessary previous information. Two additional state variables, the search paths $\boldsymbol{p}^\sigma, \boldsymbol{p}^c \in \mathbb{R}^n$, will provide another $2n$ parameters. Theoretical results suggests that only slight improvements can be made by storing and using (all) previously sampled candidate solutions [39, 40], given rank-based selection.

[3] In the (μ, λ)-ES, only the μ best samples are selected for the next iteration. Given $\mu = 1$, a very general optimality condition for λ states that the currently second best solution must resemble the f-value of the previous best solution [22]. Consequently, on any linear function, $\lambda = 2$ and $\lambda = 3$ are optimal [22, 37]. On the sphere function (22), $\lambda = 5$ is optimal [34]. On the latter, also $\lambda \approx 3.7\mu$ can be shown optimal for $\mu \geq 2$ and equal recombination weights [9], compare (12). For $\lambda < 5$, the original strategy parameter setting for CMA-ES has been rectified in [10], but only mirrored sampling leads to satisfactory performance in this case [10].

- Convergence renders previously sampled solutions rather meaningless, because they are too far away from the currently focused region of interest.
- Disregarding old solutions helps to avoid getting trapped in local optima.
- An elitist approach can be destructive in the presence of noise, because a supersolution can stall any further updates. Under uncertainties, any information must be used with great caution.

2. Only the *ranking of the better half* of the new candidate solutions is exploited. Function *values* are discarded as well as the ranking of the worse half of the newly sampled points. Specifically, the function f enters the algorithm only via the indices $i:\lambda$ for $i = 1,\ldots,\mu$, in that (serving as definition for $i:\lambda$)

$$f(\boldsymbol{x}_{1:\lambda}) \le f(\boldsymbol{x}_{2:\lambda}) \le \cdots \le f(\boldsymbol{x}_{\lambda:\lambda}) \tag{7}$$

   is satisfied. We choose $\mu = \lfloor \lambda/2 \rfloor$, because

   a. on a linear function in expectation the better *half* of the new solutions improve over $\boldsymbol{m}_k$ and for the same reason
   b. on the quadratic sphere function only the better half of the new solutions can improve the performance, using positive recombination weights (see (12) below). For the remaining solutions, $\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k$ needs to enter with a negative prefactor [3].

We feel that using worse points to make predictions for the location of better points might make a too strong assumption on the regularity of f in general. Indeed, optimization would be a much easier task if outstandingly bad points would allow generally valid implications on the location of good points, because bad points are generally easy to obtain.

On the highly symmetrical, isotropic sphere model, using the worse half points with the same importance than the better half points for calculating the new incumbent can render the convergence two times faster [2, 3]. In experiments with CMA-ES, we find the factor to be somewhat smaller and obtain very similar results also on the isotropic, highly multimodal Rastrigin function. On most anisotropic functions we observe performance degradations and also failures in rare cases and with noise. The picture though is more encouraging for a covariance matrix update with negative samples, as discussed below.

Because only the f-ranked solution points (rather than the f-values) are used, we denote the f-ranking also as (rank-based) *selection*. The exploitation of available information is quite conservative, reducing the possible ways of deception. As an additional advantage, function values do not need to be available (for example, when optimizing a game playing algorithm, a passably accurate selection and ranking of the μ best current players suffices to proceed to the next iteration). This leads to a strong robustness property of the algorithm: invariance to order-preserving transformations, see next section. The downside of using only the f-ranking is, that the possible convergence speed cannot be faster than linear [7, 28, 39].

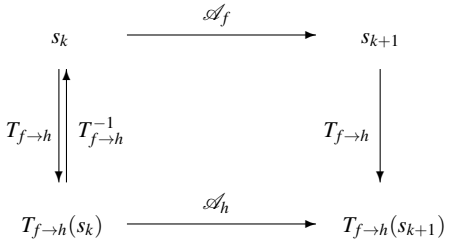

Fig. 3 Commutative diagram for invariance. Vertical arrows depict an invertible transformation (encoding) T of the state variables. Horizontal arrows depict one time step of algorithm $\mathscr{A}$, using the respective function and state variables. The two possible paths between a state at time k and a state at time $k+1$ are equivalent in all (four) cases. The two paths from upper left to lower right are reflected in Equation 8. For $f = h$ the diagram becomes trivial with $T_{f\to h}$ as the identity. One interpretation of the diagram is that given $T_{f\to h}^{-1}$, any function h can be optimized like f

4 Invariance

We begin with a general definition of invariance of a search algorithm $\mathscr{A}$. In short, invariance means that $\mathscr{A}$ does not change its behavior under exchange of f with an equivalent function $h \in \mathscr{H}(f)$, in general conditionally to change of the initial conditions.

Definition 1 (Invariance). Let $\mathscr{H}$ a be mapping from the set of all functions into its power set, $\mathscr{H} : \{\mathbb{R}^n \to \mathbb{R}\} \to 2^{\{\mathbb{R}^n \to \mathbb{R}\}}$, $f \mapsto \mathscr{H}(f)$. Let S be the state space of the search algorithm, $s \in S$ and $\mathscr{A}_f : S \to S$ an iteration step of the algorithm under objective function f. The algorithm $\mathscr{A}$ **is invariant under** $\mathscr{H}$ (in other words: invariant under the exchange of f with elements of $\mathscr{H}(f)$) if for all $f \in \{\mathbb{R}^n \to \mathbb{R}\}$, there exists for all $h \in \mathscr{H}(f)$ a bijective state space transformation $T_{f\to h} : S \to S$ such that for all states $s \in S$

$$\mathscr{A}_h \circ T_{f\to h}(s) = T_{f\to h} \circ \mathscr{A}_f(s) \quad , \tag{8}$$

or equivalently

$$\mathscr{A}_h(s) = T_{f\to h} \circ \mathscr{A}_f \circ T_{f\to h}^{-1}(s) \quad . \tag{9}$$

If $T_{f\to h}$ is the identity for all $h \in \mathscr{H}(f)$, the algorithm is *unconditionally invariant* under $\mathscr{H}$. For randomized algorithms, the equalities hold almost surely, given appropriately coupled random number realizations, otherwise in distribution. The set of functions $\mathscr{H}(f)$ is an invariance set of f for algorithm $\mathscr{A}$.

The simplest example where unconditional invariance trivially holds is $\mathscr{H} : f \mapsto \{f\}$. Any algorithm is unconditionally invariant under the "exchange" of f with f. The idea of invariance is depicted in the commutative diagram in Fig. 3. The two possible paths from the upper left to the lower right are reflected in Equation (8).

Equation (9) implies (trivially) for all $k \in \mathbb{N}$ that

$$\mathscr{A}_h^k(s) = T_{f\to h} \circ \mathscr{A}_f^k \circ T_{f\to h}^{-1}(s) \quad , \tag{10}$$

where $\mathscr{A}^k(s)$ denotes k iteration steps of the algorithm starting from s. Equation (10) reveals that for all $h \in \mathscr{H}(f)$, the algorithm $\mathscr{A}$ optimizes the function h with initial state s just like the function f with initial state $T_{f\to h}^{-1}(s)$. In the lucky scenario, $T_{f\to h}$ is the identity and $\mathscr{A}$ behaves identical on f and h. Otherwise, first s must be moved to $T_{f\to h}^{-1}(s)$, such that *after an adaptation phase* any function h is optimized just like the function f. This is particularly attractive, if f is the easiest function in the invariance class. The adaptation time naturally depends on the distance between s and $T_{f\to h}^{-1}(s)$.

We give the first example of unconditional invariance to order-preserving transformations of f.

Proposition 1 (Invariance to order-preserving transformations). *For all strictly increasing functions $g : \mathbb{R} \to \mathbb{R}$ and for all $f : \mathbb{R}^n \to \mathbb{R}$, the CMA-ES behaves identical on the objective function $\boldsymbol{x} \mapsto f(\boldsymbol{x})$ and the objective function $\boldsymbol{x} \mapsto g(f(\boldsymbol{x}))$. In other words, CMA-ES is unconditionally invariant under*

$$\mathscr{H}_{\text{monoton}} : f \mapsto \{g \circ f \mid g \text{ is stricly increasing}\} \ . \tag{11}$$

Additionally, for each $f : \mathbb{R}^n \to \mathbb{R}$, the set of functions $\mathscr{H}_{\text{monoton}}(f)$—the orbit of f—is an equivalence class of functions with indistinguishable search trace.

Proof idea. Only the f-ranking of solutions is used in CMA-ES and g does not change this ranking. We define the equivalence relation as $f \sim h$ iff $\exists g$ strictly increasing such that $f = g \circ h$. Then, reflexivity, symmetry and transitivity for the equivalence relation $\sim$ can be shown elementarily recognizing that the identity and g^{-1} and compositions of strictly increasing functions are strictly increasing. □

The CMA-ES depends only on the sub-level sets $\{\boldsymbol{x} \mid f(\boldsymbol{x}) \leq \alpha\}$ for $\alpha \in \mathbb{R}$. The monotonous transformation g does not change the sub-level sets, that is $\{\boldsymbol{x} \mid g(f(\boldsymbol{x})) \leq g(\alpha)\} = \{\boldsymbol{x} \mid f(\boldsymbol{x}) \leq \alpha\}$.

5 Update of the Incumbent

Given the restricted usage of information from the evaluations of f, the incumbent is generally updated with a weighted mean of mutation steps

$$\boldsymbol{m}_{k+1} = \boldsymbol{m}_k + c_{\text{m}} \sum_{i=1}^{\mu} w_i \, (\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k) \tag{12}$$

with

$$\sum_{i=1}^{\mu} |w_i| = 1, \quad w_1 \geq w_2 \cdots \geq w_\mu, \quad 0 < c_{\text{m}} \leq 1 \ . \tag{13}$$

The question of how to choose optimal weight values w_i is pursued in [3] and the default values in Table 2 of the appendix approximate the optimal positive values on

the infinite dimensional sphere model. As discussed above, we add the constraints

$$w_\mu > 0 \quad \text{and} \quad \mu \leq \lambda/2\,, \tag{14}$$

while the formulation with (12) also covers more general settings. Usually, we set the learning rate $c_\mathrm{m} = 1$ and the computation of the new incumbent simplifies to

$$\boldsymbol{m}_{k+1} = \sum_{i=1}^{\mu} w_i \boldsymbol{x}_{i:\lambda}\ . \tag{15}$$

A learning rate of one seems to be the largest sensible setting. A value larger than one should only be advantageous, if σ_k is too small and implies that the step-size heuristic should be improved. Very small σ_k together with $c_\mathrm{m} \gg 1$ resemble a classical gradient descent scenario.

The amount of utilized information can be quantified via the *variance effective selection mass*, or *effective* μ

$$\mu_\mathrm{eff} = \left(\sum_{i=1}^{\mu} w_i^2\right)^{-1}\ , \tag{16}$$

where we can easily derive the tight bounds $1 < \mu_\mathrm{eff} \leq \mu$. Usually, a weight setting with $\mu_\mathrm{eff} \approx \lambda/4$ is appropriate. Given μ_eff, the specific choice of the weights is comparatively uncritical. The presented way to update the incumbent using a weighted mean of all μ selected points gives raise for the name $(\mu/\mu_\mathrm{w},\lambda)$-CMA-ES.

Proposition 2 (Random ranking and stationarity of the incumbent). *Under (pure) random ranking, $\boldsymbol{m}_k$ follows an unbiased random walk*

$$\boldsymbol{m}_{k+1} \sim \boldsymbol{m}_k + \frac{\sigma_k}{\sqrt{\mu_\mathrm{eff}}}\mathcal{N}(\boldsymbol{0},\boldsymbol{C}_k) \tag{17}$$

and consequently

$$E(\boldsymbol{m}_{k+1}|\boldsymbol{m}_k) = \boldsymbol{m}_k\ . \tag{18}$$

Pure random ranking means that the index values $i:\lambda \in \{1,\ldots,\lambda\}$ do not depend on $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_\lambda$, for all $i = 1,\ldots,\lambda$, for example when $f(\boldsymbol{x})$ is a random variable with a density and does not depend on $\boldsymbol{x}$, or when $i:\lambda$ is set to i.

Proof idea. Equation (17) follows from Equations (5), (12) and (16), and (18) follows because $E\mathcal{N}(\boldsymbol{0},\boldsymbol{C}) = \boldsymbol{0}$ by definition. □

The proposition affirms, that only selection (f-ranking) can induce a biased movement of the incumbent $\boldsymbol{m}$.

Proposition 3 (Maximum likelihood estimate of the mean). *Given $\boldsymbol{x}_{1:\lambda},\ldots,\boldsymbol{x}_{\mu:\lambda}$, the incumbent $\boldsymbol{m}_{k+1}$ maximizes, independent of the positive definite matrix $\boldsymbol{C}$, the weighted likelihood*

$$\boldsymbol{m}_{k+1} = \operatorname*{arg\,max}_{\boldsymbol{m}\in\mathbb{R}^n} \prod_{i=1}^{\mu} p_{\mathcal{N}}^{w_i}\left(\boldsymbol{x}_{i:\lambda}\,|\,\boldsymbol{m}\right), \tag{19}$$

where $p_{\mathcal{N}}^{w_i}(\boldsymbol{x}\,|\,\boldsymbol{m}) = (p_{\mathcal{N}}(\boldsymbol{x}\,|\,\boldsymbol{m}))^{w_i}$ *and* $p_{\mathcal{N}}(\boldsymbol{x}\,|\,\boldsymbol{m})$ *denotes the density of* $\mathcal{N}(\boldsymbol{m},\boldsymbol{C})$ *at point* $\boldsymbol{x}$*, or equivalently the weighted log-likelihood*

$$\boldsymbol{m}_{k+1} = \operatorname*{arg\,max}_{\boldsymbol{m}\in\mathbb{R}^n} \sum_{i=1}^{\mu} w_i \times \log p_{\mathcal{N}}\left(\boldsymbol{x}_{i:\lambda}\,|\,\boldsymbol{m}\right), \tag{20}$$

Proof idea. We exploit the one-dimensional normal density and the fact that the multivariate normal distribution, after a coordinate system rotation, can be decomposed into n independent mariginal distributions. □

Finally. we find translation invariance, a property that every continuous search algorithm should enjoy.

Proposition 4 (Translation invariance). *The CMA-ES is translation invariant, that is, invariant under*

$$\mathscr{H}_{\text{trans}} : f \mapsto \{h_{\boldsymbol{a}} : \boldsymbol{x} \mapsto f(\boldsymbol{x}-\boldsymbol{a}) \mid \boldsymbol{a} \in \mathbb{R}^n\}, \tag{21}$$

with the bijective state transformation, $T_{f\to h_{\boldsymbol{a}}}$*, that maps* $\boldsymbol{m}$ *to* $\boldsymbol{m}+\boldsymbol{a}$ *(compare Figure 3). In other words, the trace of* $\boldsymbol{m}_k+\boldsymbol{a}$ *is the same for all functions* $h_{\boldsymbol{a}} \in \mathscr{H}_{\text{trans}}$*.*

Proof idea. We consider Fig. 3: an iteration step with state $(\boldsymbol{m}_k, \sigma_k, \boldsymbol{C}_k, \dots)$ using cost function $\boldsymbol{x} \mapsto f(\boldsymbol{x})$ in the upper path is equivalent with an iteration step with state $(\boldsymbol{m}_k+\boldsymbol{a}, \sigma_k, \boldsymbol{C}_k, \dots)$ using cost function $h_{\boldsymbol{a}} : \boldsymbol{x} \mapsto f(\boldsymbol{x}-\boldsymbol{a})$ in the lower path. □

Translation invariance, meaning also that $\boldsymbol{m}_k - \boldsymbol{m}_0$ does not depend on $\boldsymbol{m}_0$, is a rather indispensable property for a search algorithm. Nevertheless, because $\boldsymbol{m}_k$ depends on $\boldsymbol{m}_0$, a reasonable proposition for $\boldsymbol{m}_0$, depending on f, is advisable.

6 Step-Size Control

Step-size control aims to make a search algorithm adaptive to the overall scale of search. Step-size control allows for fast convergence to an optimum and serves to satisfy the following basic demands on a search algorithm.

1. Solving linear functions, like $f(\boldsymbol{x}) = x_1$. On linear functions we desire a geometrical increase of the f-gain $f(\boldsymbol{m}_k) - f(\boldsymbol{m}_{k+1})$ with increasing k.
2. Solving the simplest convex-quadratic function, the sphere function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} (x_i - x_i^*)^2 = \|\boldsymbol{x}-\boldsymbol{x}^*\|^2 \ , \tag{22}$$

fast. We desire

$$\frac{\|\boldsymbol{m}_k - \boldsymbol{x}^*\|}{\|\boldsymbol{m}_0 - \boldsymbol{x}^*\|} \approx \exp\left(-c\frac{k}{n}\right) \quad , \tag{23}$$

such that $c \ll 0.02\min(n, \lambda)$, because $c \approx 0.25\,\lambda$ is the optimal value which can be achieved with optimal step-size and optimal positive weights for $\lambda \gg n$ ($c \approx 0.5\,\lambda$ can be achieved using also negative weights for $\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k$ in (12), see [3]). The optimal step-size changes when approaching the optimum.

Additionally, step-size control will provide scale invariance, as explicated below.

Unfortunately, step-size control can hardly be derived from first principles and therefore relies on some internal model or some heuristics. Line-search is one such heuristic that decides on the realized step length *after* the direction of the step is given. Surprisingly, a line-search can gain very little over a fixed (optimal) step length given in each iteration [27]. Recent theoretical results even seem to indicate that in the limit for $n \to \infty$ the optimal progress rate cannot be improved at all by a cost-free ray search on a half-line (given positive weights) or by a line search otherwise [30]. A few further heuristics for step-size control are well-recognized.

1. Controlling the success rate of new candidate solutions, compared to the best solution seen so far (1/5-th success rule) [34, 36].
2. Sampling different candidate solutions with different step-sizes (self-adaptation) [34, 37]. Selected solutions also retain their step-size.
3. Testing different step-sizes by conducting additional test steps in direction $\boldsymbol{m}_{k+1} - \boldsymbol{m}_k$, resembling a rudimentary line-search (two point adaptation) [18, 35].
4. Controlling the length of the search path, taken over a number of iterations (*cumulative step-size adaptation*, CSA, or *path length control*) [33].

In our context, the last two approaches find reasonable values for σ in simple test cases (like ridge topologies).

We use cumulative step-size adaptation here. The underlying design principle is to achieve perpendicularity of successive steps. Perpendicularity is measured using an *evolution path* and a *variable metric*.

Conceptually, an **evolution path**, or search path, of length j is the vector

$$\boldsymbol{m}_k - \boldsymbol{m}_{k-j} \quad , \tag{24}$$

that is, the total displacement of the mean during j iterations. For technical convenience, and in order to satisfy the stationary condition (26), we compute the search path, $\boldsymbol{p}^\sigma$, in an iterative momentum equation with the initial path $\boldsymbol{p}_0^\sigma = \boldsymbol{0}$ as

$$\boldsymbol{p}_{k+1}^\sigma = (1 - c_\sigma)\boldsymbol{p}_k^\sigma + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\mathrm{eff}}}\, \boldsymbol{C}_k^{-\frac{1}{2}} \frac{\boldsymbol{m}_{k+1} - \boldsymbol{m}_k}{\sigma_k} \, . \tag{25}$$

The factor $1 - c_\sigma > 0$ is the decay weight and $1/c_\sigma \approx n/3$ is the backward time horizon—after $1/c_\sigma$ iterations about $1 - \exp(-1) \approx 63\%$ of the information has been replaced; $\boldsymbol{C}_k^{-\frac{1}{2}}$ is the positive symmetric square root[4] of $\boldsymbol{C}_k^{-1}$. The remaining

[4] The positive symmetric square root satisfies $\boldsymbol{C}_k^{-\frac{1}{2}} \boldsymbol{C}_k^{-\frac{1}{2}} = \boldsymbol{C}_k^{-1}$, has only positive eigenvalues and is unique.

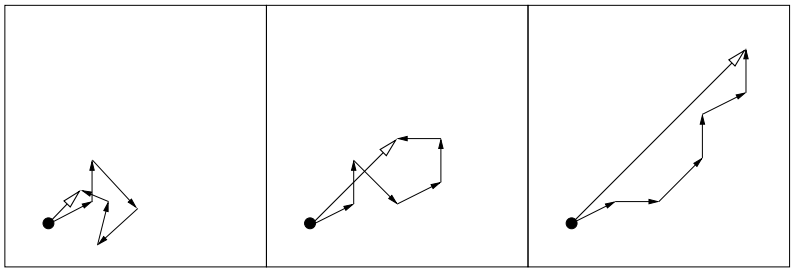

Fig. 4 Schematic depiction of three evolution paths in the search space (each with six successive steps of $\boldsymbol{m}_k$). Left: single steps cancel each other out and the evolution path is short. Middle: steps are "on average orthogonal". Right: steps are positively correlated and the evolution path is long. The length of the path is a good indicator for optimality of the step-size

factors are, without further degree of freedom, chosen to guaranty the stationarity

$$\boldsymbol{p}_k^\sigma \sim \mathcal{N}(\mathbf{0},\mathbf{I}) \quad \text{for } k = 1,2,3,\ldots\,, \tag{26}$$

given $\boldsymbol{p}_0^\sigma \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ and pure random ranking of $\boldsymbol{x}_{i:\lambda}$ in all preceding time steps.

The length of the evolution path is used to update the step-size σ either following [29]

$$\sigma_{k+1} = \sigma_k \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_{k+1}^\sigma\|^2 - n}{2n}\right)\right) \tag{27}$$

or via

$$\sigma_{k+1} = \sigma_k \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_{k+1}^\sigma\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right)\,, \tag{28}$$

where $d_\sigma \approx 1$. The step-size increases/decreases *iff* $\|\boldsymbol{p}_{k+1}^\sigma\|^2$ or $\|\boldsymbol{p}_{k+1}^\sigma\|$ is larger/smaller than its expected value. Equation (27) is more appealing and easier to analyze, but (28) might have an advantage in practice. In practice, also an upper bound to the argument of exp is sometimes useful.

Figure 4 depicts the idea of the step-size control schematically.

- If steps are positively correlated, the evolution path tends to be long (right picture). A similar trajectory could be covered by fewer but longer steps and the step-size is increased.
- If steps are negatively correlated they tend to cancel each other out and the evolution path is short (left picture). Shorter steps seem more appropriate and the step-size is decreased.
- If the f-ranking does not affect the length of the evolution path, the step-size is unbiased (middle picture).

We note two major postulates related to step-size control and two major design principles of the step-size update.

Postulate 1 (Conjugate steps) *Successive iteration steps should be approximately $\boldsymbol{C}^{-1}$-conjugate, that is, orthogonal with respect to the inner product (and metric) defined by $\boldsymbol{C}^{-1}$.*

As a consequence of this postulate, we have used perpendicularity as optimality criterion for step-size control.

If steps are uncorrelated, like under random selection, they indeed become approximately $\boldsymbol{C}^{-1}$-conjugate, that is $(\boldsymbol{m}_{k+1} - \boldsymbol{m}_k)^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{m}_k - \boldsymbol{m}_{k-1} \approx 0$, see [15]. This means the steps are orthogonal with respect to the inner product defined by $\boldsymbol{C}^{-1}$ and therefore orthogonal in the coordinate system defined by $\boldsymbol{C}$. In this coordinate system, the coordinate axes, where the independent sampling takes place, are eigenvectors of $\boldsymbol{C}$. Seemingly uncorrelated steps are the desired case and achieved by using $\boldsymbol{C}^{-1/2}$ in (25).

In order to better understand the following assertions, we rewrite the step-size update (28), only using an *additive* update term,

$$\log \sigma_{k+1} = \log \sigma_k + \frac{c_\sigma}{d_\sigma} \left(\frac{\|\boldsymbol{p}_{k+1}^\sigma\|}{\mathsf{E}\|\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})\|} - 1 \right) . \tag{29}$$

First, in accordance with our stationary design principle, we establish a stationarity condition on the step-size.

Proposition 5 (Stationarity of step-size). *Given pure random ranking and $\boldsymbol{p}_0^\sigma \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, the quantity $\log \sigma_k$ performs an unbiased random walk (see Equation (29)). Consequently, the step-size obeys the stationarity condition*

$$E(\log \sigma_{k+1} | \sigma_k) = \log \sigma_k . \tag{30}$$

Proof idea. We analyze the update equations (29) and (25). □

Postulate 2 (Behavior on linear functions [14]) *On a linear function, the dispersion of new candidate solutions should increase geometrically fast in the iteration sequence, that is, linearly on the log scale. Given σ_k^β as dispersion measure with $\beta > 0$, we can set w.l.o.g. $\beta = 1$ and demand for some $\alpha > 0$*

$$E(\log \sigma_{k+1} | \sigma_k) \geq \log \sigma_k + \alpha . \tag{31}$$

The CMA-ES satisfies the postulate for some k_0 and all $k \geq k_0$, because on a linear function the expected length of the evolution path increases monotonically. We reckon that $k_0 \propto 1/c_\sigma$. Finally, we investigate the more abstract conception of scale invariance as depicted in Fig. 5.

Proposition 6 (Scale invariance). *The CMA-ES is invariant under*

$$\mathscr{H}_{\mathrm{scale}} : f \mapsto \{ h_\alpha : \boldsymbol{x} \mapsto f(\boldsymbol{x}/\alpha) \mid \alpha > 0 \} \tag{32}$$

with the associated bijective state space transformation

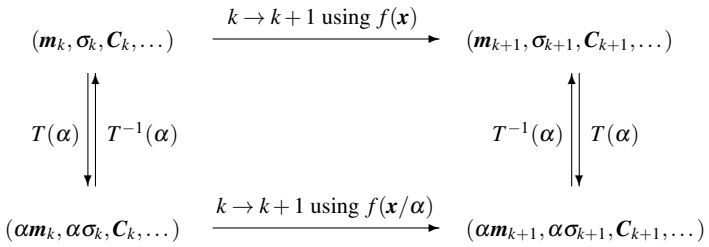

Fig. 5 Commutative diagram for scale invariance. Vertical arrows depict an invertible transformation (encoding) T of all state variables of CMA-ES with $T(\alpha) : (\boldsymbol{m}, \sigma, \boldsymbol{C}, \boldsymbol{p}^\sigma, \boldsymbol{p}^\mathrm{c}) \mapsto (\alpha\boldsymbol{m}, \alpha\sigma, \boldsymbol{C}, \boldsymbol{p}^\sigma, \boldsymbol{p}^\mathrm{c})$. Horizontal arrows depict one time step of CMA-ES, applied to the respective tuple of state variables. The two possible paths between a state at time k and a state at time $k+1$ are equivalent in all (four) cases. For $\alpha = 1$ the diagram becomes trivial. The diagram suggests that CMA-ES is invariant under the choice of $\alpha > 0$ in the sense that, given T and T^{-1} were available, any function $\boldsymbol{x} \mapsto f(\alpha\boldsymbol{x})$ is (at least) as easy to optimization as f

$$T : (\boldsymbol{m}, \sigma, \boldsymbol{C}, \boldsymbol{p}^\sigma, \boldsymbol{p}^\mathrm{c}) \mapsto (\alpha\boldsymbol{m}, \alpha\sigma, \boldsymbol{C}, \boldsymbol{p}^\sigma, \boldsymbol{p}^\mathrm{c}) \,.$$

That means for all states $(\boldsymbol{m}_k, \sigma_k, \boldsymbol{C}_k, \boldsymbol{p}_k^\sigma, \boldsymbol{p}_k^\mathrm{c})$

$$\mathit{CMA\text{-}ES}_h(T(\boldsymbol{m}_k, \sigma_k, \boldsymbol{C}_k, \boldsymbol{p}_k^\sigma, \boldsymbol{p}_k^\mathrm{c})) = T(\mathit{CMA\text{-}ES}_f(\underbrace{\boldsymbol{m}_k, \sigma_k, \boldsymbol{C}_k, \boldsymbol{p}_k^\sigma, \boldsymbol{p}_k^\mathrm{c}}_{= T^{-1}(T(\boldsymbol{m}_k, \sigma_k, \boldsymbol{C}_k, \boldsymbol{p}_k^\sigma, \boldsymbol{p}_k^\mathrm{c}))})) \,, \tag{33}$$

see Fig. 5. Furthermore, for any given $f : \mathbb{R}^n \to \mathbb{R}$*, the set of functions* $\mathscr{H}_\mathrm{scale}(f)$*—the orbit of* f*—is an equivalence class.*

Proof idea. We investigate the update equations of the state variables comparing the two possible paths from the lower left to the lower right in Fig. 5. The equivalence relation property can be shown elementarily (compare Proposition 1) or using the property that the set $\{\alpha > 0\}$ is a transformation group over the set $\{h : \mathbb{R}^n \to \mathbb{R}\}$ and therefore induces the equivalence classes $\mathscr{H}_\mathrm{scale}(f)$ (see also Proposition 9). □

Invariance allows to draw the commutative diagram of Fig. 5. Scale invariance can be interpreted in several ways.

- The choice of scale α is irrelevant for the algorithm, that is, the algorithm has no intrinsic (built-in) notion of scale.
- The transformation T in Fig. 5 is a change of coordinate system (here: change of scale) and the update equations are independent of the actually chosen coordinate system, that is, they could be formulated in an algebraic way.
- For functions in the equivalence class $\mathscr{H}_\mathrm{scale}(f)$, the trace of the algorithm $(\alpha\boldsymbol{m}_k, \alpha\sigma_k, \boldsymbol{C}_k, \boldsymbol{p}_k^\sigma, \boldsymbol{p}_k^\mathrm{c})$ will be identical for all $k = 0, 1, 2 \dots$, given that $\boldsymbol{m}_0$ and σ_0 are chosen appropriately, for example $\sigma_0 = 1/\alpha$ and $\boldsymbol{m}_0 = \sigma_0 \times \boldsymbol{a}$. Then the trace for $k = 0$ equals $(\alpha\boldsymbol{m}_0, \alpha\sigma_0, \boldsymbol{C}_0, \dots) = (\boldsymbol{a}, 1, \boldsymbol{C}_0, \dots)$ and the trace does not depend on α for any $k \geq 0$.

- From the last point follows that the step-size control has a distinct role in scale invariance. In practice, when α is unknown, adaptation of the step-size that achieves $\sigma_k \propto 1/\alpha$ can render the algorithm virtually independent of α.

Scale invariance and step-size control also facilitate the possibility of **linear convergence** in k to the optimum $\boldsymbol{x}^*$, in that

$$\lim_{k\to\infty} \sqrt[k]{\frac{\|\boldsymbol{m}_k - \boldsymbol{x}^*\|}{\|\boldsymbol{m}_0 - \boldsymbol{x}^*\|}} = \exp\left(-\frac{c}{n}\right) \tag{34}$$

exists with $c > 0$ or equivalently

$$\begin{aligned}\lim_{k\to\infty} \frac{1}{k} \log \|\boldsymbol{m}_k - \boldsymbol{x}^*\| &= \lim_{k\to\infty} \frac{1}{k} \log \frac{\|\boldsymbol{m}_k - \boldsymbol{x}^*\|}{\|\boldsymbol{m}_0 - \boldsymbol{x}^*\|} \\ &= \lim_{k\to\infty} \frac{1}{k} \sum_{k=1}^{t} \log \frac{\|\boldsymbol{m}_k - \boldsymbol{x}^*\|}{\|\boldsymbol{m}_{k-1} - \boldsymbol{x}^*\|} \\ &= -\frac{c}{n}\end{aligned} \tag{35}$$

and similarly

$$E\left(\log \frac{\|\boldsymbol{m}_{k+1} - \boldsymbol{x}^*\|}{\|\boldsymbol{m}_k - \boldsymbol{x}^*\|}\right) \to -\frac{c}{n} \quad \text{for } k \to \infty\,. \tag{36}$$

Hence, c denotes a convergence rate and for $c > 0$ the algorithm converges "log-linearly" (in other words, geometrically fast) to the optimum.

In the beginning of this section we had stated two basic demands on a search algorithm, step-size control is meant to address, namely solving linear functions and the sphere function appropriately fast. We now pursue, with a single experiment, whether the demands are satisfied.

Figure 6 shows a run on the objective function $f : \mathbb{R}^n \to \mathbb{R}, \boldsymbol{x} \mapsto \|\boldsymbol{x}\|$, with $n = 20$, $\lambda = 12$ (the default value, see Table 2) and with $\sigma_0 = 10^{-9}$ chosen far too small given that $\boldsymbol{m}_0 = \mathbf{1}$. The outcome when repeating this experiment looks always very similar. We discuss the demands in turn.

1. During the first 170 iterations the algorithm virtually "observes" the linear function $\boldsymbol{x} \mapsto \sum_{i=1}^{20} x_i$ at point $\mathbf{1} \in \mathbb{R}^{20}$. We see during this phase that σ increases geometrically fast (linearly on the log scale). From this observation, and the invariance properties of the algorithm (also rotation invariance, see below), we can safely imply that the demand for linear functions is satisfied.
2. After the adaptation of σ after about 180 iterations, linear convergence to the optimum can be observed. We compute the convergence rate between iteration 180 and 600 from the graph. Starting with $\frac{\|\boldsymbol{m}_k\|}{\|\boldsymbol{m}_0\|} \approx \exp\left(-c\frac{k}{n}\right)$ from (23) we replace $\boldsymbol{m}_0$ with $\boldsymbol{m}_{180}$ and compute

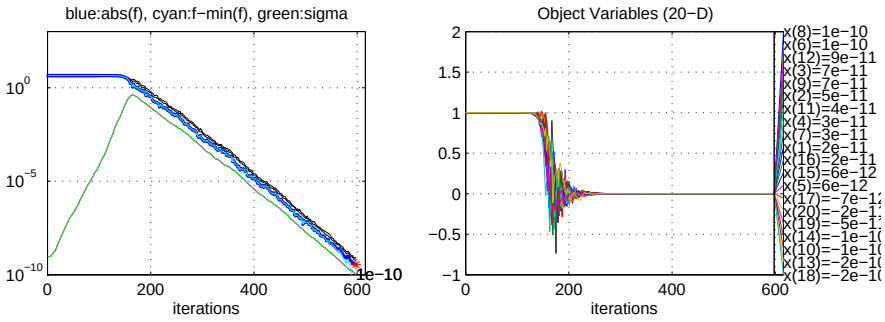

Fig. 6 A run of CSA-ES (Equations (5), (15), (25) and (28)) on the objective function $f : \mathbb{R}^{20} \to \mathbb{R}, \boldsymbol{x} \mapsto \|\boldsymbol{x}\|$, as a member of the equivalence class of functions $\boldsymbol{x} \mapsto g(\|\alpha \boldsymbol{x} - \boldsymbol{x}^*\|)$ with identical behavior, given $\sigma_0 \propto 1/\alpha$ and $\boldsymbol{m}_0 = \sigma_0 \times (\text{const} + \boldsymbol{x}^*)$. Here, $\boldsymbol{m}_0 = \mathbf{1}$ and the inital step-size $\sigma_0 = 10^{-9}$ is chosen far too small. Left: $f(\boldsymbol{m}_k)$ (think blue graph) and σ_k versus iteration number k in a semi-log plot. Right: all components of $\boldsymbol{m}_k$ versus k.

$$\frac{\|\boldsymbol{m}_{k=600}\|}{\|\boldsymbol{m}_{k=180}\|} \approx \frac{10^{-9.5}}{10^{0}} \approx \exp\left(-c\frac{600-180}{20}\right) . \tag{37}$$

Solving for c yields $c \approx 1.0$ and with $\min(n,\lambda) = \lambda = 12$ we get $c \approx 1.0 \not\ll 0.24 = 0.02 \min(n,\lambda)$. Our demand on the convergence rate c is more than satisfied. The same can be observed when covariance matrix adaptation is applied additionally (not shown).

The demand on the convergence (23) can be rewritten in that

$$\log \|\boldsymbol{m}_k - \boldsymbol{x}^*\| \approx -c\frac{k}{n} + \text{const} . \tag{38}$$

The k in the RHS nominator implies *linear convergence* in the number of iterations. The n in the denominator implies **linear scale-up**: the number of iterations to reduce the distance to the optimum by a given factor increases linearly with the dimension n. Linear *convergence* can also be achieved with covariance matrix adaptation. Given $\lambda \not\gg n$, linear *scale-up* cannot be achieved with covariance matrix adaptation alone, because a reliable setting for the learning rate for the covariance matrix is $o(1/n)$. However, step-size control is reliable and achieves linear scale-up given the step-size damping parameter $d_\sigma = O(1)$ in (28). Scale-up experiments are inevitable to support this claim and have been done, for example, in [25].

7 Covariance Matrix Adaptation

In the remainder we exploit the f-ranked (i.e. selected and ordered) set $(\boldsymbol{x}_{1:\lambda}, \ldots, \boldsymbol{x}_{\mu:\lambda})$ to update the covariance matrix $\boldsymbol{C}$. First, we note that the covariance matrix

represents *variation* parameters. Consequently, an apparent principle is to encourage, or *reinforce variations that have been successful*—just like successful candidate solutions are reinforced in the update of $\boldsymbol{m}$ in (15). Based on the current set of f-ranked points, the successful variations are (by definition)

$$\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k \quad \text{for } i = 1, \ldots, \mu \,. \tag{39}$$

Remark that "successful variation" does not imply $f(\boldsymbol{x}_{i:\lambda}) < f(\boldsymbol{m}_k)$ which is neither necessary nor important nor even desirable in general. Even the demand $f(\boldsymbol{x}_{1:\lambda}) < f(\boldsymbol{m}_k)$ would often result in a far too small step-size.

7.1 The Rank-μ Matrix

From the successful variations in (39) we form a covariance matrix

$$\boldsymbol{C}_{k+1}^{\mu} = \sum_{i=1}^{\mu} w_i \frac{\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k}{\sigma_k} \times \frac{\left(\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k\right)^{\mathrm{T}}}{\sigma_k} \,. \tag{40}$$

Equation (40) is analogous to (15) where successful solution points are used to form the new incumbent. We can easily derive the condition

$$E(\boldsymbol{C}_{k+1}^{\mu} | \boldsymbol{C}_k) = \boldsymbol{C}_k \tag{41}$$

under pure random ranking thus explaining the factors $1/\sigma_k$ in (40).

Assuming the weights w_i as given, the matrix $\boldsymbol{C}_{k+1}^{\mu}$ maximizes the (weighted) likelihood of the f-ranked steps.

Proposition 7 (Maximum likelihood estimate of $\boldsymbol{C}$). *Given $\mu \geq n$, the matrix $\boldsymbol{C}_{k+1}^{\mu}$ maximizes the weighted log-likelihood*

$$\boldsymbol{C}_{k+1}^{\mu} = \underset{\boldsymbol{C} \ pos\ def}{\arg\max} \sum_{i=1}^{\mu} w_i \times \log p_{\mathcal{N}} \left(\frac{\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k}{\sigma_k} \middle| \boldsymbol{C} \right) \,, \tag{42}$$

where $p_{\mathcal{N}}(\boldsymbol{x} \,|\, \boldsymbol{C})$ denotes the density of $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C})$ at point $\boldsymbol{x}$, and therefore the RHS of (42) reads more explicitly

$$\underset{\boldsymbol{C} \ pos\ def}{\arg\max} \left(-\frac{1}{2} \log\det(\alpha \boldsymbol{C}) - \frac{1}{2\sigma_k{}^2} \sum_{i=1}^{\mu} w_i (\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k)^{\mathrm{T}} \boldsymbol{C}^{-1} (\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k) \right) \tag{43}$$

where $\alpha = 2\pi\sigma_k{}^2$ is irrelevant for the result.

Proof idea. The proof is non-trivial but works similar to the classical non-weighted case. □

In contrast to the computation of $\boldsymbol{m}$ in (12), we are not aware of a derivation for optimality of certain weight values in (40). Future results might reveal that different weights and/or even a different value for μ are desirable for (12) and (40). Before we turn finally to the covariance matrix update, we scrutinize the computation of $\boldsymbol{C}_{k+1}^{\mu}$.

What is missing?

In Section 3 we have argued to use only the μ best solutions from the last iteration to update distribution parameters. For a covariance matrix update, disregarding the worst solutions might be too conservative and a negative update of the covariance matrix with the μ worst solutions is proposed in [29]. This idea is not accommodated in this chapter, but has been recently exploited with consistently good results [4, 26]. An inherent inconsistency with negative updates though is that long steps tend to be worse merely because they are long (and not because they represent a bad direction) meanwhile, unfortunately, long steps also lead to stronger updates.

At first sight we might believe to have covered all variation information given by $\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k$ in the covariance matrix $\boldsymbol{C}_{k+1}^{\mu}$. On closer inspection we find that the outer product in (40) removes the sign: using $-(\boldsymbol{x}_{i:\lambda} - \boldsymbol{m})$ instead of $\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}$ in (40) yields the same $\boldsymbol{C}_{k+1}^{\mu}$. One possibility to recover the sign information is to favor the direction $\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}$ over $-(\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}) = \boldsymbol{m}_k - \boldsymbol{x}_{i:\lambda}$ in some way. This seems difficult to accomplish without affecting either the distribution mean (interfering with Proposition 3) or the maximum entropy property. Therefore, we choose a different way to recover the sign information.

7.2 Another Evolution Path

We recover the sign information in a classical and rather heuristic way, which turns out to be nevertheless quite effective. We consider an evolution path $\boldsymbol{x} - \boldsymbol{m}_{k-j}$ for $j > 0$, where $\boldsymbol{x}$ might be $\boldsymbol{m}_{k+1}$ or any $\boldsymbol{x}_{i:\lambda}$. We decompose the path into the recent step and the old path

$$\boldsymbol{x} - \boldsymbol{m}_{k-j} = \boldsymbol{x} - \boldsymbol{m}_k + \boldsymbol{m}_k - \boldsymbol{m}_{k-j} \quad . \tag{44}$$

Switching the sign of the last step means using the vector $\boldsymbol{m}_k - \boldsymbol{x}$ instead of $\boldsymbol{x} - \boldsymbol{m}_k$ and we get in this case

$$\begin{aligned} \boldsymbol{m}_k - \boldsymbol{x} + \boldsymbol{m}_k - \boldsymbol{m}_{k-j} &= 2(\boldsymbol{m}_k - \boldsymbol{x}) + \boldsymbol{x} - \boldsymbol{m}_{k-j} \\ &= \boldsymbol{x} - \boldsymbol{m}_{k-j} - 2(\boldsymbol{x} - \boldsymbol{m}_k) \, . \end{aligned} \tag{45}$$

Comparing the last line with the LHS of (44), we see that now the sign of the recent step matters. Only in the trivial cases, if either $\boldsymbol{x} = \boldsymbol{m}_k$ (zero step) or $\boldsymbol{m}_k = \boldsymbol{m}_{k-j}$ (pre-

vious zero path) the outer products of (44) and (45) are identical. Because we will compute the evolution path over a considerable number of iterations j, the specific choice for $\boldsymbol{x}$ should become rather irrelevant and we will use $\boldsymbol{m}_{k+1}$ in the following.

In practice, we compute the evolution path, analogous to (25). We set $\boldsymbol{p}_0^{\mathrm{c}} = \mathbf{0}$ and use the momentum equation

$$\boldsymbol{p}_{k+1}^{\mathrm{c}} = (1-c_{\mathrm{c}})\boldsymbol{p}_k^{\mathrm{c}} + h_\sigma \sqrt{c_{\mathrm{c}}(2-c_{\mathrm{c}})\mu_{\mathrm{eff}}}\, \frac{\boldsymbol{m}_{k+1}-\boldsymbol{m}_k}{\sigma_k}\,, \tag{46}$$

where $h_\sigma = 1$ if $\|\boldsymbol{p}_{k+1}^{\sigma}\|^2 < \left(1-(1-c_\sigma)^{2(k+1)}\right)\left(2+\frac{2}{n+1}\right)n$ and zero otherwise; h_σ stalls the update whenever $\|\boldsymbol{p}_{k+1}^{\sigma}\|$ is large. The implementation of h_σ supports the judgment of pursuing a heuristic rather than a first principle here, and is driven by two considerations.

1. Given a fast increase of the step-size (induced by the fact that $\|\boldsymbol{p}_{k+1}^{\sigma}\|$ is large), the "visible" landscape will change fast and the adaptation of the covariance matrix to the current landscape seems inappropriate, in particular, because
2. the covariance matrix update using $\boldsymbol{p}^{\mathrm{c}}$ is asymmetric: a large variance in a single direction can be *introduced* fast (while $\|\boldsymbol{p}_{k+1}^{\mathrm{c}}\|$ is large), but the large variance can only be *removed* on a significantly longer time scale. For this reason in particular, an unjustified update should be avoided.

While in (46), again, $1-c_{\mathrm{c}}$ is the decay factor and $1/c_{\mathrm{c}} \approx (n+4)/4$, the remaining constants are determined by the stationarity condition

$$\boldsymbol{p}_{k+1}^{\mathrm{c}} \sim \boldsymbol{p}_k^{\mathrm{c}}\,, \tag{47}$$

given $\boldsymbol{p}_k^{\mathrm{c}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{C}_k)$ and pure random ranking and $h_\sigma \equiv 1$.

The evolution path $\boldsymbol{p}^{\mathrm{c}}$ heavily exploits the sign information. Let us consider, for a given $\boldsymbol{y} \in \mathbb{R}^n$, two hypothetical situations with $\boldsymbol{m}_{k+1} - \boldsymbol{m}_k = \alpha^k \boldsymbol{y}$, for $k = 0, 1, 2, \ldots$. We find that for $k \to \infty$

$$\text{if } \alpha^k = 1 \text{ then } \boldsymbol{p}_k^{\mathrm{c}} \to \sqrt{\frac{2-c_{\mathrm{c}}}{c_{\mathrm{c}}}}\,\boldsymbol{y} \approx \sqrt{\frac{n+2}{2}}\,\boldsymbol{y} \tag{48}$$

$$\text{if } \alpha^k = (-1)^k \text{ then } \boldsymbol{p}_k^{\mathrm{c}} \to (-1)^{k-1}\sqrt{\frac{c_{\mathrm{c}}}{2-c_{\mathrm{c}}}}\,\boldsymbol{y} \approx (-1)^{k-1}\sqrt{\frac{2}{n+2}}\,\boldsymbol{y}\,. \tag{49}$$

Both equations follow from solving the stationarity condition $x = (1-c_{\mathrm{c}}) \times (\pm x) + \sqrt{c_{\mathrm{c}}(2-c_{\mathrm{c}})}$ for x. Combining both equations, we get the ratio between maximal and minimal possible length of $\boldsymbol{p}^{\mathrm{c}}$, given the input vectors have constant length, as

$$\frac{2-c_{\mathrm{c}}}{c_{\mathrm{c}}} \approx \frac{n+2}{2}\,. \tag{50}$$

Additionally to the matrix $\boldsymbol{C}_{k+1}^{\mu}$, we use the rank-one matrix $\boldsymbol{p}_{k+1}^{\mathrm{c}}{\boldsymbol{p}_{k+1}^{\mathrm{c}}}^{\mathrm{T}}$ to introduce the missing sign information into the covariance matrix. The update is specified be-

low in (51). The update implements the principal heuristic of reinforcing successful variations for variations observed over several iterations.

Evaluation of the Cumulation Heuristic

We evaluate the effect of the evolution path for covariance matrix adaptation. Figure 7 shows running length measurements of the $(\mu/\mu_{\mathrm{w}},\lambda)$-CMA-ES depending on the choice of c_{c} on the cigar function (see legend). The graphs in the left plot are typical example data to identify a good parameter setting. Ten values for c_{c}^{-1} between 1 and $10n$ are shown for each dimension. Larger values are not regarded as sensible. The setting $c_{\mathrm{c}} = 1$ means that the heuristic is switched off. Improvements over the setting $c_{\mathrm{c}} = 1$ can be observed in particular for larger dimensions, where, up to $n = 100$, the function can be solved up to ten times faster. For $c_{\mathrm{c}}^{-1} = n$ the performance is for all dimensions close to optimal.

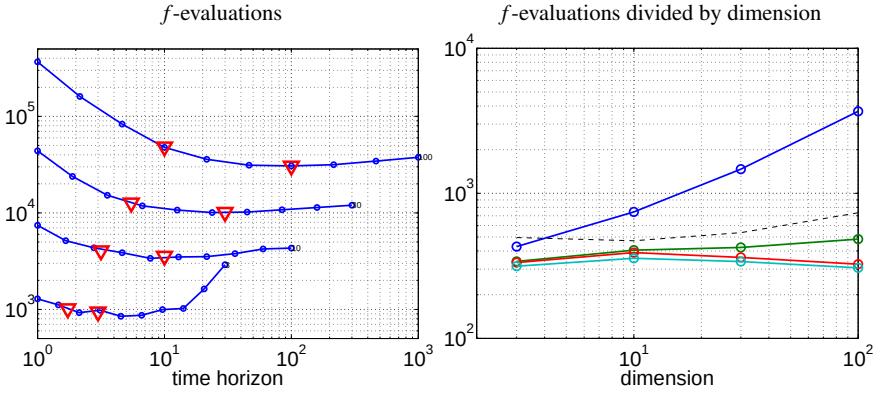

Fig. 7 Number of function evaluations to reach $f(\boldsymbol{x}) < 10^{-6}$ on $f(\boldsymbol{x}) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$ with $\boldsymbol{m}_0 = \boldsymbol{1}$ and $\sigma_0 = 1$. For a (backward) time horizon of $c_{\mathrm{c}}^{-1} = 1$, the cumulation heuristic is, by definition, switched off. Left figure: number of function evaluations, where each point represents a single run, plotted versus the backward time horizon of the evolution path, c_{c}^{-1}, for $n = [3;10;30;100]$ (from bottom to top). Triangles show averages for $c_{\mathrm{c}}^{-1} = \sqrt{n}$ and n, also shown on the right. Right figure: average number of function evaluations divided by n, from $[10;3;2;1] = \lfloor 10/\lfloor\sqrt{n}\rfloor\rfloor$ runs, plotted versus n for (from top to bottom) $c_{\mathrm{c}}^{-1} = 1; \sqrt{n}; \frac{n+3}{3}; n$. Compared to $c_{\mathrm{c}} = 1$, the speed-up exceeds in all cases a factor of $\sqrt{n}/2$ (dashed line).

The right plot shows the running lengths for four different parameter settings versus dimension. For $n = 3$ the smallest speed-up of about 25% is observed for all variants with $c_{\mathrm{c}}^{-1} > 1$. The speed-up grows to a factor of roughly 2, 4, and 10 for dimensions 10, 30, and 100, respectively, and always exceeds a factor of $\sqrt{n}/2$. For $c_{\mathrm{c}} = 1$ (heuristic off) the scaling with the dimension is $\approx n^{1.7}$. For $c_{\mathrm{c}}^{-1} = \sqrt{n}$ the scaling becomes $\approx n^{1.1}$ and about linear for $c_{\mathrm{c}}^{-1} \geq n/3$. These findings hold for any function, where the predominant task is to acquire the orientation of a constant number of "long axes", in other words to find a few insensitive directions, where yet

a large distance needs to be traversed. The assertion in [38] that $c_c^{-1} \propto n$ is needed to get a significant scaling improvement turns out to be wrong. For larger population sizes λ, where the rank-μ update becomes more effective, the positive effect reduces and almost vanishes with $\lambda = 10n$.

The same experiment has been conducted on other (unimodal) functions. While on many functions the cumulation heuristic is less effective and yields only a rather n-independent and small speed-up (e.g. on the Rosenbrock function somewhat below a factor of two), we have not seen an example yet, where it compromises the performance remarkably. Hence the default choice has become $c_c^{-1} \approx n/4$ (see Table 2 in the appendix), because (a) the update for the covariance matrix will have a time constant of $c_1^{-1} \approx n^2/2$ and we feel that c_1^{-1}/c_c^{-1} should not be smaller than n, and (b) in our additional experiments the value $c_c^{-1} = n$ is indeed sometimes worse than smaller values.

7.3 The Covariance Matrix Update

The final covariance matrix update combines a rank-one update using $\boldsymbol{p}^{\mathrm{c}}\boldsymbol{p}^{\mathrm{c}\mathrm{T}}$ and a rank-μ update using $\boldsymbol{C}_{k+1}^{\mu}$,

$$\boldsymbol{C}_{k+1} = (1 - c_1 - c_\mu + c_\varepsilon)\,\boldsymbol{C}_k + c_1 \boldsymbol{p}_{k+1}^{\mathrm{c}} {\boldsymbol{p}_{k+1}^{\mathrm{c}}}^{\mathrm{T}} + c_\mu \boldsymbol{C}_{k+1}^{\mu}\,, \tag{51}$$

where $\boldsymbol{p}^{\mathrm{c}}$ and $\boldsymbol{C}_{k+1}^{\mu}$ are defined in (46) and (40) respectively, and $c_\varepsilon = (1 - {h_\sigma}^2)\, c_1 c_c (2 - c_c)$ is of minor relevance and makes up for the loss of variance in case of $h_\sigma = 0$. The constants $c_1 \approx 2/n^2$ and $c_\mu \approx \mu_{\mathrm{eff}}/n^2$ for $\mu_{\mathrm{eff}} < n^2$ are learning rates satisfying $c_1 + c_\mu \leq 1$. The approximate values reflect the rank of the input matrix or the number of input samples, divided by the degrees of freedom of the covariance matrix. The remaining degrees of freedom are covered by the old covariance matrix $\boldsymbol{C}_k$. Again, the equation is governed by a stationarity condition.

Proposition 8 (Stationarity of covariance matrix $\boldsymbol{C}$). *Given pure random ranking and $\boldsymbol{p}_k^{\mathrm{c}} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_k)$ and $h_\sigma = 1$, we have*

$$E(\boldsymbol{C}_{k+1} | \boldsymbol{C}_k) = \boldsymbol{C}_k\,. \tag{52}$$

Proof idea. Compute the expected value of Equation (51). □

Finally, we can state general linear invariance for CMA-ES, analogous to scale invariance in Proposition 6 and Fig. 5.

Proposition 9 (Invariance under general linear transformations). *The CMA-ES is invariant under full rank linear transformations of the search space, that is, for each $f : \mathbb{R}^n \to \mathbb{R}$ invariant under*

$$\mathscr{H}_{\mathrm{GL}} : f \mapsto \{f \circ \boldsymbol{B}^{-1} : \boldsymbol{x} \mapsto f(\boldsymbol{B}^{-1}\boldsymbol{x}) \mid \boldsymbol{B} \text{ is a full rank } n \times n \text{ matrix}\}\,. \tag{53}$$

The respective bijective state space transformation reads

$$T_{\boldsymbol{B}} : (\boldsymbol{m}, \sigma, \boldsymbol{C}, \boldsymbol{p}^{\sigma}, \boldsymbol{p}^{\mathrm{c}}) \mapsto (\boldsymbol{B}\boldsymbol{m}, \sigma, \boldsymbol{B}\boldsymbol{C}\boldsymbol{B}^{\mathrm{T}}, \boldsymbol{p}^{\sigma}, \boldsymbol{B}\boldsymbol{p}^{\mathrm{c}}) \,. \tag{54}$$

Furthermore, for each f, the set $\mathscr{H}_{\mathrm{GL}}(f)$ is an equivalence class with identical algorithm trace $T_{\boldsymbol{B}}(\boldsymbol{m}_k, \sigma_k, \boldsymbol{C}_k, \boldsymbol{p}_k^{\sigma}, \boldsymbol{p}_k^{\mathrm{c}})$ for a state s and the initial state $(\boldsymbol{m}_0, \sigma_0, \boldsymbol{C}_0, \boldsymbol{p}_0^{\sigma}, \boldsymbol{p}_0^{\mathrm{c}}) = T_{\boldsymbol{B}}^{-1}(s)$.

Proof idea. Straight forward computation of the updated tuple. The equivalence relation property can be shown elementarily (compare Proposition 1) or by recognizing that the set of full rank matrices is a transformation group over the set $\{f : \mathbb{R}^n \to \mathbb{R}\}$ with group action $(\boldsymbol{B}, f) \mapsto f \circ \boldsymbol{B}^{-1}$ and therefore induces the equivalence classes $\mathscr{H}_{\mathrm{GL}}(f)$ as orbits of f under the group action. □

A commutative diagram, analogous to Fig. 5, applies with $T_{\boldsymbol{B}}$ in place of $T(\alpha)$ and using $f(\boldsymbol{B}^{-1}\boldsymbol{x})$ in the lower path. The transformation $\boldsymbol{B}$ can be interpreted as a change of basis and therefore CMA-ES is invariant under linear coordinate system transformations. All further considerations made for scale invariance hold for invariance under general linear transformations likewise.

Because an appropriate (initial) choice of $\boldsymbol{B}$ is usually not available in practice, general linear invariance must be complemented with adaptivity of $\boldsymbol{C}$ to make it useful in practice and eventually *adapt* a linear encoding [17].

Corollary 1 (Adaptive linear encoding and variable metric [17]). *The covariance matrix adaptation implements an adaptive linear problem encoding, that is, in other words, an adaptive change of basis, or a change of coordinate system, or a variable metric for an evolution strategy.*

Proof idea. (The proof can be found in [16]). General linear invariance achieves identical performance on $f(\boldsymbol{B}^{-1}\boldsymbol{x})$ under respective initial conditions. Here, $\boldsymbol{B}$ is the linear problem encoding used within the algorithm. Changing (or adapting) $\boldsymbol{C}$ without changing $\boldsymbol{m}$ turns out to be equivalent with changing the encoding (or representation) $\boldsymbol{B}$ in a particular way without changing $\boldsymbol{B}^{-1}\boldsymbol{m}$ (see also [13, 16]). Also, for each possible encoding we find a respective covariance matrix $\boldsymbol{B}\boldsymbol{B}^{\mathrm{T}}$. □

While adaptation of $\boldsymbol{C}$ is essential to implement general linear invariance, rotation invariance does not necessarily depend on an adaptation of $\boldsymbol{C}$: rotation invariance is already achieved for $\boldsymbol{C} \equiv \mathbf{I}$, because $\boldsymbol{B}\mathbf{I}\boldsymbol{B}^{\mathrm{T}} = \mathbf{I}$ when $\boldsymbol{B}$ is a rotation matrix, compare (54). Nevertheless, it is important to note that covariance matrix adaptation *preserves* rotation invariance.

Corollary 2 (Rotation invariance). *The CMA-ES is invariant under search space rotations.*

Proof idea. Rotation invariance follows from Proposition 9 when restricted to the orthogonal group with $\boldsymbol{B}\boldsymbol{B}^{\mathrm{T}} = \mathbf{I}$ (for *any* initial state). □

8 An Experiment on Two Noisy Functions

We advocate testing new search algorithms always on pure random, on linear and on various (non-separable) quadratic functions with various initializations. For the $(\mu/\mu_{\mathrm{w}},\lambda)$-CMA-ES this has been done elsewhere with the expected results: parameters are unbiased on pure random functions, the step-size σ grows geometrically fast on linear functions, and on convex quadratic functions the level sets of the search distribution align with the level sets of the cost function, in that $\boldsymbol{C}^{-1}$ aligns to the Hessian up to a scalar factor and small stochastic fluctuations [15, 25].

Here, we show results on the well-known Rosenbrock function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n-1} 100\,(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\,,$$

where the possible achievement is less obvious. In order to "unsmoothen" the landscape, a noise term is added: each function value is multiplied with

$$\exp\left(\frac{\alpha_N}{2n} \times (G + C/10)\right) + \frac{\alpha_N}{2n} \times (G + C/10)\,, \tag{55}$$

where G and C are standard Gauss (normal) and standard Cauchy distributed random numbers, respectively. All four random numbers in (55) are sampled independently each time f is evaluated. The term is a mixture between the common normal noise $1+G$, which we believe has a principal "design flaw" [31], and the log-normal noise $\exp(G)$ which is alone comparatively easy to solve, each mixed with a heavy tail distribution which cannot be alleviated through averaging. We believe that this adds several difficulties on top of each other.

We show results for two noise levels, $\alpha_N = 0.01$ and $\alpha_N = 1$. A section through the 5-D and the 20-D landscape for $\alpha_N = 1$ is shown in Fig. 8. The lower dimensional landscape appears more disturbed but is not more difficult to optimize.

Figure 9 shows the output from a typical run for $\alpha_N = 0.01$ of the $(\mu/\mu_{\mathrm{w}},\lambda)$-CMA-ES with $\boldsymbol{m}_0 = -\mathbf{1}$ and $\sigma_0 = 1$ (correctly presuming that in all variables $m_i \pm 3\sigma_0$ embrace the optimum at $\mathbf{1}$). The calling sequence in Matlab was[5]

```
opts.evalparallel = 'on'; % only one feval() call per iteration
cmaes('frosennoisy', -ones(20,1), 1, opts);  % run CMA-ES
plotcmaesdat;             % plot figures using output files
```

The default population size for $n = 20$ is $\lambda = 12$. An error of 10^{-9}, very close to the global optimum, is reached after about 20 000 function evaluations (without covariance matrix adaptation it takes about 250 000 function evaluations to reach 10^{-2}). The effect of the noise is hardly visible in the performance. In some cases, the

[5] Source code is available at `http://www.lri.fr/~hansen/cmaes_inmatlab.html` and will be accessible at `http://cma.gforge.inria.fr/` in future. In our experiment, version 3.40.beta was used with Matlab.

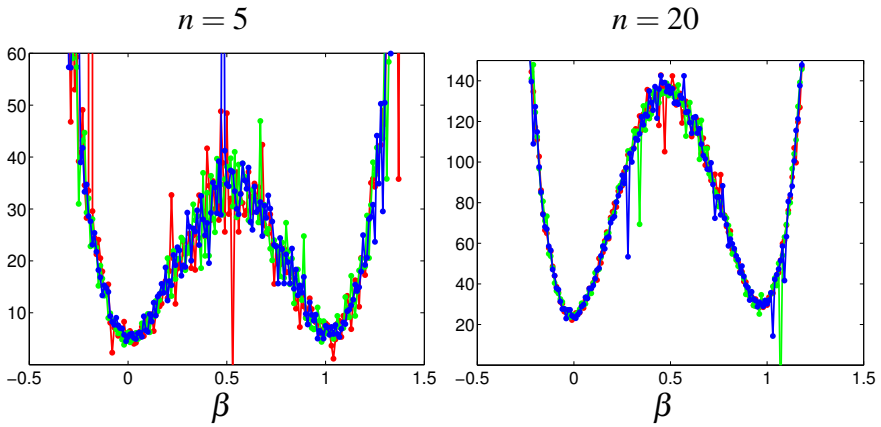

Fig. 8 Both figures show three sections of the Rosenbrock function for $\alpha_N = 1$ and argument $\boldsymbol{x} = \beta \times \mathbf{1} + \frac{1}{20}\mathcal{N}(\mathbf{0}, \mathbf{I})$. All graphs show 201 points for $\beta \in [-0.5, 1.5]$ and a single realization of $\mathcal{N}(\mathbf{0}, \mathbf{I})$ in each subfigure. The left basin about zero is initially highly attractive (compare e.g. Fig. 9, upper right) but is not nearby a local or global optimum. The basin around $\beta = 1$ is close to the global optimum at $\mathbf{1}$ and monotonically (non visibly) connected to the left basin

optimization only finds the local optimum of the function close to $(-1, 1, \ldots, 1)^{\mathrm{T}}$, in some cases the noise leads to a failure to approach any optimum (see also below).

The main challenge on the Rosenbrock function is to follow a winding ridge, in the figure between evaluation 1000 and 15 000. The ridge seems not particularly narrow: the observed axis ratio is about twenty, corresponding to a condition number of 400. But the ridge constantly changes its orientation (witness by the lower right sub-figure). Many stochastic search algorithms are not able to follow this ridge and get stuck with a function value larger than one.

In Fig. 10, the noise term is set to $\alpha_N = 1$ generating a highly rugged landscape (Fig. 8) and making it even harder to follow the winding ridge. Most search algorithms will fail to solve this function[6]. Now, two additional heuristics are examined:

First, restarting the algorithm with increasing population size (IPOP, [6]). The population size is doubled for each restart. A larger population size λ is more robust to rugged landscapes, mainly because the sample variance can be larger (for $\mu_{\mathrm{eff}} < n$, the optimal step-size on the sphere function is proportional to μ_{eff} [2]). Restarting with increasing population sizes is a very effective heuristic when a good termination criterion is available.

Second, applying an uncertainty-handling (UH, [24]). The uncertainty-handling reevaluates a few solutions and measures their resulting rank changes [24]. If the rank changes exceed a threshold, an action is taken. Here, σ is increased. This pre-

[6] There is a simple way to smoothen the landscape: a single evaluation can be replaced by the median (not the mean) of a number of evaluations. Only a few evaluations reduce the dispersion considerably, but about 1000 evaluations are necessary to render the landscape similarly smooth as with $\alpha_N = 0.01$. Together with (μ/μ_{w})-CMA-ES, single evaluations, as in Fig. 10, need overall the least number of function evaluations (comprising restarts).

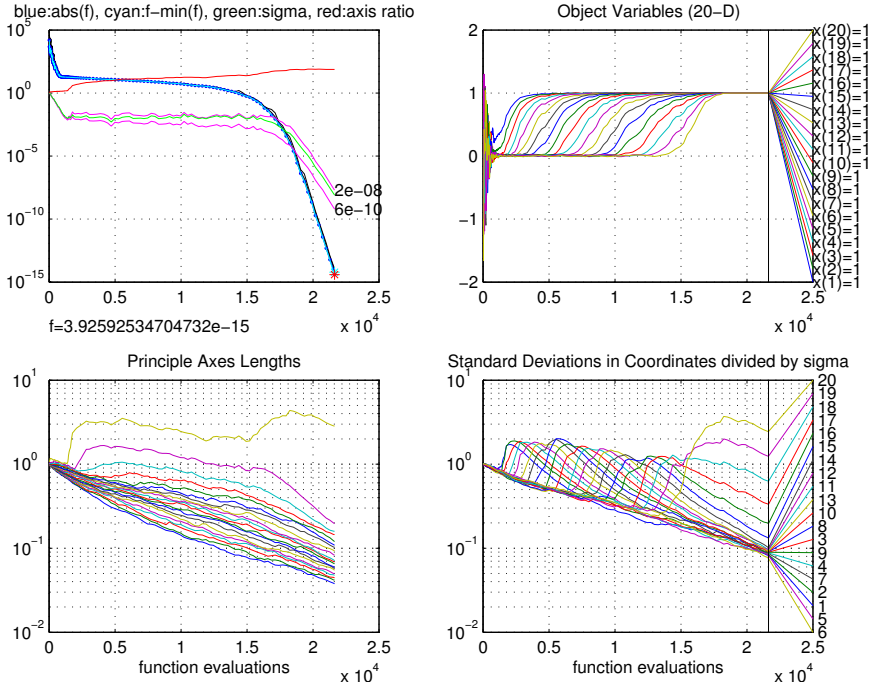

Fig. 9 A typical run of the $(\mu/\mu_{\mathrm{w}},\lambda)$-CMA-ES on the Rosenbrock function ($n = 20$) with a small disturbance of the function value ($\alpha_N = 0.01$). All values are plotted against number of objective function evaluations. Upper left: iteration-wise best function value (thick blue graph), median and worst function value (black graphs, mainly hidden), square root of the condition number of $\boldsymbol{C}_k$ (increasing red graph), smallest and largest coordinate-wise standard deviation of the distribution $\mathcal{N}\left(\mathbf{0},{\sigma_k}^2\boldsymbol{C}_k\right)$ with final values annotated (magenta), and σ_k following closely the largest standard deviation (light green). Lower left: square roots of eigenvalues of $\boldsymbol{C}_k$, sorted. Upper right: incumbent solution $\boldsymbol{m}_k$. Lower right: square roots of diagonal elements of $\boldsymbol{C}_k$

vents to getting stuck, when the noise disturbs the selection too severely, but it can also lead to divergence. This is of lesser relevance, because in this case the original algorithm would most likely have been stuck anyway. Again, a good termination criterion is essential.

Remark that in both cases, for restarts and with the uncertainty handling, another possible action is to increase the number of function evaluations used for each individual in replacing a single value with a median.

For running IPOP-UH-CMA-ES, the following sequence is added before calling `cmaes`.

```
opts.restarts = 1;              % maximum number of restarts
opts.StopOnStagnation = 'yes';  % terminate long runs
opts.noise.on = 'yes';          % activate uncertainty-handling
```

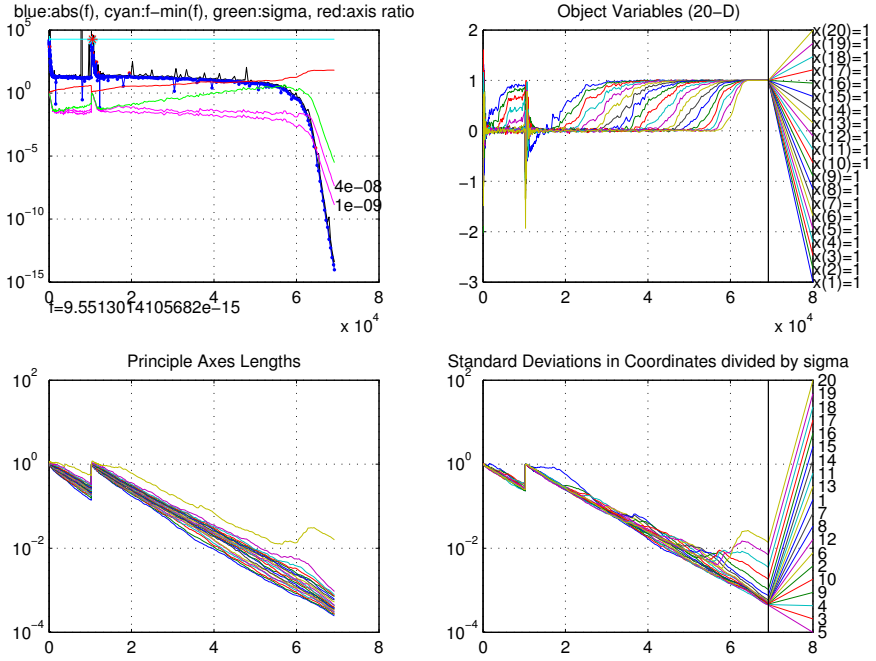

Fig. 10 A typical run of the IPOP-UH-CMA-ES on the noisy Rosenbrock function ($n = 20$, $\alpha_N = 1$), a (μ/μ_w)-CMA-ES with Uncertainly Handling restarted with Increasing POPulation size. The highly rugged lines, partly beyond 10^5, in the upper left depict the worst measured function value (out of λ). One restart was necessary to converge close to the global optimum. See also Fig. 9 for more explanations

Each restart uses the same initial conditions, here $\boldsymbol{m}_0 = -\mathbf{1}$ and $\sigma_0 = 1$ from above. For $\alpha_N = 0.01$ (Fig. 9) the uncertainty-handing increases the running length by about 15%, simply due to the reevaluations (not shown). For $\alpha_N = 1$ in Fig. 10, it shortens the running length by a factor of about ten by reducing the number of necessary restarts. Typical for noisy functions, the restart was invoked due to stagnation of the run [20]. When repeating this experiment, in about 75% one restart is needed to finally converge to the global optimum with $\lambda = 24$. Without uncertainty-handling it takes usually five to six restarts and a final population size of $\lambda \geq 384$. Without covariance matrix adaptation it takes about 70 times longer to reach a similar precision as in Fig. 10.

Experiments with the well-known Ellipsoid function,

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} 10^{6\frac{i-1}{n-1}} x_i^2$$

with the same noisy multiplier and $\alpha_N = 1$ are shown in Fig. 11 for IPOP-CMA-ES (left) and UH-CMA-ES (right). The function is less difficult and can be solved with

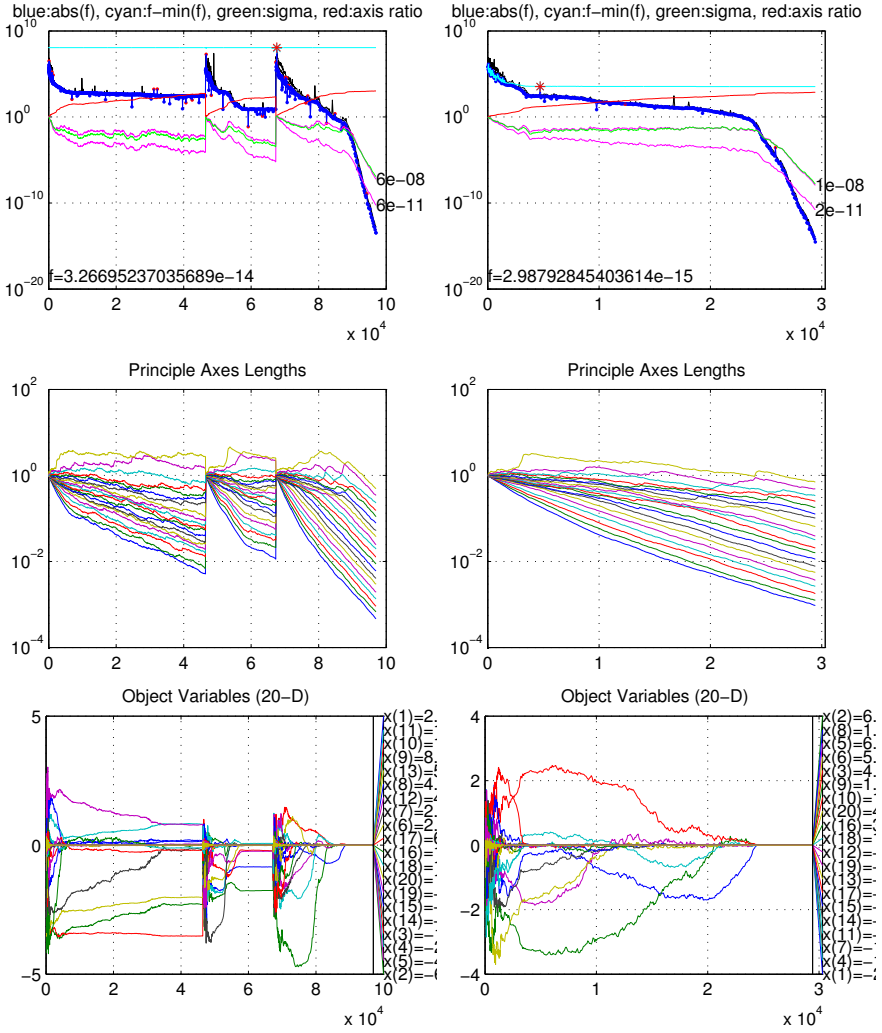

Fig. 11 Two typical runs of the IPOP-CMA-ES (left) and UH-CMA-ES (right, with uncertainty-handling) on the noisy Ellipsoid function ($n = 20$, $\alpha_N = 1$). With $\alpha_N = 0$ the Ellipsoid is solved in about 22000 function evaluations. In the lower left we can well observe that the algorithm gets stuck "in the middle of nowhere" during the first two launches. See also Fig. 9 for more explanations

a population size $\lambda = 48$ using the IPOP approach and with the default population size of 12 with UH-CMA-ES.

9 Summary

Designing a search algorithm is intricate. We recapitulate the principled design ideas for deriving the CMA-ES algorithm.

- Using a minimal amount of **prior assumptions** on the cost function f in order to achieve maximal robustness and minimal susceptibility to deceptiveness.
  - Generating candidate solutions by sampling a **maximum entropy** distribution adds the least amount of unwarranted information. This implies the stochastic nature of the algorithm and that no *construction* of potentially better points is undertaken. This also implies an internal quadratic model—at least when the distribution has finite variances—and stresses the importance of neighborhood. Consequently, a variable neighborhood suggests itself.
  - **Unbiasedness** of all algorithm components, given the objective function is random and independent of its argument. This principle suggests that only the current state and the selection information should bias the behavior of the algorithm. Adding another bias would add additional prior assumptions. We have deliberately violated this principle for uncertainty-handling as used in one experiment, where the step-size is increased under highly perturbed selection.
  - Only the *ranking* of the most recently sampled candidate solutions is used as feed-back from the objective function. This implies an attractive invariance property of the algorithm.

  Exploiting more specific information on f effectively, for example smoothness, convexity, or (partial) separability will lead to different and more specific design decisions, with a potential advantage on smooth, convex or separable functions respectively.

- Introducing and maintaining **invariance** properties. Even invariance is related to avoiding prior assumptions as it implies not exploiting specific structure of the objective function f (for example separability). We can differentiate two main cases.
  - Unconditional invariance properties do not depend on the initial conditions of the algorithm and strengthen any empirical performance observation. They allow to unconditionally generalize empirical observations to the equivalence class of functions induced by the invariance property.
  - Invariance properties that depend on state variables of the algorithm (like σ_k for scale invariance in Fig. 5) must be complemented with adaptivity. They are particularly attractive, if adaptivity can drive the algorithm fast into the most desirable state. This behavior can be empirically observed for CMA-ES on the equivalence class of convex-quadratic functions. Step-size control drives step-size σ_k close to its optimal value and adaptation of the covariance matrix reduces these functions to the sphere model.

- Exploiting all **available information** effectively. The available information and its exploitation are highly restricted by the first two demands. Using a *deterministic* ranking and *different* weights for updating $\boldsymbol{m}$ and $\boldsymbol{C}$ are due to this design principle. Also the evolution paths in (46) in (51) are governed by exploiting otherwise unused sign information. Using the evolution paths does not violate any of the above demands, but allows to additionally exploit dependency information between successive time steps of the algorithm.
- Solving the two most basic continuous domain functions reasonably fast. Solving the linear function and the sphere function reasonably fast implies to introduce step-size control. These two functions are quite opposed: the latter requires convergence, the former requires divergence of the algorithm.

Finally, two **heuristic concepts** are applied in CMA-ES.

- Reinforcement of the better solutions and the better steps (variations) when updating mean and variances respectively. This seems a rather unavoidable heuristic given a conservative use of information from f. This heuristic bears the maximum likelihood principle.
- Orthogonality of successive steps. This heuristic is a rather common conception in continuous domain search.

Pure random search, where the sample distribution remains constant in the iteration sequence, follows most of the above design principles and has some attractive robustness features. However, pure random search neither accumulates information from the past in order to modify the search distribution, nor changes and adapts internal state variables. Adaptivity of state variables however detaches the algorithm from the initial conditions and let (additional) invariance properties come to life. Only invariance to increasing f-value transformations (Proposition 1) is independent of state variables of the search algorithm. We draw the somewhat surprising conclusion that the abstract notion of invariance leads, by advising the introduction of adaptivity, when carefully implemented, to a vastly improved practical performance.

Despite its generic, principled design, the **practical performance** of CMA-ES turns out to be surprisingly competitive, or even superior, also in comparatively specific problem classes. This holds in particular when more than $100n$ function evaluations are necessary to find a satisfactory solution [21]—even for example on smooth unimodal non-quadratic functions [8], or on highly multimodal functions [23] and on noisy or highly rugged functions [20]. In contrast, much better search heuristics are available given (nearly) convex-quadratic problems or (partially) separable multimodal problems.

Acknowledgements The authors would like to express their gratitude to Marc Schoenauer for his kind and consistent support.

References

1. Akimoto, Y., Nagata, Y., Ono, I., Kobayashi, S.: Bidirectional relation between CMA evolution strategies and natural evolution strategies. In: R. Schaefer, C. Cotta, J. Kolodziej, G. Rudolph (eds.) Parallel Problem Solving from Nature - PPSN XI, Proceedings, Part I, *Lecture Notes in Computer Science*, vol. 6238, pp. 154–163. Springer (2010)
2. Arnold, D.: Optimal weighted recombination. In: Foundations on Genetic Algorithms FOGA 2005, *Lecture Notes in Computer Science LNCS*, vol. 3469, pp. 215–237. Springer (2005)
3. Arnold, D.: Weighted multirecombination evolution strategies. Theoretical computer science **361**(1), 18–37 (2006)
4. Arnold, D.V., Hansen, N.: Active covariance matrix adaptation for the (1+1)-CMA-ES. In: Genetic and Evolutionary Computation Conference, GECCO 2010, Proceedings, pp. 385–392 (2010)
5. Arnold, L., Auger, A., Hansen, N., Ollivier, Y.: Information-geometric optimization algorithms: A unifying picture via invariance principles. Arxiv preprint arXiv:1106.3708 (2011)
6. Auger, A., Hansen, N.: A restart CMA evolution strategy with increasing population size. In: B. McKay, et al. (eds.) The 2005 IEEE International Congress on Evolutionary Computation (CEC 2005), vol. 2, pp. 1769–1776 (2005)
7. Auger, A., Hansen, N.: Reconsidering the progress rate theory for evolution strategies in finite dimensions. In: Proceedings of the 8th annual conference on genetic and evolutionary computation GECCO, pp. 445–452. ACM (2006)
8. Auger, A., Hansen, N., Zerpa, J., Ros, R., Schoenauer, M.: Experimental comparisons of derivative free optimization algorithms. In: 8th international symposion on experimental algorithms SEA 2009, *Lecture Notes in Computer Science LNCS*, vol. 5526, pp. 3–15. Springer (2009)
9. Beyer, H.G.: The Theory of Evolution Strategies. Natural Computing Series. Springer, Heidelberg (2001)
10. Brockhoff, D., Auger, A., Hansen, N., Arnold, D.V., Hohm, T.: Mirrored sampling and sequential selection for evolution strategies. In: R. Schaefer et al. (ed.) Parallel Problem Solving from Nature (PPSN XI), *LNCS*, vol. 6238, pp. 11–20. Springer (2010)
11. Glasmachers, T., Schaul, T., Sun, Y., Wierstra, D., Schmidhuber, J.: Exponential natural evolution strategies. In: M. Pelikan, J. Branke (eds.) Genetic and Evolutionary Computation Conference, GECCO 2010, Proceedings, pp. 393–400. ACM (2010)
12. Hansen, N.: The CMA evolution strategy: A tutorial. http://www.lri.fr/~hansen/cmatutorial.pdf
13. Hansen, N.: Invariance, self-adaptation and correlated mutations in evolution strategies. In: M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. Merelo, H.P. Schwefel (eds.) Proceedings of PPSN VI, Parallel Problem Solving from Nature, pp. 355–364. Springer (2000)
14. Hansen, N.: An analysis of mutative σ-self-adaptation on linear fitness functions. Evolutionary Computation **14**(3), 255–275 (2006)
15. Hansen, N.: The CMA evolution strategy: a comparing review. In: J. Lozano, P. Larranaga, I. Inza, E. Bengoetxea (eds.) Towards a new evolutionary computation. Advances on estimation of distribution algorithms, pp. 75–102. Springer (2006)
16. Hansen, N.: Adaptive encoding for optimization. Research Report RR-6518, INRIA (2008). URL http://hal.inria.fr/inria-00275983/en/
17. Hansen, N.: Adaptive encoding: How to render search coordinate system invariant. In: G. Rudolph, et al. (eds.) Parallel Problem Solving from Nature (PPSN X), LNCS, pp. 205–214 (2008)
18. Hansen, N.: CMA-ES with two-point step-size adaptation. Tech. Rep. RR-6527, INRIA (2008). URL http://hal.inria.fr/inria-00276854/en/
19. Hansen, N.: Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In: Workshop Proceedings of the GECCO Genetic and Evolutionary Computation Conference, pp. 2389–2395. ACM (2009)
20. Hansen, N.: Benchmarking a BI-population CMA-ES on the BBOB-2009 noisy testbed. In: Workshop Proceedings of the GECCO Genetic and Evolutionary Computation Conference, pp. 2397–2402. ACM (2009)

21. Hansen, N., Auger, A., Ros, R., Finck, S., Pošík, P.: Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In: Workshop Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010), pp. 1689–1696. ACM Press (2010)
22. Hansen, N., Gawelczyk, A., Ostermeier, A.: Sizing the population with respect to the local progress in $(1, \lambda)$-evolution strategies—a theoretical analysis. In: Evolutionary Computation, 1995., IEEE International Conference on, vol. 1, pp. 80–85 (1995)
23. Hansen, N., Kern, S.: Evaluating the CMA evolution strategy on multimodal test functions. In: X. Yao, et al. (eds.) Parallel Problem Solving from Nature PPSN VIII, *Lecture Notes in Computer Science LNCS*, vol. 3242, pp. 282–291. Springer (2004)
24. Hansen, N., Niederberger, S., Guzzella, L., Koumoutsakos, P.: A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Transactions on Evolutionary Computation **13**(1), 180–197 (2009)
25. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9**(2), 159–195 (2001)
26. Hansen, N., Ros, R.: Benchmarking a weighted negative covariance matrix update on the bbob-2010 noiseless testbed. In: Genetic and Evolutionary Computation Conference, GECCO 2010, Companion Material, pp. 1673–1680 (2010)
27. Jägersküpper, J.: Lower bounds for hit-and-run direct search. In: Yao, Xin et al. (ed.) Stochastic Algorithms: Foundations and Applications - SAGA 2007, LNCS 4665, pp. 118–129. Springer Berlin, Heidelberg (2007)
28. Jägersküpper, J.: Lower bounds for randomized direct search with isotropic sampling. Operations Research Letters **36**(3), 327–332 (2008)
29. Jastrebski, G., Arnold, D.: Improving evolution strategies through active covariance matrix adaptation. In: The 2006 IEEE International Congress on Evolutionary Computation (CEC 2006), pp. 2814–2821 (2006)
30. Jebalia, M.: personal communication
31. Jebalia, M., Auger, A., Hansen, N.: Log linear convergence and divergence of the scale-invariant (1+1)-ES in noisy environments. Algorithmica p. in print (2011)
32. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: L.J. Eshelman (ed.) Proceedings of the 6th International Conference on Genetic Algorithms, ICGA, pp. 184–192. Morgan Kaufmann (1995)
33. Ostermeier, A., Gawelczyk, A., Hansen, N.: Step-size adaptation based on non-local use of selection information. In: Y. Davidor, et al. (eds.) Parallel Problem Solving from Nature PPSN IV, *Lecture Notes in Computer Science LNCS*, vol. 866, pp. 189–198. Springer (1994)
34. Rechenberg, I.: Evolutionsstrategie, Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. frommann-holzboog, Stuttgart (1973)
35. Salomon, R., van Hemmen, J.L.: Accelerating backpropagation through dynamic self-adaptation. Neural Networks **9**(4), 589–601 (1996)
36. Schumer, M., Steiglitz, K.: Adaptive step size random search. IEEE Transactions on Automatic Control **13**(3), 270–276 (1968)
37. Schwefel, H.P.: Numerical Optimization of Computer Models. John Wiley & Sons, Inc., New York, NY, USA (1981)
38. Suttorp, T., Hansen, N., Igel, C.: Efficient covariance matrix update for variable metric evolution strategies. Machine Learning **75**(2), 167–197 (2009)
39. Teytaud, O., Fournier, H.: Lower bounds for evolution strategies using VC-dimension. In: Parallel Problem Solving from Nature PPSN X, *Lecture Notes in Computer Science LNCS*, vol. 5199, pp. 102–111. Springer (2008)
40. Teytaud, O., Gelly, S.: General lower bounds for evolutionary algorithms. In: Parallel Problem Solving from Nature PPSN IX, *Lecture Notes in Computer Science LNCS*, vol. 4193, pp. 21–31. Springer (2006)
41. Wierstra, D., Schaul, T., Peters, J., Schmidhuber, J.: Natural evolution strategies. In: IEEE Congress on Evolutionary Computation, pp. 3381–3387. IEEE (2008)

Appendix

The $(\mu/\mu_{\mathrm{w}},\lambda)$-CMA-ES, as described in this chapter, is summarized in Table 1. We have $\boldsymbol{p}^{\sigma}_{k=0} = \boldsymbol{p}^{\mathrm{c}}_{k=0} = \mathbf{0}$, $\boldsymbol{C}_{k=0} = \mathbf{I}$, while $\boldsymbol{m}_{k=0} \in \mathbb{R}^n$ and $\sigma_{k=0} > 0$ are user defined. Additionally, $\boldsymbol{x}_{i:\lambda}$ is the i-th best of the solutions $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_\lambda$,

Table 1 Summary of the update equations for the state variables in the $(\mu/\mu_{\mathrm{w}},\lambda)$-CMA-ES with iteration index $k = 0,1,2,\ldots$. The chosen ordering of equations allows to remove the iteration index in all variables but $\boldsymbol{m}_k$. Unexplained parameters and constants are given in Table 2

Given $k \in \mathbb{N}\cup\{0\}$, $\boldsymbol{m}_k \in \mathbb{R}^n$, $\sigma_k \in \mathbb{R}_+$, $\boldsymbol{C}_k \in \mathbb{R}^{n\times n}$ positive definite, $\boldsymbol{p}^{\sigma}_k \in \mathbb{R}^n$, and $\boldsymbol{p}^{\mathrm{c}}_k \in \mathbb{R}^n$

$$\boldsymbol{x}_i \sim \boldsymbol{m}_k + \sigma_k \times \mathcal{N}(\mathbf{0},\boldsymbol{C}_k) \qquad \text{is normally distributed for } i = 1,\ldots,\lambda \tag{56}$$

$$\boldsymbol{m}_{k+1} = \boldsymbol{m}_k + c_{\mathrm{m}} \sum_{i=1}^{\mu} w_i \left(\boldsymbol{x}_{i:\lambda} - \boldsymbol{m}_k\right) \qquad \text{where } f(\boldsymbol{x}_{1:\lambda}) \le \cdots \le f(\boldsymbol{x}_{\mu:\lambda}) \le f(\boldsymbol{x}_{\mu+1:\lambda}) \ldots \tag{57}$$

$$\boldsymbol{p}^{\sigma}_{k+1} = (1-c_\sigma)\boldsymbol{p}^{\sigma}_k + \sqrt{c_\sigma(2-c_\sigma)\mu_{\mathrm{eff}}}\, \boldsymbol{C}_k^{-\frac{1}{2}} \frac{\boldsymbol{m}_{k+1}-\boldsymbol{m}_k}{c_{\mathrm{m}}\,\sigma_k} \tag{58}$$

$$\boldsymbol{p}^{\mathrm{c}}_{k+1} = (1-c_{\mathrm{c}})\boldsymbol{p}^{\mathrm{c}}_k + h_\sigma \sqrt{c_{\mathrm{c}}(2-c_{\mathrm{c}})\mu_{\mathrm{eff}}} \frac{\boldsymbol{m}_{k+1}-\boldsymbol{m}_k}{c_{\mathrm{m}}\,\sigma_k} \tag{59}$$

$$\begin{aligned}\boldsymbol{C}_{k+1} = {} & (1-c_1+(1-{h_\sigma}^2)\,c_1 c_{\mathrm{c}}(2-c_{\mathrm{c}}))\,\boldsymbol{C}_k \\ & + c_1 \boldsymbol{p}^{\mathrm{c}}_{k+1} {\boldsymbol{p}^{\mathrm{c}}_{k+1}}^{\mathrm{T}} + c_\mu \sum_{i=1}^{\mu} w_i \left(\frac{\boldsymbol{x}_{i:\lambda}-\boldsymbol{m}_k}{\sigma_k} \times \frac{\left(\boldsymbol{x}_{i:\lambda}-\boldsymbol{m}_k\right)^{\mathrm{T}}}{\sigma_k} - \boldsymbol{C}_k \right)\end{aligned} \tag{60}$$

$$\sigma_{k+1} = \sigma_k \times \exp\left(1 \wedge \frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}^{\sigma}_{k+1}\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right) \tag{61}$$

$$h_\sigma = \begin{cases} 1 & \text{if } \frac{\|\boldsymbol{p}^{\sigma}_{k+1}\|^2}{1-(1-c_\sigma)^{2(k+1)}} < \left(2+\frac{4}{n+1}\right) n \\ 0 & \text{otherwise} \end{cases},$$

for $\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\| = \sqrt{2}\,\Gamma(\frac{n+1}{2})/\Gamma(\frac{n}{2}) \approx \sqrt{n-1/2}$ we use the better approximation $\sqrt{n}\left(1-\frac{1}{4n}+\frac{1}{21n^2}\right)$, and $\boldsymbol{C}_k^{-\frac{1}{2}}$ is symmetric with positive eigenvalues and satisfies $\boldsymbol{C}_k^{-\frac{1}{2}}\boldsymbol{C}_k^{-\frac{1}{2}} = (\boldsymbol{C}_k)^{-1}$. The binary $\wedge$ operator depicts the minimum of two values with low operator precedence. The default parameter values are shown in Table 2.

Table 2 Default parameter values of the (μ/μ_{w})-CMA-ES, where by definition $\sum_{i=1}^{\mu}|w_i| = 1$ and $\mu_{\mathrm{eff}}^{-1} = \sum_{i=1}^{\mu} w_i^2$

$$\lambda = 4 + \lfloor 3\ln n \rfloor \quad \text{population size, see also [6, 19]}$$

$$\mu = \left\lfloor \frac{\lambda}{2} \right\rfloor \quad \text{parent number}$$

$$w_i = \frac{\ln\left(\frac{\lambda+1}{2}\right) - \ln i}{\sum_{j=1}^{\mu}\left(\ln\left(\frac{\lambda+1}{2}\right) - \ln j\right)} \quad \text{recombination weights for } i = 1,\ldots,\mu$$

$$c_{\mathrm{m}} = 1 \quad \text{learning rate for the mean, sometimes interpreted as rescaled mutation with } \kappa = \frac{1}{c_{\mathrm{m}}} \geq 1$$

$$c_\sigma = \frac{\mu_{\mathrm{eff}} + 2}{n + \mu_{\mathrm{eff}} + 5} \quad \text{cumulation constant for step-size, } 1/c_\sigma \text{ is the respective time constant}$$

$$d_\sigma = 1 + c_\sigma + 2\max\left(0, \sqrt{\frac{\mu_{\mathrm{eff}} - 1}{n+1}} - 1\right) \quad \text{step-size damping, is usually close to one}$$

$$c_{\mathrm{c}} = \frac{4 + \mu_{\mathrm{eff}}/n}{n + 4 + 2\mu_{\mathrm{eff}}/n} \quad \text{cumulation constant for } \boldsymbol{p}^{\mathrm{c}}$$

$$c_1 = \frac{\alpha_{\mathrm{cov}}}{(n+1.3)^2 + \mu_{\mathrm{eff}}} \quad \text{covariance matrix learning rate for the rank one update using } \boldsymbol{p}^{\mathrm{c}}$$

$$c_\mu = \min\left(1 - c_1, \alpha_{\mathrm{cov}}\frac{\mu_{\mathrm{eff}} - 2 + 1/\mu_{\mathrm{eff}}}{(n+2)^2 + \alpha_{\mathrm{cov}}\mu_{\mathrm{eff}}/2}\right) \quad \text{covariance matrix learning rate for rank-}\mu\text{ update}$$

$$\alpha_{\mathrm{cov}} = 2 \wedge \lambda/3 \quad \text{could be chosen} < 2, \text{ e.g. } \alpha_{\mathrm{cov}} = 0.5 \text{ for noisy problems}$$