Completely Derandomized Self-Adaptation in Evolution Strategies

Nikolaus Hansen hansen@bionik.tu-berlin.de
Technische Universität Berlin, Fachgebiet für Bionik, Sekr. ACK 1, Ackerstr. 71–76, 13355 Berlin, Germany

Andreas Ostermeier ostermeier@bionik.tu-berlin.de
Technische Universität Berlin, Fachgebiet für Bionik, Sekr. ACK 1, Ackerstr. 71–76, 13355 Berlin, Germany

Abstract

This paper puts forward two useful methods for self-adaptation of the mutation distribution – the concepts of *derandomization* and *cumulation*. Principle shortcomings of the concept of *mutative* strategy parameter control and two levels of derandomization are reviewed. Basic demands on the self-adaptation of *arbitrary* (normal) mutation distributions are developed. Applying arbitrary, normal mutation distributions is equivalent to applying a general, linear problem encoding.

The underlying objective of mutative strategy parameter control is roughly to favor previously selected mutation steps in the future. If this objective is pursued rigorously, a completely derandomized self-adaptation scheme results, which adapts arbitrary normal mutation distributions. This scheme, called *covariance matrix adaptation* (CMA), meets the previously stated demands. It can still be considerably improved by cumulation – utilizing an *evolution path* rather than single search steps.

Simulations on various test functions reveal local and global search properties of the evolution strategy with and without covariance matrix adaptation. Their performances are comparable only on perfectly scaled functions. On badly scaled, non-separable functions usually a speed up factor of several orders of magnitude is observed. On moderately mis-scaled functions a speed up factor of three to ten can be expected.

Keywords

Evolution strategy, self-adaptation, strategy parameter control, step size control, derandomization, derandomized self-adaptation, covariance matrix adaptation, evolution path, cumulation, cumulative path length control.

1 Introduction

The evolution strategy (ES) is a stochastic search algorithm that addresses the following search problem: Minimize a non-linear objective function that is a mapping from search space $\mathcal{S} \subseteq \mathbb{R}^n$ to $\mathbb{R}$. Search steps are taken by stochastic variation, so-called mutation, of (recombinations of) points found so far. The best out of a number of new search points are selected to continue. The mutation is usually carried out by adding a realization of a normally distributed random vector. It is easy to imagine that the parameters of the normal distribution play an essential role for the performance[1] of the search

[1] Performance, as used in this paper, always refers to the (expected) number of required objective *function evaluations* to reach a certain function value.

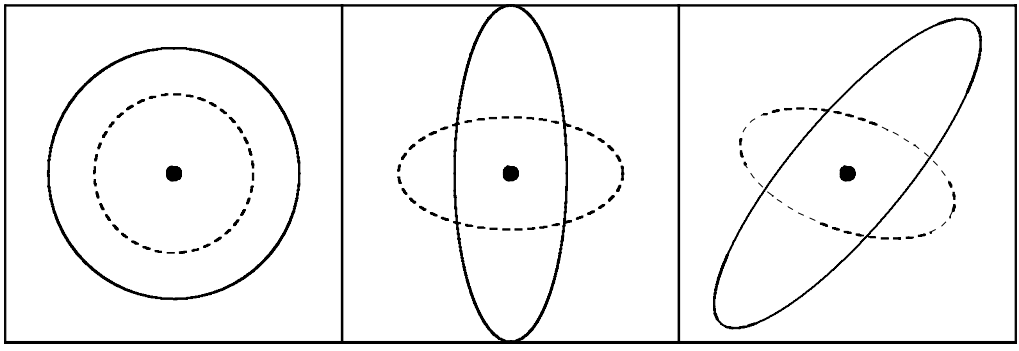

Figure 1: One-σ lines of equal probability density of two normal distributions respectively. Left: one free parameter (circles). Middle: n free parameters (axis-parallel ellipsoids). Right: $(n^2+n)/2$ free parameters (arbitrarily oriented ellipsoids).

algorithm. This paper is specifically concerned with the adjustment of the parameters of the normal mutation distribution.

Among others, the parameters that parameterize the mutation distribution are called *strategy parameters*, in contrast to object parameters that define points in search space. Usually, no particularly detailed knowledge about the suitable choice of strategy parameters is available. With respect to the mutation distribution, there is typically only a small width of strategy parameter settings where substantial search progress can be observed (Rechenberg, 1973). Good parameter settings differ remarkably from problem to problem. Even worse, they usually change during the search process (possibly by several orders of magnitude). For this reason, self-adaptation of the mutation distribution that dynamically adapts strategy parameters during the search process is an essential feature in ESs.

We briefly review three consecutive steps of adapting normal mutation distributions in ESs.

1. The normal distribution is chosen to be isotropic. Surfaces of equal probability density are circles (Figure 1, left), or (hyper-)spheres if $n \geq 3$. Overall variance of the distribution—in other words the (global) step size or the expected step length—is the only free strategy parameter.

2. The concept of global step size can be generalized.[2] Each coordinate axis is assigned a different variance (Figure 1, middle)—often referred to as individual step sizes. There are n free strategy parameters. The disadvantage of this concept is the dependency on the coordinate system. Invariance against rotation of the search space is lost. Why invariance is an important feature of an ES is discussed in Section 6.

3. A further generalization dynamically adapts the orthogonal coordinate system, where each coordinate axis is assigned a different variance (Figure 1, right). Any normal distribution (with zero mean) can be produced. This concept results in

[2]There is more than one sensible generalization. For example, it is possible to provide one arbitrarily oriented axis with a different variance. Then $n+1$ parameters have to be adapted. Such an adaptation can be formulated independent of the coordinate system.

$(n^2+n)/2$ free strategy parameters. If an adaptation mechanism is suitably formulated, the invariance against rotations of search space is restored.

The adaptation of strategy parameters in ESs typically takes place in the concept of *mutative strategy parameter control* (MSC). Strategy parameters are mutated, and a new search point is generated by means of this mutated strategy parameter setting.

We exemplify the concept formulating an $(1, \lambda)$-ES with (purely) mutative control of one global step size. $\boldsymbol{x}^{(g)} \in \mathbb{R}^n$ and $\sigma^{(g)} \in \mathbb{R}^+$ are the object parameter vector and step size[3] of the parent at generation g. The mutation step from generation g to $g+1$ reads for each offspring $k = 1, \ldots, \lambda$

$$\sigma_k^{(g+1)} = \sigma^{(g)} \exp(\xi_k) \quad (1)$$

$$\boldsymbol{x}_k^{(g+1)} = \boldsymbol{x}^{(g)} + \sigma_k^{(g+1)} \boldsymbol{z}_k \ , \quad (2)$$

where:

$\xi_k \in \mathbb{R}$, for $k = 1, \ldots, \lambda$ independent realizations of a random number with zero mean. Typically, ξ_k is normally distributed with standard deviation $1/\sqrt{2n}$ (Bäck and Schwefel, 1993). We usually prefer to choose $\mathrm{P}(\xi_k = 0.3) = \mathrm{P}(\xi_k = -0.3) = 1/2$ (Rechenberg, 1994).

$\boldsymbol{z}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \in \mathbb{R}^n$, for $k = 1, \ldots, \lambda$ independent realizations of a $(\boldsymbol{0}, \boldsymbol{I})$-normally distributed random vector, where $\boldsymbol{I}$ is the unity matrix. That is, components of $\boldsymbol{z}_k$ are independent and identically $(0, 1)$-normally distributed.

After λ mutation steps are carried out, the best offspring (with respect to the object parameter vector $\boldsymbol{x}_k^{(g)}$) is selected to start the next generation step. Equation (1) facilitates the mutation on the strategy parameter level. The standard deviation of ξ represents the mutation strength on the strategy parameter level.

This adaptation concept was introduced by Rechenberg (1973) for global and individual step sizes. Schwefel (1981) expanded the mutative approach to the adaptation of arbitrary normal mutation distributions, which is discussed in more detail in Section 3.1. Ostermeier et al. (1994b) introduced a first level of derandomization into strategy parameter control that facilitates an individual step size adaptation in constantly small populations. (If the standard mutative strategy parameter control is used to adapt individual step sizes, based on our experience, the population size has to scale linearely with the problem dimension n.)

In this paper, a second level of derandomization is put forward: The original objective of mutative strategy parameter control — that is to favor strategy parameter settings that produce selected steps with high probability (again) — is explicitely realized. Complete derandomization, applied to the adaptation of arbitrary normal mutation distributions, leads almost inevitably to the *covariance matrix adaptation* (CMA) described in Section 3.2.

The paper is organized as follows. In Section 2, we discuss the basic shortcomings of the concept of mutative strategy parameter control and review the derandomized approach to strategy parameter control in detail. In Section 3, the different concepts are investigated with respect to the adaptation of arbitrary normal mutation distributions.

[3] In this paper, *step size* always refers to σ. Step size σ is the component-wise standard deviation of the random vector $\sigma\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \in \mathbb{R}^n$. For this vector, $n\sigma^2$ can be interpreted as *overall variance* and $\mathrm{E}[||\sigma\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})||] \approx \sigma\sqrt{n}$ is the expected *step length*.

General demands on such an adaptation mechanism are developed, and the CMA-ES is introduced. In Section 4, the objective of strategy parameter control is expanded using search paths (evolution paths) rather than single search steps as adaptation criterion. This concept is implemented by means of so-called cumulation. In Section 5, we formulate the CMA-ES algorithm, an evolution strategy that adapts arbitrary, normal mutation distributions within a completely derandomized adaptation scheme with cumulation. Section 6 discusses the test functions used in Section 7, where various simulation results are presented. In Section 8, a conclusion is given. Appendix A provides a MATLAB implementation of the CMA-ES.

2 Derandomization of Mutative Strategy Parameter Control

In the concept of mutative strategy parameter control (MSC), the selection probability of a strategy parameter setting is the probability to produce (with these strategy parameters) an object parameter setting that will be selected. Selection probability of a strategy parameter setting can also be identified with its fitness. Consequently, we assume the following aim behind the concept of MSC:[4] Find the strategy parameters with the highest selection probability or, in other words, raise the probability of mutation steps that produced selected individuals.[5] This implicitly assumes that strategy parameters that produced selected individuals before are suitable parameters in the (near) future. One idea of derandomization is to increase the probability of producing previously selected mutation steps again in a more direct way than MSC.

Before the concept of derandomization is discussed in detail, some important points concerning the concept of MSC are reviewed.

- Selection of the strategy parameter setting is indirect. The selection process operates on the object parameter adjustment. Comparing two different strategy parameter settings, the better one has (only) a higher probability to be selected — due to object parameter realization. Differences between these selection probabilities can be quite small, that is, the selection process on the strategy parameter level is highly disturbed. One idea of derandomization is to reduce or even eliminate this disturbance.

- The mutation on the strategy parameter level, as with any mutation, produces different individuals that undergo selection. Mutation strength on the strategy parameter level must ensure a significant selection difference between the individuals. In our view, this is the primary task of the mutation operator.

- Mutation strength on the strategy parameter level is usually kept constant throughout the search process. Therefore, the mutation operator (on the strategy parameter level) must facilitate an effective mutation strength that is virtually independent of the actual position in strategy parameter space. This can be complicated, because the best distance measure may not be obvious, and the position-independent formulation of a mutation operator can be difficult (see Section 3.1).

[4] We assume there is one best strategy parameter setting at each time step. Alternatively, to apply different parameter settings at the same time means to challenge the idea of a normal search distribution. We found this alternative to be disadvantageous.

[5] Maximizing the selection probability is, in general, not identical with maximizing a progress rate. This is one reason for the often observed phenomenon that the global step size is adapted to be too small by MSC. This problem is addressed in the path length control (Equations (16) and (17)) by the so-called cumulation (Section 4).

- The possible (and the realized) change rate of the strategy parameters between two generations is an important factor. It gives an upper limit for the adaptation speed. When adapting n or even n^2 strategy parameters, where n is the problem dimension, adaptation speed becomes an important factor for the performance of the ES. If only one global step size is adapted, the performance is limited by the possible adaptation speed only on a linear objective function (see discussion of parameter d from (3) below).

  On the one hand, it seems desirable to realize change rates that are as large as possible—achieving a fast change and consequently a short adaptation time. On the other hand, there is an upper bound to the realized change rates due to the finite amount of selection information. Greater changes cannot rely on valid information and lead to stochastic behavior. (This holds for any adaptation mechanism.) As a simple consequence, the change rate of a single strategy parameter must decrease with an increasing number of strategy parameters to be adapted, assuming a certain constant selection scheme.

  In MSC, the change rate of strategy parameters obviously depends (more or less directly) on the mutation strength on the strategy parameter level. Based on this observation, considerable theoretical efforts were made to calculate the optimal mutation strength for the global step size (Schwefel, 1995; Beyer, 1996b). But, in general, the conflict between an optimal change rate versus a significant selection difference (see above) cannot be resolved by choosing an ambiguous compromise for the mutation strength on the strategy parameter level (Ostermeier et al., 1994a). The mutation strength that achieves an optimal change rate is usually smaller than the mutation strength that achieves a suitable selection difference. The discrepancy increases with increasing problem dimension and with increasing number of strategy parameters to be adapted. One idea of derandomization is to explicitly unlink the change rate from the mutation strength resolving this discrepancy.

  Parent number μ and the recombination procedure have a great influence on the possible change rate of the strategy parameters between (two) generations. Assuming a certain mutation strength on the strategy parameter level, the possible change rate can be tuned downwards by increasing μ. This is most obvious for intermediate multi-recombination: The mean change of μ recombined individuals is approximately $\sqrt{\mu}$ times smaller than the mean change of a single individual. Within the concept of MSC, choosing μ and an appropriate recombination mechanism is the only way to tune the change rate independently of the mutation strength (downwards). Therefore, it is not a surprising observation that a successful strategy parameter adaptation in the concept of MSC strongly depends on a suitable choice of μ: In our experience μ has to scale linearly with the number of strategy parameters to be adapted. One objective of derandomization is to facilitate a reliable and fast adaptation of strategy parameters independent of the population size, even in small populations.

We start our discussion of derandomization from the ES with mutative strategy parameter control formulated in Equations (1) and (2). For the sake of simplicity, we still consider adaptation of one global step size only. (The technique of derandomization becomes especially important if a large number of strategy parameters has to be adapted.) The first level of derandomization (Ostermeier et al., 1994a) facilitates independent control of mutation strength and change rate of strategy parameters. This can

be achieved by slight changes in the formulation of the mutation step. For $k = 1, \ldots, \lambda$,

$$\sigma_k^{(g+1)} = \sigma^{(g)} \exp\left(\frac{\xi_k}{d}\right) \tag{3}$$

$$\boldsymbol{x}_k^{(g+1)} = \boldsymbol{x}^{(g)} + \sigma^{(g)} \exp\left(\xi_k\right) \boldsymbol{z}_k \ , \tag{4}$$

where $d \geq 1$ is the damping parameter, and the symbols from Equations (1) and (2) in Section 1 are used. This can still be regarded as a mutative approach to strategy parameter control: If $d = 1$, Equations (3) and (4) are identical to (1) and (2), because $\sigma^{(g)} \exp\left(\xi_k\right)$ in Equation (4) equals $\sigma_k^{(g+1)}$ in Equation (2). Enlarging the damping parameter d reduces the change rate between $\sigma^{(g)}$ and $\sigma_k^{(g+1)}$ leaving the selection relevant mutation strength on the strategy parameter level (the standard deviation of ξ_k in Equation (4)) unchanged. Instead of choosing the standard deviation to be $1/\sqrt{2n}$ and $d = 1$, as in the purely mutative approach, it is more sensible to choose the standard deviation $1/4$ and $d = \sqrt{n/8}$, assuming $n \geq 8$, facilitating the same rate of change with a larger mutation strength. Large values for d scale down fluctuations that occur due to stochastic effects, but decrease the possible change rate. For standard deviation $1/4$ within the interval $1 \leq d \lessapprox \sqrt{n/4}$, neither the fluctuations for small d nor the limited change rate for large d decisively worsen the performance on the sphere objective function $\sum_i x_i^2$. But if n or even n^2 strategy parameters have to be adapted, the trade-off between large fluctuations and a large change rate come the fore. Then the change rate must be tuned carefully.

As Rechenberg (1994) and Beyer (1998) pointed out, this concept—mutating big and inheriting small—can easily be applied to the object parameter mutation as well. It will be useful in the case of distorted selection on the object parameter level and is implicitly implemented by intermediate multi-recombination.

In general, the first level of derandomization yields the following effects:

- Mutation strength (in our example, standard deviation of ξ_k) can be chosen comparatively large in order to reduce the disturbance of strategy parameter selection.
- The ratio between change rate and mutation strength can be tuned downwards (in our example, by means of d) according to the problem dimension and the number of strategy parameters to be adapted. This scales down stochastic fluctuations of the strategy parameters.
- μ and λ can be chosen independent of the adaptation mechanism. They, in particular, become independent of the number of strategy parameters to be adapted.

The second level of derandomization completely removes the selection disturbance on the strategy parameter level. To achieve this, mutation on the strategy parameter level by means of sampling the random number ξ_k in Equation (3) is omitted. Instead, the realized step length $||\boldsymbol{z}||$ is used to change σ. This leads to

$$\sigma_k^{(g+1)} = \sigma^{(g)} \exp\left(\frac{||\boldsymbol{z}_k|| - \mathrm{E}[||\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})||]}{d}\right) \tag{5}$$

$$\boldsymbol{x}_k^{(g+1)} = \boldsymbol{x}^{(g)} + \sigma^{(g)} \boldsymbol{z}_k \tag{6}$$

$||\boldsymbol{z}_k|| - \mathrm{E}[||\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})||]$ in Equation (5) replaces ξ_k in Equation (3). Note that $\mathrm{E}[||\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})||]$ is the expectation of $||\boldsymbol{z}_k||$ under random selection.[6] Equation (5) is not intended to

[6] Random selection occurs if the objective function returns random numbers independent of $\boldsymbol{x}$.

be used in practice, because the selection relevance of $||\boldsymbol{z}_k||$ vanishes for increasing n.[7] One can interpret Equation (5) as a special mutation on the strategy parameter level or as an adaptation procedure for σ (which must only be done for selected individuals). Now, all stochastic variation of the object parameters—namely originated by the random vector realization $\boldsymbol{z}_k$ in Equation (6)—is used for strategy parameter adaptation in Equation (5). In other words, the actually realized mutation step on the object parameter level is used for strategy parameter adjustment. If $||\boldsymbol{z}_k||$ is smaller than expected, $\sigma^{(g)}$ is decreased. If $||\boldsymbol{z}_k||$ is larger than expected $\sigma^{(g)}$ is increased. Selecting a small/large mutation step directly leads to reduction/enlargement of the step size.

The disadvantage of Equation (5) compared to Equation (1) is that it cannot be expanded easily for the adaptation of other strategy parameters. The expansion to the adaptation of all distribution parameters is the subject of this paper.

In general, the completely derandomized adaptation follows the following principles:

1. The mutation distribution is changed, so that the probability to produce the selected mutation step (again) is increased.

2. There is an explicit control of the change rate of strategy parameters (in our example, by means of d).

3. Under random selection, strategy parameters are stationary. In our example, expectation of $\log \sigma^{(g+1)}$ equals $\log \sigma^{(g)}$.

In the next section, we discuss the adaptation of all variances and covariances of the mutation distribution.

3 Adapting Arbitrary Normal Mutation Distributions

We give two motivations for the adaptation of arbitrary normal mutation distributions with zero mean.

- The suitable encoding of a given problem is crucial for an efficient search with an evolutionary algorithm.[8] Desirable is an adaptive encoding mechanism. To restrict such an adaptive encoding/decoding to linear transformations seems reasonable: Even in the linear case, $\mathcal{O}(n^2)$ parameters have to be adapted. It is easy to show that in the case of additive mutation, linear transformation of the search space (and search points accordingly) is equivalent to linear transformation of the mutation distribution. Linear transformation of the $(\boldsymbol{0}, \boldsymbol{I})$-normal mutation distribution always yields a normal distribution with zero mean, while any normal distribution with zero mean can be produced by a suitable linear transformation. This yields equivalence between an adaptive general linear encoding/decoding and the adaptation of all distribution parameters in the covariance matrix.

- We assume the objective function to be non-linear and significantly non-separable—otherwise the search problem is comparatively easy, because it can be solved by solving n one-dimensional problems. Adaptation to non-separable objective functions is not possible if only individual step sizes are adapted. This

[7] For an efficient, completely derandomized global step size adaptation, the evolution path $\boldsymbol{p}_k^{(g+1)} = (1-c)\boldsymbol{p}^{(g)} + \sqrt{c(2-c)}\,\boldsymbol{z}_k$, where $2/(n+2) \leq c \leq \sqrt{2/(n+1)}$, must be used instead of $\boldsymbol{z}_k$ in Equation (5). This method is referred to as *cumulative path length control* and implemented with Equations (16) and (17).

[8] In biology, this is equivalent to a suitable genotype-phenotype transformation.

demands a more general mutation distribution and suggests the adaptation of arbitrary normal mutation distributions.

Choosing the mean value to be zero seems to be self-evident and is in accordance with Beyer and Deb (2000). The current parents must be regarded as the best approximation to the optimum known so far. Using nonzero mean is equivalent to moving the population to another place in parameter space. This can be interpreted as extrapolation. We feel extrapolation is an anti-thesis to a basic paradigm of evolutionary computation. Of course, compared to simple ESs, extrapolation should gain a significant speed up on many test functions. But, compared to an ES with derandomized strategy parameter control of the complete covariance matrix, the advantage from extrapolation seems small. Correspondingly, we find algorithms with nonzero mean (Ostermeier, 1992; Ghozeil and Fogel, 1996; Hildebrand et al., 1999) unpromising.

In our opinion, any adaptation mechanism that adapts a general linear encoding/decoding has to meet the following fundamental demands:

Adaptation: The adaptation must be successful in the following sense: After an adaptation phase, progress-rates comparable to those on the (hyper-)sphere objective function $\sum_i x_i^2$ must be realized on any convex-quadratic objective function.[9] This must hold especially for large problem conditions (greater than 10^4) *and* non-systematically oriented principal axes of the objective function.

Performance: The performance (with respect to objective function evaluations) on the (hyper-)sphere function should be comparable to the performance of the $(1, 5)$-ES with optimal step size, where $\ln(f_{\text{best}}^{(g)}/f_{\text{best}}^{(g+n)})/\lambda \approx 0.14$.[10] A loss by a factor up to ten seems to be acceptable.

Invariance: The invariance properties of the $(1, \lambda)$-ES with isotropic mutation distribution with respect to transformation of object parameter space and function value should be preserved. In particular, translation, rotation, and reflection of object parameter space (and initial point accordingly) should have no influence on strategy behavior as well as any strictly monotonically increasing, i.e., order-preserving transformation of the objective function value.

Stationarity: To get a reasonable strategy behavior in cases of low selection pressure, some kind of stationarity condition on the strategy parameters should be fulfilled under purely random selection. This is analogous to choosing the mean value to be zero for the object parameter mutation. At least non-stationarity must be quantified and judged.

From a conceptual point of view, one primary aim of strategy parameter adaptation is to become invariant (modulo initialization) against certain transformations of the object parameter space. For any fitness function $f : \boldsymbol{x} \mapsto f(c \cdot \boldsymbol{A} \cdot \boldsymbol{x})$, the global step size adaptation can yield invariance against $c \neq 0$. An adaptive general linear

[9] That is any function $f : \boldsymbol{x} \mapsto \boldsymbol{x}^{\mathrm{T}} \boldsymbol{H} \boldsymbol{x} + \boldsymbol{b}^{\mathrm{T}} \boldsymbol{x} + c$, where the Hessian matrix $\boldsymbol{H} \in \mathbb{R}^{n \times n}$ is symmetric and positive definite, $\boldsymbol{b} \in \mathbb{R}^n$ and $c \in \mathbb{R}$. Due to numerical requirements, the condition of $\boldsymbol{H}$ may be restricted to, e.g., 10^{14}. We use the expectation of $\frac{n}{k} \ln((f_{\text{best}}^{(g)} - f^*)/(f_{\text{best}}^{(g+k)} - f^*))$ with $k \in \mathbb{N}$ as the progress measure (where f^* is the fitness value at the optimum), because it yields comparable values independent of $\boldsymbol{H}$, $\boldsymbol{b}$, c, n, and k. For $\boldsymbol{H} = \boldsymbol{I}$ and large n, this measure yields values close to the common normalized progress measure $\varphi^\star := \frac{n}{r^{(g)}}(r^{(g)} - \mathrm{E}[r^{(g+1)}])$ (Beyer, 1995), where r is the distance to the optimum.

[10] Not taking into account weighted recombination, the theoretically optimal $(1 + 1)$-ES yields approximately 0.20.

encoding/decoding can additionally yield invariance against the full rank $n \times n$ matrix $\boldsymbol{A}$. This invariance is a good starting point for achieving the adaptation demand. Also, from a practical point of view, invariance is an important feature for assessing a search algorithm (compare Section 6). It can be interpreted as a feature of robustness. Therefore, even to achieve an additional invariance seems to be very attractive.

Keeping in mind the stated demands, we discuss two approaches to the adaptation of arbitrary normal mutation distributions in the next two sections.

3.1 A (Purely) Mutative Approach: The Rotation Angle Adaptation

Schwefel (1981) proposed a mutative approach to the adaptation of arbitrary normal mutation distributions with zero mean, often referred to as *correlated mutations*. The distribution is parameterized by n standard deviations σ_i and $(n^2-n)/2$ rotation angles α_j that undergo a selection-recombination-mutation scheme.

The algorithm of a $(\mu/\rho_{\mathrm{I}}, \lambda)$-CORR-ES as used in Section 7.2 is briefly described. At generation $g = 1, 2, \ldots$ for each offspring ρ parents are selected. The (component wise) arithmetic mean of step sizes, angles, and object parameter vectors of the ρ parents, denoted with $\sigma^{\mathrm{rec}}_{i=1,\ldots,n}$, $\alpha^{\mathrm{rec}}_{j=1,\ldots,(n^2-n)/2}$, and $\boldsymbol{x}^{\mathrm{rec}}$, are starting points for the mutation. Step sizes and angles read component wise

$$\sigma_i^{(g+1)} = \sigma_i^{\mathrm{rec}} \cdot \exp\left(\mathcal{N}\left(0, \frac{1}{2n}\right) + \mathcal{N}_i\left(0, \frac{1}{2\sqrt{n}}\right)\right) \tag{7}$$

$$\alpha_j^{(g+1)} = \left(\alpha_j^{\mathrm{rec}} + \mathcal{N}_j\left(0, \left(\frac{5}{180}\pi\right)^2\right) + \pi\right) \bmod 2\pi - \pi \tag{8}$$

The random number $\mathcal{N}(0, 1/(2n))$ in Equation (7), which denotes a normal distribution with zero mean and standard deviation $\sqrt{1/(2n)}$, is only drawn once for all $i = 1, \ldots, n$. The modulo operation ensures the angles to be in the interval $-\pi \leq \alpha_j^{(g+1)} < \pi$, which is, in our experience, of minor relevance. The chosen distribution variances reflect the typical parameter setting (see Bäck and Schwefel (1993)). The object parameter mutation reads

$$\boldsymbol{x}^{(g+1)} = \boldsymbol{x}^{\mathrm{rec}} + R\left(\alpha_1^{(g+1)}, \ldots, \alpha_{(n^2-n)/2}^{(g+1)}\right) \cdot \begin{pmatrix} \sigma_1^{(g+1)} & & \\ & \ddots & \\ & & \sigma_n^{(g+1)} \end{pmatrix} \cdot \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{9}$$

The spherical $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ distribution is multiplied by a diagonal step size matrix determined by $\sigma_1^{(g+1)}, \ldots, \sigma_n^{(g+1)}$. The resulting axis-parallel mutation ellipsoid is newly oriented by successive rotations in all (i.e., $(n^2-n)/2$) two-dimensional subspaces spanned by canonical unit vectors. This complete rotation matrix is denoted with $R(.)$. This algorithm allows one to generate any n-dimensional normal distribution with zero mean (Rudolph, 1992).

It is generally recognized that the typical values for $\mu = 15$ and $\lambda = 100$ are not sufficient for this adaptation mechanism (Bäck et al., 1997). Due to the mutative approach, the parent number has presumably to scale with n^2. Choosing $\mu \approx n^2/2$, which is roughly the number of free strategy parameters, performance on the sphere problem declines with increasing problem dimension and becomes unacceptably poor for $n \gtrapprox 10$. The performance demand cannot be met. This problem is intrinsic to the (purely) mutative approach.

The parameterization by rotation angles causes the following effects:

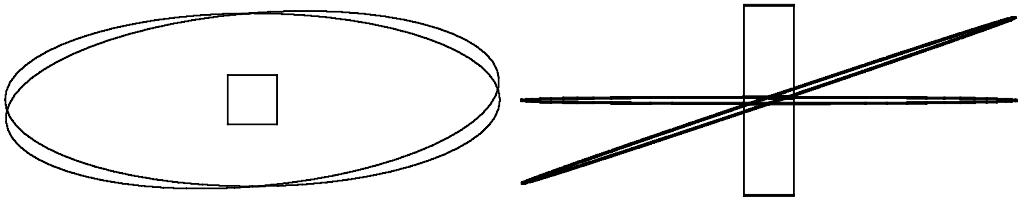

Figure 2: Lines of equal probability density of two normal distributions ($n = 2$) with axis ratios of $1:3$ (left) and $1:300$ (right) before and after rotation by $\frac{5}{180}\pi$. The right figure is enlarged in the y-direction; the rectangle covers the identical area in the left and in the right. Comparing the cross section areas of the ellipsoids, the distributions with axis ratio of $1:3$ are very similar, while the distributions with axis ratio $1:300$ differ substantially.

- The effective mutation strength strongly depends on the position in strategy parameter space. Mutation ellipsoids with high axis ratios are mutated much more strongly than those with low axis ratios: Figure 2 shows two distributions with different axis ratios before and after rotation by the typical variance. The resulting cross section area, a simple measure of their similarity, differs in a wide range. For axis ratio $1:1$ (spheres, not shown) it is 100%. For axis ratio $1:3$ (Figure 2 left) it is about 90%, while it decreases to roughly 5% for axis ratio $1:300$ (Figure 2 right). The section area tends to become zero for increasing axis ratios. The mutation strength (on the rotation angles) cannot be adjusted independently of the actual position in strategy parameter space.

- In principle, invariance against a new orientation of the search space is lost, because rotation planes are chosen with respect to the given coordinate system. In simulations, we found the algorithm to be dependent even on permutations of the coordinate axes (Hansen et al., 1995; Hansen, 2000)! Furthermore, its performance depends highly on the orientation of the given coordinate system (compare Section 7.2). The invariance demand is not met.

Taking into account these difficulties, it is not surprising to observe that the adaptation demand is not met. Progress rates on convex-quadratic functions with high axis ratios can be several orders of magnitude lower than progress rates achieved on the sphere problem (Holzheuer, 1996; Hansen, 2000, and Section 7.2).

When the typical intermediate recombination is applied to the step sizes, they increase unbounded under random selection. The systematic drift is slow, usually causes no problems, and can even be an advantage in some situations. Therefore, we assume the stationarity demand to be met.

Using another parameterization together with a suitable mutation operator can solve the demands for adaptation and invariance without giving up the concept of MSC (Ostermeier and Hansen, 1999). To satisfy the performance demand the concept of MSC has to be modified.

3.2 A Completely Derandomized Approach: The Covariance Matrix Adaptation (CMA)

The covariance matrix adaptation (Hansen and Ostermeier, 1996) is a second-level (i.e., completely) derandomized self-adaptation scheme. First, it directly implements the

aim of MSC to raise the probability of producing successful mutation steps: The covariance matrix of the mutation distribution is changed in order to increase the probability of producing the selected mutation step again. Second, the rate of change is adjusted according to the number of strategy parameters to be adapted (by means of c_{cov} in Equation (15)). Third, under random selection, the expectation of the covariance matrix $\boldsymbol{C}$ is stationary. Furthermore, the adaptation mechanism is inherently independent of the given coordinate system. In short, the CMA implements a principle component analysis of the previously selected mutation steps to determine the new mutation distribution. We give an illustrative description of the algorithm.

Consider a special method to produce realizations of a n-dimensional normal distribution with zero mean. If the vectors $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m \in \mathbb{R}^n$, $m \geq n$, span $\mathbb{R}^n$, and $\mathcal{N}(0,1)$ denotes independent $(0,1)$-normally distributed random numbers, then

$$\mathcal{N}(0,1)\,\boldsymbol{z}_1 + \ldots + \mathcal{N}(0,1)\,\boldsymbol{z}_m \tag{10}$$

is a normally distributed random vector with zero mean.[11] Choosing $\boldsymbol{z}_i$, $i = 1, \ldots, m$, appropriate any normal distribution with zero mean can be realized.

The distribution in Equation (10) is generated by adding the "line distributions" $\mathcal{N}(0,1)\,\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{z}_i\boldsymbol{z}_i^{\mathrm{T}})$. If the vector $\boldsymbol{z}_i$ is given, the normal (line) distribution with zero mean, which produces the vector $\boldsymbol{z}_i$ with the highest probability (density) of all normal distributions with zero mean, is $\mathcal{N}(\boldsymbol{0}, \boldsymbol{z}_i\boldsymbol{z}_i^{\mathrm{T}})$. (The proof is simple and omitted.)

The covariance matrix adaptation (CMA) constructs the mutation distribution out of selected mutation steps. In place of the vectors $\boldsymbol{z}_i$ in Equation (10), the selected mutation steps are used with exponentially decreasing weights. An example is shown in Figure 3, where $n = 2$. The isotropic initial distribution is constructed by means of the unit vectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. At every generation $g = 1, 2, \ldots$, the selected mutation step $\boldsymbol{z}_{\text{sel}}^{(g)}$ is added to the vector tuple. All other vectors are multiplied by a factor $q < 1$.[12] According to Equation (10), after generation $g = 3$, the distribution reads

$$\mathcal{N}(0,1)\,q^3\boldsymbol{e}_1 + \mathcal{N}(0,1)\,q^3\boldsymbol{e}_2 + \mathcal{N}(0,1)\,q^2\boldsymbol{z}_{\text{sel}}^{(1)} + \mathcal{N}(0,1)\,q\boldsymbol{z}_{\text{sel}}^{(2)} + \mathcal{N}(0,1)\,\boldsymbol{z}_{\text{sel}}^{(3)} \tag{11}$$

This mutation distribution tends to reproduce the mutation steps selected in the past generations. In the end, it leads to an alignment of the distributions before and after selection, i.e., an alignment of the recent mutation distribution with the distribution of the selected mutation steps. If both distributions become alike, as under random selection, in expectation, no further change of the distributions takes place (Hansen, 1998).

This illustrative but also formally precise description of the CMA differs in three points from the CMA-ES formalized in Section 5. These extensions are as follows:

- Apart from the adaptation of the distribution shape, the overall variance of the mutation distribution is adapted separately. We found the additional adaptation of the overall variance useful for at least two reasons. First, changes of the overall variance and of the distribution shape should operate on different time scales. Due to the number of parameters to be adapted, the adaptation of the covariance matrix, i.e., the adaptation of the distribution shape, must operate on a time scale of n^2. Adaptation of the overall variance should operate on a time scale n, because the variance should be able to change as fast as required on simple objective functions

[11] The covariance matrix of this normally distributed random vector reads $\boldsymbol{z}_1\boldsymbol{z}_1^{\mathrm{T}} + \ldots + \boldsymbol{z}_m\boldsymbol{z}_m^{\mathrm{T}}$.

[12] q adjusts the change rate, and q^2 corresponds to $1 - c_{\text{cov}}$ in Equation (15).

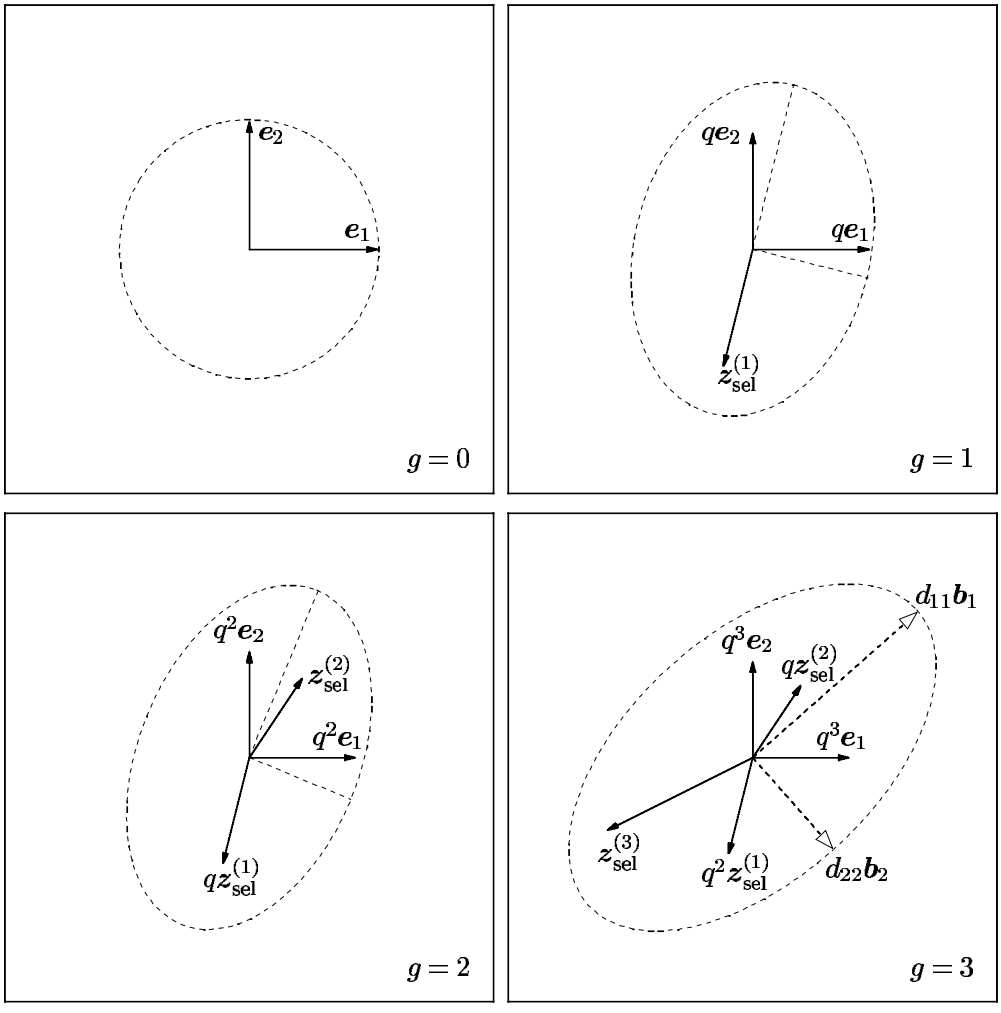

Figure 3: Construction of the mutation distribution in the CMA, where $n = 2$. The initial configuration (upper left) consists of the orthogonal unit vectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. They produce an isotropic mutation distribution (dashed). The vectors $\boldsymbol{z}_{\text{sel}}^{(1)}, \boldsymbol{z}_{\text{sel}}^{(2)}$, and $\boldsymbol{z}_{\text{sel}}^{(3)}$ are added successively at generations $g = 1, 2, 3$, while old vectors are multiplied by $q = 0.91$. The covariance matrix of the distribution after generation $g = 3$ reads $\boldsymbol{C}^{(3)} = q^6\boldsymbol{e}_1\boldsymbol{e}_1^{\mathrm{T}} + q^6\boldsymbol{e}_2\boldsymbol{e}_2^{\mathrm{T}} + \sum_{i=1}^{3} q^{2(3-i)}\boldsymbol{z}_{\text{sel}}^{(i)}\left(\boldsymbol{z}_{\text{sel}}^{(i)}\right)^{\mathrm{T}} = d_{11}^2\boldsymbol{b}_1\boldsymbol{b}_1^{\mathrm{T}} + d_{22}^2\boldsymbol{b}_2\boldsymbol{b}_2^{\mathrm{T}}$. The numbers d_{11}^2 and d_{22}^2 are eigenvalues of $\boldsymbol{C}^{(3)}$; $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are the corresponding eigenvectors with unit length. Realizations from the distribution can be drawn with Equation (11) or more easily with $\mathcal{N}(0,1)\, d_{11}\boldsymbol{b}_1 + \mathcal{N}(0,1)\, d_{22}\boldsymbol{b}_2$.

like the sphere $\sum_i x_i^2$. Second, if overall variance is not adapted faster than the distribution shape, an (initially) small overall variance can jeopardize the search process. The strategy "sees" a linear environment, and adaptation (erroneously) enlarges the variance in one direction.

- The CMA is formulated for parent number $\mu \geq 1$ and weighted multi-recombination, resulting in a more sophisticated computation of $\boldsymbol{z}_{\text{sel}}^{(g)}$.
- The technique of cumulation is applied to the vectors, that construct the distribution. Instead of the single mutation steps $\boldsymbol{z}_{\text{sel}}^{(g)}$, evolution paths $\boldsymbol{p}^{(g)}$ are used. An evolution path $\boldsymbol{p}^{(g)}$ represents a sequence of selected mutation steps. The technique of cumulation is motivated in the next section.

The adaptation scheme is formalized by means of the covariance matrix of the distribution, because storing as many as g vectors is not practicable. That is, in every generation g, after selection of the best search points has taken place,

1. The covariance matrix of the new distribution $\boldsymbol{C}^{(g)}$ is calculated due to $\boldsymbol{C}^{(g-1)}$ and the vector $\boldsymbol{z}_{\text{sel}}^{(g)}$ (with cumulation $\boldsymbol{p}^{(g)}$). In other words, the covariance matrix of the distribution is adapted.
2. Overall variance is adapted by means of the cumulative path length control. Referring to Equation (5), $\boldsymbol{z}_{\boldsymbol{k}}$ is replaced by a "conjugate" evolution path ($\boldsymbol{p}_{\sigma}$ in Equations (16) and (17)).
3. From the covariance matrix, the principal axes of the new distribution and their variances are calculated. To produce realizations from the new distribution the n line distributions are added, which correspond to the principal axes of the mutation distribution ellipsoid (compare Figure 3).

Storage requirements are $\mathcal{O}(n^2)$. We note, that "generally, for any moderate n, this is an entirely trivial disadvantage" (Press et al., 1992). For computational and numerical requirements, refer to Sections 5.2 and 7.1.

4 Utilizing the Evolution Path: Cumulation

The concept of MSC utilizes selection information of a single generation step. In contrast, it can be useful to utilize a whole path taken by the population over a number of generations. We call such a path an *evolution path* of the ES. The idea of the evolution path is similar to the idea of isolation. The performance of a strategy can be evaluated significantly better after a couple of steps are taken than after a single step. It is worth noting that both quantity *and* quality of the evaluation basis can improve.

Accordingly, to reproduce successful evolution paths seems more promising than to reproduce successful single mutation steps. The former is more likely to maximize a progress rate while the latter emphasizes the selection probability. Consequently, we expand the idea of strategy parameter control: The adaptation should maximize the probability to reproduce successful, i.e., actually taken, evolution paths rather than single mutation steps (Hansen and Ostermeier, 1996).

An evolution path contains additional selection information compared to single steps – correlations between single steps successively taken in the generation sequence are utilized. If successively selected mutation steps are parallel correlated (scalar product greater than zero), the evolution path will be comparatively long. If the steps are anti-parallel correlated (scalar product less than zero), the evolution path will be comparatively short. Parallel correlation means that successive steps are going in the same direction. Anti-parallel correlation means that the steps cancel each other out. We assume both correlations to be inefficient. This is most obvious if the correlation/anti-correlation between successive steps is perfect. These steps can exactly be replaced with the enlarged/reduced first step.

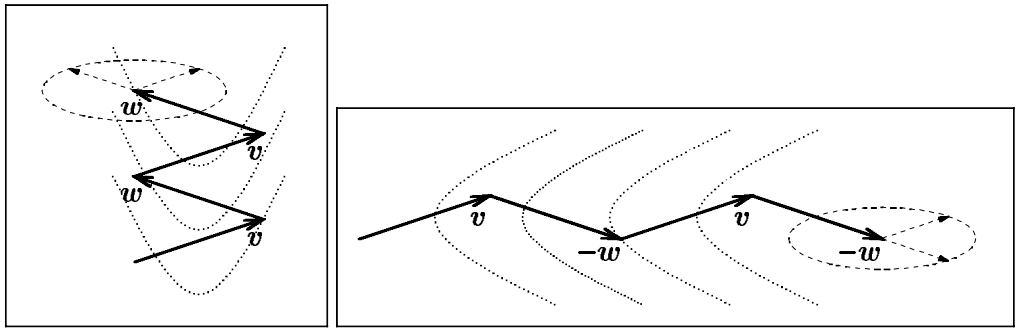

Figure 4: Two idealized evolution paths (solid) in a ridge topography (dotted). The distributions constructed by the *single* steps (dashed, reduced in size) are identical.

Consequently, to maximize mutation step efficiency, it is necessary to realize longer steps in directions where the evolution path is long—here the same distance can be covered by fewer but longer steps. On the other side, it is appropriate to realize shorter steps in directions where the evolution path is short. Both can be achieved using evolution paths rather than single mutation steps for the construction of the mutation distribution as described in Section 3.2.

We calculate an evolution path within an iterative process by (weighted) summation of successively selected mutation steps. The evolution path $\boldsymbol{p} \in \mathbb{R}^n$ obeys

$$\boldsymbol{p}^{(g+1)} = (1-c)\,\boldsymbol{p}^{(g)} + \sqrt{c(2-c)}\,\boldsymbol{z}_{\text{sel}}^{(g+1)} \quad , \tag{12}$$

where $0 < c \leq 1$ and $\boldsymbol{p}^{(0)} = \mathbf{0}$. This procedure, introduced in Ostermeier et al. (1994b), is called *cumulation*. $\sqrt{c(2-c)}$ is a normalization factor: Assuming $\boldsymbol{z}_{\text{sel}}^{(g+1)}$ and $\boldsymbol{p}^{(g)}$ in Equation (12) to be independent and identically distributed yields $\boldsymbol{p}^{(g+1)} \sim \boldsymbol{p}^{(g)}$ independently of $c \in]0;1]$. Variances of $\boldsymbol{p}^{(g+1)}$ and $\boldsymbol{p}^{(g)}$ are identical because $1^2 = (1-c)^2 + \sqrt{c(2-c)}^2$. If $c = 1$, no cumulation takes place, and $\boldsymbol{p}^{(g+1)} = \boldsymbol{z}_{\text{sel}}^{(g+1)}$. The life span of information accumulated in $\boldsymbol{p}^{(g)}$ is roughly $1/c$ (Hansen, 1998): After about $1/c$ generations the original information in $\boldsymbol{p}^{(g)}$ is reduced by the factor $1/\mathrm{e} \approx 0.37$. That means c^{-1} can roughly be interpreted as the number of summed steps.

The benefit from cumulation is shown with an idealized example. Consider the two different sequences of four consecutive generation steps in Figure 4. Compare any evolution path, that is, any sum of consecutive steps in the left and in the right of the figure. They differ significantly with respect to direction and length. Notice that the difference is only due to the sign of vectors two and four.

Construction of a distribution from the single mutation steps (single vectors in the figure) according to Equation (10) leads to exactly the same result in both cases (dashed). Signs of constructing vectors do not matter, because the vectors are multiplied by a $(0,1)$-normally distributed random number that is symmetrical about zero.

The situation changes when cumulation is applied. We focus on the left of Figure 4. Consider a continuous sequence of the shown vectors—i.e., an alternating sequence of the two vectors $\boldsymbol{v}$ and $\boldsymbol{w}$. Then $\boldsymbol{z}_{\text{sel}}^{(2i-1)} = \boldsymbol{v}$ and $\boldsymbol{z}_{\text{sel}}^{(2i)} = \boldsymbol{w}$ for $i = 1, 2, \ldots$, and according to Equation (12), the evolution path $\boldsymbol{p}^{(g)}$ alternately reads

$$\boldsymbol{p}_{\mathbf{v}}^{(g)} := \sqrt{c(2-c)}\left(\boldsymbol{v} + (1-c)\boldsymbol{w} + (1-c)^2\boldsymbol{v} + (1-c)^3\boldsymbol{w} + \ldots + (1-c)^{g-1}\boldsymbol{z}_{\text{sel}}^{(1)}\right)$$

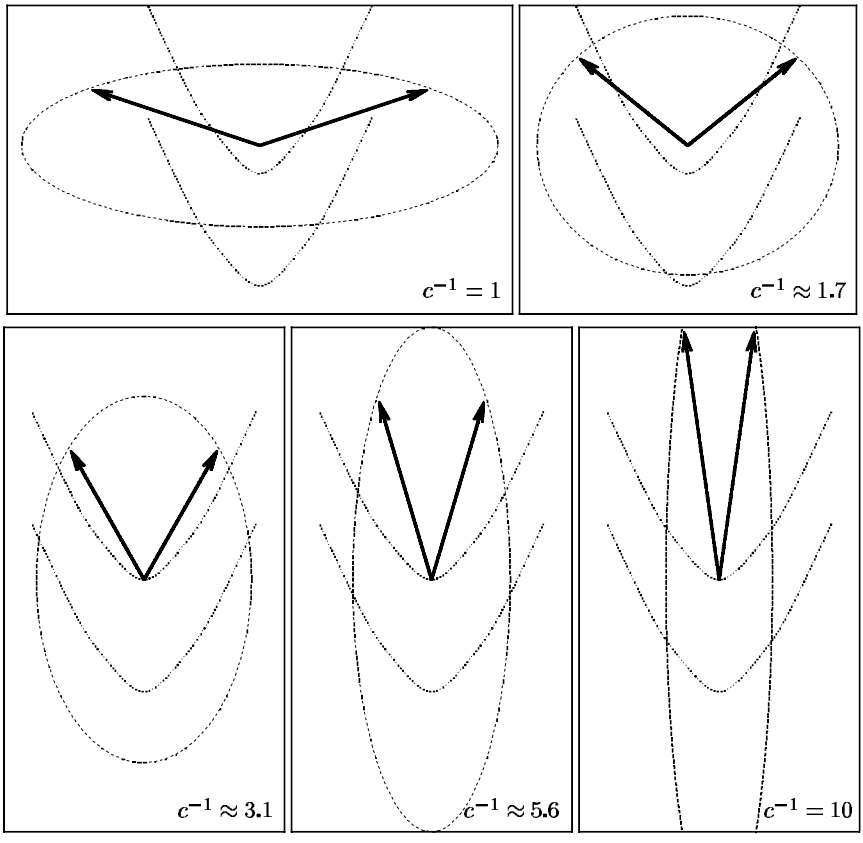

Figure 5: Resulting cumulation vectors $\boldsymbol{p}_{\mathbf{v}}$ and $\boldsymbol{p}_{\mathbf{w}}$ and distribution ellipsoids for an alternating selection of the two vectors $\boldsymbol{v}$ and $\boldsymbol{w}$ from Figure 4, left. For $c = 1$, no cumulation takes place, and $\boldsymbol{p}_{\mathbf{v}} = \boldsymbol{v}$ and $\boldsymbol{p}_{\mathbf{w}} = \boldsymbol{w}$ (upper left). Shown is the result for values of $c^{-1} = 1; \sqrt{3}; \sqrt{10}; \sqrt{30}; \sqrt{100}$, respectively.

at odd generation number g and

$$\boldsymbol{p}_{\mathbf{w}}^{(g)} := \sqrt{c(2-c)} \left(\boldsymbol{w} + (1-c)\boldsymbol{v} + (1-c)^2\boldsymbol{w} + (1-c)^3\boldsymbol{v} + \ldots + (1-c)^{g-1}\boldsymbol{z}_{\text{sel}}^{(1)} \right)$$

at even generation number. For $g \to \infty$ we get

$$\begin{aligned} \boldsymbol{p}_{\mathbf{v}}^{(g)} \to \boldsymbol{p}_{\mathbf{v}} &= \frac{1}{\sqrt{c(2-c)}} (\boldsymbol{v} + (1-c)\boldsymbol{w}) \qquad \text{and} \\ \boldsymbol{p}_{\mathbf{w}}^{(g)} \to \boldsymbol{p}_{\mathbf{w}} &= \frac{1}{\sqrt{c(2-c)}} (\boldsymbol{w} + (1-c)\boldsymbol{v}) \end{aligned}$$

After $3/c$ generations, the deviation from these limits is under 5%.

To visualize the effect of cumulation, these vectors and the related distributions are shown in Figure 5 for different values of c. With increasing life span c^{-1}, the vectors $\boldsymbol{p}_{\mathbf{v}}$ and $\boldsymbol{p}_{\mathbf{w}}$ become more and more similar. The distribution scales down in the horizontal direction, increasing in the vertical direction. Already for a life span of approximately three generations, the effect appears to be very substantial. Notice again that this effect

is due to correlations between consecutive steps. This information cannot be detected utilizing single mutation steps separately. Because additional information is evaluated, cumulation can make adaptation of the distribution parameters faster *and* more reliable.

In our research, we repeatedly found cumulation to be a very useful concept. In the following CMA-ES algorithm, we use cumulation in two places: primarily for the cumulative path length control, where cumulation takes place in Equation (16), and the step size σ is adapted from the generated evolution path $\boldsymbol{p}_\sigma$ in Equation (17); second, for the adaptation of the mutation distribution as exemplified in this section and taking place in Equation (14).

5 The $(\mu_\mathrm{W}, \lambda)$-CMA-ES Algorithm

We formally describe an evolution strategy utilizing the methods illustrated in Sections 3.2 and 4, denoted by CMA-ES.[13] Based on a suggestion by Ingo Rechenberg (1998), weighted recombination from the μ best out of λ individuals is applied, denoted with $(\mu_\mathrm{W}, \lambda)$.[14] Weighted recombination is a generalization of intermediate multi-recombination, denoted $(\mu_\mathrm{I}, \lambda)$, where all recombination weights are identical. A MATLAB implementation of the $(\mu_\mathrm{W}, \lambda)$-CMA-ES algorithm is given in Appendix A.

The transition from generation g to $g+1$ of the parameters $\boldsymbol{x}_1^{(g)}, \ldots, \boldsymbol{x}_\lambda^{(g)} \in \mathbb{R}^n$, $\boldsymbol{p}_\mathrm{c}^{(g)} \in \mathbb{R}^n$, $\boldsymbol{C}^{(g)} \in \mathbb{R}^{n \times n}$, $\boldsymbol{p}_\sigma^{(g)} \in \mathbb{R}^n$, and $\sigma^{(g)} \in \mathbb{R}^+$, as written in Equations (13) to (17), completely defines the algorithm. Initialization is $\boldsymbol{p}_\mathrm{c}^{(0)} = \boldsymbol{p}_\sigma^{(0)} = \boldsymbol{0}$ and $\boldsymbol{C}^{(0)} = \boldsymbol{I}$ (unity matrix), while the initial object parameter vector $\langle \boldsymbol{x} \rangle_\mathrm{w}^{(0)}$ and the initial step size $\sigma^{(0)}$ have to be chosen problem dependent. All vectors are assumed to be column vectors.

The object parameter vector $\boldsymbol{x}_k^{(g+1)}$ of individual $k = 1, \ldots, \lambda$ reads

$$\boldsymbol{x}_k^{(g+1)} = \langle \boldsymbol{x} \rangle_\mathrm{w}^{(g)} + \sigma^{(g)} \underbrace{\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{z}_k^{(g+1)}}_{\sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{C}^{(g)}\right)} \quad , \tag{13}$$

where the following apply:

- $\boldsymbol{x}_k^{(g+1)} \in \mathbb{R}^n$, object parameter vector of the k^{th} individual in generation $g+1$.
- $\langle \boldsymbol{x} \rangle_\mathrm{w}^{(g)} := \frac{1}{\sum_{i=1}^{\mu} w_i} \sum_{i=1}^{\mu} w_i \boldsymbol{x}_{i:\lambda}^{(g)}$, $w_i \in \mathbb{R}^+$, weighted mean of the μ best individuals of generation g. The index $i:\lambda$ denotes the i^{th} best individual.
- $\sigma^{(g)} \in \mathbb{R}^+$, step size in generation g. $\sigma^{(0)}$ is the initial component wise standard deviation of the mutation step.
- $\boldsymbol{z}_k^{(g+1)} \in \mathbb{R}^n$, for $k = 1, \ldots, \lambda$ and $g = 0, 1, 2, \ldots$ independent realizations of a $(\boldsymbol{0}, \boldsymbol{I})$-normally distributed random vector. Components of $\boldsymbol{z}_k^{(g+1)}$ are independent $(0,1)$-normally distributed.
- $\boldsymbol{C}^{(g)}$ Symmetrical positive definite $n \times n$-matrix. $\boldsymbol{C}^{(g)}$ is the covariance matrix of the normally distributed random vector $\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. $\boldsymbol{C}^{(g)}$ determines $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$. $\boldsymbol{C}^{(g)} = \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \left(\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)}\right)^\mathrm{T} = \boldsymbol{B}^{(g)} \left(\boldsymbol{D}^{(g)}\right)^2 \left(\boldsymbol{B}^{(g)}\right)^\mathrm{T}$, which is a singular value decomposition of $\boldsymbol{C}^{(g)}$. Initialization $\boldsymbol{C}^{(0)} = \boldsymbol{I}$.

[13] The algorithm described here is identical to the algorithms in Hansen and Ostermeier (1996) setting $\mu = 1$, $c_\mathrm{c} = c_\sigma$, and $d_\sigma = \frac{1}{\beta \hat{\chi}_n}$; in Hansen and Ostermeier (1997) setting $w_1 = \ldots = w_\mu$, $c_\mathrm{c} = c_\sigma$ and $d_\sigma = \frac{1}{D}$; and in Hansen (1998) setting $w_1 = \ldots = w_\mu$.

[14] This notation follows Beyer (1995), simplifying the $(\mu/\mu_\mathrm{I}, \lambda)$-notation for intermediate multi-recombination to $(\mu_\mathrm{I}, \lambda)$, avoiding the misinterpretation of μ and μ_I being different numbers.

$\boldsymbol{D}^{(g)}$ $n \times n$-diagonal matrix (step size matrix). $d_{ij} = 0$ for $i \neq j$ and diagonal elements $d_{ii}^{(g)}$ of $\boldsymbol{D}^{(g)}$ are square roots of eigenvalues of the covariance matrix $\boldsymbol{C}^{(g)}$.

$\boldsymbol{B}^{(g)}$ orthogonal $n \times n$-matrix (rotation matrix) that determines the coordinate system, where the scaling with $\boldsymbol{D}^{(g)}$ takes place. Columns of $\boldsymbol{B}^{(g)}$ are (defined as) the normalized eigenvectors of the covariance matrix $\boldsymbol{C}^{(g)}$. The i^{th} diagonal element of $\boldsymbol{D}^{(g)}$ squared $d_{ii}^{(g)^2}$ is the corresponding eigenvalue to the i^{th} column of $\boldsymbol{B}^{(g)}$, $\boldsymbol{b}_i^{(g)}$. That is, $\boldsymbol{C}^{(g)}\boldsymbol{b}_i^{(g)} = d_{ii}^{(g)^2}\boldsymbol{b}_i^{(g)}$ for $i = 1, \ldots, n$. $\boldsymbol{B}$ is orthogonal, i.e., $\boldsymbol{B}^{-1} = \boldsymbol{B}^{\mathrm{T}}$.

The surfaces of equal probability density of $\boldsymbol{D}^{(g)}\boldsymbol{z}_k^{(g+1)}$ are axis-parallel (hyper-) ellipsoids. $\boldsymbol{B}^{(g)}$ reorients these distribution ellipsoids to become coordinate system independent. The covariance matrix $\boldsymbol{C}^{(g)}$ determines $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$ apart from signs of the columns in $\boldsymbol{B}^{(g)}$ and permutations of columns in both matrices accordingly. Conferring to Equation (13), notice that $\sigma^{(g)}\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ is $(\boldsymbol{0}, \sigma^{(g)^2}\boldsymbol{C}^{(g)})$-normally distributed.

$\boldsymbol{C}^{(g)}$ is adaptated by means of the evolution path $\boldsymbol{p}_{\mathrm{c}}^{(g+1)}$. For the construction of $\boldsymbol{p}_{\mathrm{c}}^{(g+1)}$ the "weighted mean selected mutation step" $\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\langle \boldsymbol{z} \rangle_{\mathrm{w}}^{(g+1)}$ is used. Notice that the step size $\sigma^{(g)}$ is disregarded. The transition of $\boldsymbol{p}_{\mathrm{c}}^{(g)}$ and $\boldsymbol{C}^{(g)}$ reads

$$\boldsymbol{p}_{\mathrm{c}}^{(g+1)} = (1 - c_{\mathrm{c}}) \cdot \boldsymbol{p}_{\mathrm{c}}^{(g)} + c_{\mathrm{c}}^{\mathrm{u}} \cdot \underbrace{c_{\mathrm{w}}\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\langle \boldsymbol{z} \rangle_{\mathrm{w}}^{(g+1)}}_{= \frac{c_{\mathrm{w}}}{\sigma^{(g)}}\left(\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g+1)} - \langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g)}\right)} \tag{14}$$

$$\boldsymbol{C}^{(g+1)} = (1 - c_{\mathrm{cov}}) \cdot \boldsymbol{C}^{(g)} + c_{\mathrm{cov}} \cdot \boldsymbol{p}_{\mathrm{c}}^{(g+1)} \left(\boldsymbol{p}_{\mathrm{c}}^{(g+1)}\right)^{\mathrm{T}} , \tag{15}$$

where the following apply:

$\boldsymbol{p}_{\mathrm{c}}^{(g+1)} \in \mathbb{R}^n$, sum of weighted differences of points $\langle \boldsymbol{x} \rangle_{\mathrm{w}}$. Initialization $\boldsymbol{p}_{\mathrm{c}}^{(0)} = \boldsymbol{0}$. Note that $\boldsymbol{p}_{\mathrm{c}}^{(g+1)} \left(\boldsymbol{p}_{\mathrm{c}}^{(g+1)}\right)^{\mathrm{T}}$ is a symmetrical $n \times n$-matrix with rank one.

$c_{\mathrm{c}} \in]0, 1]$ determines the cumulation time for $\boldsymbol{p}_{\mathrm{c}}$, which is roughly $1/c_{\mathrm{c}}$.

$c_{\mathrm{c}}^{\mathrm{u}} := \sqrt{c_{\mathrm{c}}(2 - c_{\mathrm{c}})}$ normalizes the variance of $\boldsymbol{p}_{\mathrm{c}}$, because $1^2 = (1 - c_{\mathrm{c}})^2 + c_{\mathrm{c}}^{\mathrm{u}2}$ (see Section 4).

$c_{\mathrm{w}} := \frac{\sum_{i=1}^{\mu} w_i}{\sqrt{\sum_{i=1}^{\mu} w_i^2}}$ is chosen that under random selection $c_{\mathrm{w}}\langle \boldsymbol{z} \rangle_{\mathrm{w}}^{(g+1)}$ and $\boldsymbol{z}_k^{(g+1)}$ have the same variance (and are identically distributed).

$\langle \boldsymbol{z} \rangle_{\mathrm{w}}^{(g+1)} := \frac{1}{\sum_{i=1}^{\mu} w_i} \sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}^{(g+1)}$, with $\boldsymbol{z}_k^{(g+1)}$ from Equation (13). The index $i : \lambda$ denotes the index of the i^{th} best individual from $\boldsymbol{x}_1^{(g+1)}, \ldots, \boldsymbol{x}_\lambda^{(g+1)}$. The weights w_i are identical with those for $\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g+1)}$.

$c_{\mathrm{cov}} \in [0, 1[$, change rate of the covariance matrix $\boldsymbol{C}$. For $c_{\mathrm{cov}} = 0$, no change takes place.

Equation (15) realizes the distribution change as exemplified in Section 3.2 and Figure 3.

An additional adaptation of the global step size $\sigma^{(g)}$ is necessary, taking place on a considerably shorter time scale. For the cumulative path length control in the CMA-ES,

a "conjugate" evolution path $\boldsymbol{p}_\sigma^{(g+1)}$ is calculated, where scaling with $\boldsymbol{D}^{(g)}$ is omitted.

$$\boldsymbol{p}_\sigma^{(g+1)} = (1-c_\sigma)\cdot\boldsymbol{p}_\sigma^{(g)} + c_\sigma^{\mathrm{u}}\cdot \underbrace{c_{\mathrm{w}}\boldsymbol{B}^{(g)}\langle\boldsymbol{z}\rangle_{\mathrm{w}}^{(g+1)}}_{=\boldsymbol{B}^{(g)}\left(\boldsymbol{D}^{(g)}\right)^{-1}\left(\boldsymbol{B}^{(g)}\right)^{-1}\frac{c_{\mathrm{w}}}{\sigma^{(g)}}\left(\langle\boldsymbol{x}\rangle_{\mathrm{w}}^{(g+1)}-\langle\boldsymbol{x}\rangle_{\mathrm{w}}^{(g)}\right)} \tag{16}$$

$$\sigma^{(g+1)} = \sigma^{(g)}\cdot\exp\left(\frac{1}{d_\sigma}\cdot\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|-\widehat{\chi}_n}{\widehat{\chi}_n}\right) , \tag{17}$$

where the following apply:

$\boldsymbol{p}_\sigma^{(g+1)} \in \mathbb{R}^n$, evolution path not scaled by $\boldsymbol{D}^{(g)}$. Initialization $\boldsymbol{p}_\sigma^{(0)} = \boldsymbol{0}$.

$c_\sigma \in]0,1[$ determines the cumulation time for $\boldsymbol{p}_\sigma^{(g)}$, which is roughly $1/c_\sigma$.

$c_\sigma^{\mathrm{u}} := \sqrt{c_\sigma(2-c_\sigma)}$ fulfills $(1-c_\sigma)^2 + c_\sigma^{\mathrm{u}2} = 1$ (see Section 4).

$d_\sigma \geq 1$, damping parameter, determines the possible change rate of $\sigma^{(g)}$ in the generation sequence (compare Section 2, in particular Equations (3) and (5)).

$\widehat{\chi}_n = \mathrm{E}[\|\mathcal{N}(\boldsymbol{0},\boldsymbol{I})\|] = \sqrt{2}\cdot\Gamma\left(\frac{n+1}{2}\right)/\Gamma\left(\frac{n}{2}\right)$, expectation of the length of a $(\boldsymbol{0},\boldsymbol{I})$-normally distributed random vector. A good approximation is $\widehat{\chi}_n \approx \sqrt{n}\left(1-\frac{1}{4n}+\frac{1}{21n^2}\right)$ (Ostermeier, 1997).

Apart from omitting the transformation with $\boldsymbol{D}^{(g)}$, Equations (16) and (14) are identical: In (16) we use $\boldsymbol{B}^{(g)}\langle\boldsymbol{z}\rangle_{\mathrm{w}}^{(g+1)}$ for the cumulation, instead of $\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\langle\boldsymbol{z}\rangle_{\mathrm{w}}^{(g+1)}$. Under random selection, $\boldsymbol{B}^{(g)}\langle\boldsymbol{z}\rangle_{\mathrm{w}}$ is $(\boldsymbol{0},\boldsymbol{I})$-normally distributed, independently of $\boldsymbol{C}^{(g)}$. Thereby step lengths in different directions are comparable, and the expected length of $\boldsymbol{p}_\sigma$, denoted $\widehat{\chi}_n$, is well known.[15] The cumulation in Equation (14) often speeds up the adaptation of the covariance matrix $\boldsymbol{C}$ but is not an essential feature (c_{c} can be set to 1). For the path length control, the cumulation is essential (c_σ^{-1} must not be considerably smaller than $\sqrt{n/2}$). With increasing n, the lengths of single steps become more and more similar and therefore more and more selection irrelevant.

While in Equation (13) the product matrix $\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}$ only has to satisfy the equation $\boldsymbol{C}^{(g)} = \boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\left(\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\right)^{\mathrm{T}}$, cumulative path length control in Equation (16) requires its factorization into an orthogonal and a diagonal matrix.

5.1 Parameter Setting

Besides population size λ and parent number μ, the strategy parameters $(w_1,\ldots,w_\mu)$, c_{c}, c_{cov}, c_σ, and d_σ, connected to Equations (13), (14), (15), (16), and (17), respectively, have to be chosen.[16]

The default parameter settings are summarized in Table 1. In general, the selection related parameters μ, λ, and $w_1,\ldots,w_\mu$ are comparatively uncritical and can be chosen in a wide range without disturbing the adaptation procedure. We strongly recommend always choosing $\mu \leq \lambda/2$ and the recombination weights according to $w_1 \geq \ldots \geq w_\mu$. By definition, all weights are greater than zero. In real world applications, the default settings from Table 1 are good first guesses. Only for $n < 10$ does the default value yield $\lambda > n$. For a quick impression, Table 2 lists the default λ values for a few n,

[15] Dirk Arnold (2000) suggested a remarkable simplification of Equation (17), replacing $(\|\boldsymbol{p}_\sigma^{(g+1)}\| - \widehat{\chi}_n)/\widehat{\chi}_n$ with $(\|\boldsymbol{p}_\sigma^{(g+1)}\|^2 - n)/(2n)$. This formulation fulfills demands analogously to those from Hansen (1998, 18f) on (17) and avoids the unattractive approximation of $\widehat{\chi}_n$. In preliminary investigations, both variants seem to perform equally well. If this result holds, we would prefer the latter variant.

[16] On principle, the definition of parameter μ is superfluous, because μ could be implicitely defined by setting $w_1,\ldots,w_\mu > 0$ and $w_\mu,\ldots,w_\lambda = 0$.

Table 1: Default parameter setting for the (μ_W, λ)-CMA-ES.

λ	μ	$w_{i=1,\ldots,\mu}$	c_c	c_{cov}	c_σ	d_σ
$4 + \lfloor 3\ln(n) \rfloor$	$\lfloor \lambda/2 \rfloor$	$\ln\left(\frac{\lambda+1}{2}\right) - \ln(i)$	$\frac{4}{n+4}$	$\frac{2}{(n+\sqrt{2})^2}$	$\frac{4}{n+4}$	$c_\sigma^{-1} + 1$

Table 2: Default λ values, small numbers indicate the truncated portion.

n	2	3	4	6	8	11	15	21	40	208
$4 + \lfloor 3\ln(n) \rfloor$	6.08	7.30	8.16	9.38	10.24	11.19	12.12	13.13	15.07	20.01

where $\lambda(n-1) < \lambda(n)$. In the (μ_I, λ)-CMA-ES, where $w_1 = \ldots = w_\mu$,[17] we choose $\mu = \lfloor \lambda/4 \rfloor$ as the default. To make the strategy more robust or more explorative in case of multimodality, λ can be enlarged, choosing μ accordingly. In particular, for unimodal and non-noisy problems, $\lambda = 9$ is often most appropriate even for large n.

Partly for historical reasons, in this paper, we use the (μ_I, λ)-CMA-ES. Based on our experience, for a given λ, an optimal choice of μ and $w_1, \ldots, w_\mu$ only achieves speed-up factors less than two compared to the (μ_I, λ)-CMA-ES, where $\mu \approx 0.27\lambda$. The optimal recombination weights depend on the function to be optimized, and it remains an open question whether the (μ_I, λ) or the (μ_W, λ) scheme performs better overall (using the default parameters accordingly).

If overall simulation time does not substantially exceed, say, $2n$ generations, d_σ should be chosen smaller than in Table 1, e.g., $d_\sigma = 0.5\, c_\sigma^{-1} + 1$. Increasing c_{cov}^{-1}, d_σ or μ *and* λ by a factor $\alpha > 1$ makes the strategy more robust. If this increases the number of needed function evaluations, as generally to be expected, it will be typically by a factor less than α. The explorative behavior can be improved by increasing μ *and* λ, or μ up to $\lambda/2$ in the (μ_I, λ)-CMA-ES, or d_σ in accordance with a suffiently large initial step size.

The parameter settings are discussed in more detail in the following.

λ: (Population size) In general, the equations

$$\lambda \geq 5 \qquad \text{and} \qquad 2\mu \lessapprox \lambda \lessapprox 2n + 10$$

give a reasonable choice for λ. Large values linearly worsen progress rates, e.g., on simple problems like the sphere function. Also on more sophisticated problems, performance can linearly decrease with increasing λ, because the adaptation time (in generations) is more or less independent of λ. Values considerably smaller than ten may decline the robustness of the strategy.

μ: (Parent number) We recommend choosing $2 \leq \mu \lessapprox n$. In the (μ_I, λ)-CMA-ES, in most cases, $\mu \approx 0.27\lambda$ will suffice (Beyer, 1996a; Herdy, 1993). To provide a robust strategy, large μ and if need be a larger ratio of μ/λ up to 0.5 are preferable (Hansen and Ostermeier, 1997). In particular for $n \lessapprox 5$, even $\mu = 1$ can occasionally be the best choice.

[17] The algorithm is independent from multiplication of $\boldsymbol{w} = (w_1, \ldots, w_\mu)$ with a real number greater than zero.

c_σ: (Cumulation for step size) Investigations based on Hansen (1998) give strong evidence that $c_\sigma < 1$ can and must be chosen with respect to the problem dimension

$$\sqrt{\frac{n}{2}} \lessapprox c_\sigma^{-1} \lessapprox n \ ,$$

while for large n, most sensible values are between $n/10$ and $n/2$. Large values, i.e., $c_\sigma^{-1} \approx n$, slow down the possible change rate of the step size because $d_\sigma = c_\sigma^{-1} + 1$ but can still be a good choice for certain problem instances.

d_σ: (Damping for step size) According to principle considerations, the damping parameter must be chosen $d_\sigma \approx \alpha\, c_\sigma^{-1}$, where α is near one, and $d_\sigma \geq 1$. Therefore we define $d_\sigma = \alpha\, c_\sigma^{-1} + 1$. Choosing α smaller than one, e.g., 0.3, can yield (slightly) larger progress rates on the sphere problem. Depending on n, c_σ, μ, and λ, a factor up to three can be gained. This is recommended if overall simulation time is considerably shorter than $3n\lambda$ function evaluations. Consequently one may choose $\alpha = 1 - \min(0.7, n/g_{\max})$. If α is chosen too small, oscillating behavior of the step-size can occur, and strategy performance may decline drastically. Larger values for α linearly slow down the possible change rate of the step size and (consequently) make the strategy more robust.

c_c: (Cumulation for distribution) Based on empirical observations, we suspect $c_\sigma \leq c_c \leq 1$ to be a sensible choice for c_c. Especially when long axes have to be learned, $c_c \approx c_\sigma$ should be most efficient without compromising the learning of short axes.

c_{cov}: (Change rate of the covariance matrix) With decreasing change rate c_{cov}, reliability of the adaptation increases (e.g., with respect to noise) as well as adaptation time. For $c_{\text{cov}} \to 0$, the CMA-ES becomes an ES with only global step size adaptation. For strongly disturbed problems, it can be appropriate to choose smaller change rates like $c_{\text{cov}} = \alpha/(n + \sqrt{\alpha})^2$ with $\alpha \lessapprox 1$ instead of $\alpha = 2$.

5.2 Limitations and Practical Hints

Besides the limitations of any general linear encoding/decoding scheme, the limitations for the CMA, revealed so far, result from shortage of valid selection information (Hansen and Ostermeier, 1997) or numeric precision problems. The former can be due to selection irrelevant parameters or axes, weak or distorted selection, or also, numeric precision problems. In any of these cases, a random walk on strategy parameters will appear. In the CMA, the condition of the covariance matrix $\boldsymbol{C}$ increases unbounded bringing search, with respect to the short principal axes of $\boldsymbol{C}$, to a premature end.

The initial step size $\sigma^{(0)}$ should be chosen such that σ does not tend to increase significantly within the initial $2/c_\sigma$ generations. Otherwise the initially learned distribution shape can be inefficient and may have to be unlearned consuming a considerable number of additional function evaluations. This effect of a too small $\sigma^{(0)}$ can be avoided keeping the mutation distribution shape (i.e., the covariance matrix) initially constant in case of an initially increasing σ. Furthermore, the prominent effect of the initial step size on the global search performance (see Sections 7.4 and 6) should be kept in mind.

In the practical application, a minimal variance for the mutation steps should be ensured. Remember that d_{ii} is the i^{th} diagonal element of the step size matrix $\boldsymbol{D}$. To ensure a minimal variance, the standard deviation of the shortest principal axis of the mutation ellipsoid, $\sigma \cdot \min_i(d_{ii})$, must be restricted. If it falls below the given bound,

step size σ should be enlarged (compare Appendix A). Problem specific knowledge must be used for setting the bound. Numerically, a lower limit is given by the demand $\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g)} \neq \langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g)} + 0.2 \cdot \sigma \, \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{e}_i$ for each unit vector $\boldsymbol{e}_i$. With respect to the selection procedure, even $f(\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g)}) \neq f(\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(g)} + 0.2 \cdot \sigma \, \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{e}_i)$ seems desirable.

To avoid numerical errors, a maximal condition number for $\boldsymbol{C}^{(g)}$, e.g., 10^{14}, should be ensured. If the ratio between the largest and smallest eigenvalue of $\boldsymbol{C}^{(g)}$ exceeds 10^{14}, the operation $\boldsymbol{C}^{(g)} := \boldsymbol{C}^{(g)} + (\max_i (d_{ii}^{(g)})^2 / 10^{14} - \min_i (d_{ii}^{(g)})^2) \, \boldsymbol{I}$ limits the condition number in a reasonable way to $10^{14} + 1$.

The numerical stability of computing eigenvectors and eigenvalues of the symmetrical matrix $\boldsymbol{C}$ usually poses no problems. In MATLAB, the built-in function `eig` can be used. To avoid complex solutions, the symmetry of $\boldsymbol{C}$ must be explicitly enforced (compare Appendix A). In C/C++, we used the functions `tred2` and `tqli` from Press et al. (1992), substituting `float` with `double` and setting the maximal iteration number in `tqli` to $30n$. Another implementation was done in Turbo-PASCAL. For condition numbers up to 10^{14}, we never observed numerical problems in any of these programming languages.

If the user decides to manually change object parameters during the search process – which is not recommended – the adaptation procedure (Equations (14) to (17)) must be omitted for these steps. Otherwise, the strategy parameter adaptation can be severely disturbed.

Because change of $\boldsymbol{C}^{(g)}$ is comparatively slow (time scale n^2), it is possible to update $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$ not after every generation but after $\sqrt{n}$ or even after, e.g., $n/10$ generations. This reduces the computational effort of the strategy from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^{2.5})$ or $\mathcal{O}(n^2)$, respectively. The latter corresponds to the computational effort for a fixed linear encoding/decoding of the problem as well as for producing a realization of an (arbitrarily) normally distributed, random vector on the computer. In practical applications, it is often most appropriate to update $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$ every generation.

A simple method to handle constraints repeats the generation step, defined in Equation (13), until λ or at least μ feasible points are generated before Equations (14) to (17) are applied. If the initial mutation distribution generates sufficiently numerous feasible solutions, this method can remain sufficient during the search process due to the symmetry of the mutation distribution. If the minimum searched for is located at the edge of the feasible region, the performance will usually be poor and a more sophisticated method should be used.

6 Test Functions

According to Whitley et al. (1996), we prefer test functions that are non-linear, non-separable, scalable with dimension n, and resistant to simple hill-climbing. In addition, we find it reasonable to use test functions with a comprehensible topology even if $n \geq 3$. This makes the interpretation of observed results possible and can lead to remarkable scientific conclusions. Table 3 gives the test functions used in this paper. The test suite mainly strives to test (local) convergence speed. Only functions 10–12 are multimodal.[18] Apart from functions 6–8, the global optimum is located at $\mathbf{0}$.

To yield non-separability, we set $\boldsymbol{y} := [\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n]^{\mathrm{T}} \boldsymbol{x}$, where $\boldsymbol{x}$ is the object parameter vector according to Equation (13). $[\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n]^{\mathrm{T}}$ implements an angle-preserving linear transformation of $\boldsymbol{x}$, i.e., rotation and reflection of the search space. $\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n \in \mathbb{R}^n$

[18] For higher dimension, even function 8 f_{Rosen} has a local minimum near $\boldsymbol{y} = (-1, 1, \ldots, 1)^{\mathrm{T}}$. With the given initialization of $\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(0)}$ and $\sigma^{(0)}$, the ES usually converges into the global minimum at $\boldsymbol{y} = \mathbf{1}$.

Table 3: Test functions (to be minimized), where $\boldsymbol{y} = [\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n]^{\mathrm{T}} \boldsymbol{x}$.

Function	$\sigma^{(0)}$	$\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(0)}$	f^{stop}
1. $f_{\mathrm{sphere}}(\boldsymbol{x}) = \sum_{i=1}^{n} {x_i}^2$	1	$\sum_{i=1}^{n} \boldsymbol{o}_i$	10^{-10}
2. $f_{\mathrm{Schwefel}}(\boldsymbol{y}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} y_i \right)^2$	1	$\sum_{i=1}^{n} \boldsymbol{o}_i$	10^{-10}
3. $f_{\mathrm{cigar}}(\boldsymbol{y}) = {y_1}^2 + \sum_{i=2}^{n} (1000\, y_i)^2$	1	$\sum_{i=1}^{n} \boldsymbol{o}_i$	10^{-10}
4. $f_{\mathrm{tablet}}(\boldsymbol{y}) = (1000\, y_1)^2 + \sum_{i=2}^{n} {y_i}^2$	1	$\sum_{i=1}^{n} \boldsymbol{o}_i$	10^{-10}
5. $f_{\mathrm{elli}}(\boldsymbol{y}) = \sum_{i=1}^{n} \left(1000^{\frac{i-1}{n-1}} y_i \right)^2$	1	$\sum_{i=1}^{n} \boldsymbol{o}_i$	10^{-10}
6. $f_{\mathrm{parabR}}(\boldsymbol{y}) = -y_1 + 100 \sum_{i=2}^{n} {y_i}^2$	1	$\boldsymbol{0}$	-1000
7. $f_{\mathrm{sharpR}}(\boldsymbol{y}) = -y_1 + 100 \sqrt{\sum_{i=2}^{n} {y_i}^2}$	1	$\boldsymbol{0}$	-1000
8. $f_{\mathrm{Rosen}}(\boldsymbol{y}) = \sum_{i=1}^{n-1} \left(100 \left({y_i}^2 - y_{i+1} \right)^2 + (y_i - 1)^2 \right)$	0.1	$\boldsymbol{0}$	10^{-10}
9. $f_{\mathrm{diffpow}}(\boldsymbol{y}) = \sum_{i=1}^{n} \lvert y_i \rvert^{2+10\frac{i-1}{n-1}}$	0.1	$\sum_{i=1}^{n} \boldsymbol{o}_i$	10^{-15}
10. $f_{\mathrm{Rast}}(\boldsymbol{y}) = 10n + \sum_{i=1}^{n} \left((a_i y_i)^2 - 10 \cos(2\pi \cdot a_i y_i) \right)$, $a_i = 1$	100	$\sum_{i=1}^{n} u_i \boldsymbol{o}_i$ $u_i \in [-5.12; 5.12]$	
11. $f_{\mathrm{Rast10}}(\boldsymbol{y}) = f_{\mathrm{Rast}}(\boldsymbol{y})$, where $a_i = 10^{\frac{i-1}{n-1}}$	100	$\sum_{i=1}^{n} u_i \boldsymbol{o}_i$ $u_i \in [-5.12; 5.12]$	
12. $f_{\mathrm{Rast1000}}(\boldsymbol{y}) = f_{\mathrm{Rast}}(\boldsymbol{y})$, where $a_i = 1000^{\frac{i-1}{n-1}}$	100	$\sum_{i=1}^{n} u_i \boldsymbol{o}_i$ $u_i \in [-5.12; 5.12]$	

is a randomly oriented, orthonormal basis, fixed for each simulation run. Each $\boldsymbol{o}_i$ is realized equally distributed on the unit (hyper-)sphere surface, dependently drawn so that $\langle \boldsymbol{o}_i, \boldsymbol{o}_j \rangle = 0$ if $i \neq j$. An algorithm to generate the basis is given in Figure 6. For the initial $\langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(0)} = \sum_{i=1}^{n} \boldsymbol{o}_i$, we yield $\boldsymbol{y}^{(0)} = [\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n]^{\mathrm{T}} \langle \boldsymbol{x} \rangle_{\mathrm{w}}^{(0)} = (1, \ldots, 1)^{\mathrm{T}}$. For the canonical basis, where $\boldsymbol{o}_i = \boldsymbol{e}_i$, it holds $\boldsymbol{y} = \boldsymbol{x}$ and, obviously, $\sum_i \boldsymbol{o}_i = (1, \ldots, 1)^{\mathrm{T}}$. For a non-canonical basis, only f_{sphere} remains separable.

Functions 1–5 are convex-quadratic and can be linearly transformed into the sphere problem.[19] They serve to test the adaptation demand stated in Section 3. The axis scaling between longest and shortest principal axis of functions 3–5 is 1000, i.e., the problem condition is 10^6. Taking into account real-world problems, a search strategy should be able to handle axis ratios up to this number. f_{tablet} can be interpreted as a sphere model with a smooth equality constraint in $\boldsymbol{o}_1$ direction.

f_{parabR} and f_{sharpR} facilitate straight ridge topologies. The ridge points into $\boldsymbol{o}_1$ direction that has to be enlarged unlimited. The sharp ridge function f_{sharpR} is topologically invariant from the distance to the ridge peak. In particular, the local gradient is constant, independent of the distance, which is a hard feature for a local search strategy. Results can be influenced by numerical precision problems (even dependent on the implementation of the algorithm). Therefore, we establish for this problem a minimal step size of 10^{-10} and a maximal condition number for $\boldsymbol{C}^{(g)}$ of 10^{14} (compare Section

[19] They can be written in the form $f(\boldsymbol{x}) = f_{\mathrm{sphere}}(\boldsymbol{A}\boldsymbol{x}) = (\boldsymbol{A}\boldsymbol{x})^{\mathrm{T}} \boldsymbol{A}\boldsymbol{x} = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{A}\boldsymbol{x} = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{H}\boldsymbol{x}$, where $\boldsymbol{A}$ is a full rank $n \times n$-matrix, and the Hessian matrix $\boldsymbol{H} = \boldsymbol{A}^{\mathrm{T}} \boldsymbol{A}$ is symmetric and positive definite.

FOR $i = 1$ TO n

1. Draw components of $\boldsymbol{o}_i$ independently $(0,1)$-normally distributed
2. $\boldsymbol{o}_i := \boldsymbol{o}_i - \sum_{j=1}^{i-1} \langle \boldsymbol{o}_i, \boldsymbol{o}_j \rangle \boldsymbol{o}_j$ ($\langle .,. \rangle$ denotes the canonical scalar product)
3. $\boldsymbol{o}_i := \boldsymbol{o}_i / ||\boldsymbol{o}_i||$

ROF

Figure 6: Algorithm to generate a random orthonormal basis $\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n \in \mathbb{R}^n$ (Hansen et al., 1995).

5.2 and Appendix A).

The generalized Rosenbrock function f_{Rosen}, sometimes called the "banana function," facilitates a bent ridge, where the global optimum is at $\boldsymbol{y} = \mathbf{1}$, which means $\boldsymbol{x} = \sum_{i=1}^{n} \boldsymbol{o}_i$. Like f_{Rosen}, the sum of different powers f_{diffpow} cannot be linearly transformed into the sphere problem. Here the mis-scaling continually increases while approaching the optimum. A local search strategy presumably follows a gradually narrowing ridge.

Functions 10–12 are multimodal. f_{Rast} is the generalized form of the well-known Rastrigin function, while f_{Rast10} and f_{Rast1000} are mis-scaled versions of f_{Rast}. The mis-scaling between longest and shortest axis is 10 for f_{Rast10} and 1000 for f_{Rast1000}. For $n = 20$, the factor between "adjacent" axes is 1.13 and 1.44, respectively. We feel the moderately mis-scaled Rastrigin function f_{Rast10} is a much more realistic scenario than the perfectly scaled one. On the multimodal test functions, the initial step size is chosen comparatively large. With small initial step sizes, the local optimum found by the ES is almost completely determined by the initial point that is, in each coordinate, equally distributed in $[-5.12; 5.12]$.

Invariance (compare also Section 3) is an important feature of a search strategy, because it facilitates the possibility of generalizing simulation results. Imagine, for example, translation invariance is not given. Then it is not sufficient to test $f : \boldsymbol{x} \mapsto f(\boldsymbol{x} - \boldsymbol{a})$ for, e.g., $\boldsymbol{a} = \mathbf{0}$. One has to test various different values for $\boldsymbol{a}$ and may end up with different results open to obscure interpretations. Typically, ESs are translation invariant. Simple ESs are, in addition, invariant against rotation and reflection of the search space, i.e., they are independent of the choice of the orthonormal coordinate system $\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n$. More complex ES algorithms may be lacking this invariance (compare Sections 3.1 and 7.2).[20] Any evaluation of search strategies (for example, by test functions) has to take into account this important point.

7 Simulation Results

Four different evolution strategies are experimentally investigated.

- $(\mu_{\text{I}}, \lambda)$-CMA-ES, where the default parameter setting from Table 1 is used apart from λ and μ as given below, and $w_i = 1$ for all $i = 1, \ldots, \mu$. To reduce the computational effort, the update of $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$ from $\boldsymbol{C}^{(g)}$ for Equations (13), (14), and (16) is done every $\sqrt{n}$ generations in all simulations.

[20] For example, invariance against rotation is lost when discrete recombination is applied.

- $(15/2_{\mathrm{I}}, 100)$-CORR-ES (correlated mutations), an ES with rotation angle adaptation according to Section 3.1. The initialization of the rotation angles is randomly uniform between $-\pi$ and π, identical for all initial individuals.
- $(2_{\mathrm{I}}, 10)$-PATH-ES, the same strategy as the $(2_{\mathrm{I}}, 10)$-CMA-ES, while c_{cov} is set to zero. $\boldsymbol{C}^{(g)} = \boldsymbol{I}$ for all $g = 0, 1, 2, \ldots$, and therefore, $\boldsymbol{B}^{(g)}$ and $\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}$ can be set to $\boldsymbol{I}$. Equations (14) and (15) become superfluous, and only cumulative path length control takes place.
- $(2_{\mathrm{I}}, 10)$-MUT-ES, an ES with mutative adaptation of one (global) step size and intermediate recombination on object and strategy parameters as in CORR-ES. Mutation on the strategy parameter level is carried out as in Equation (1), where ξ_k is $(0, 1/\sqrt{2n}^2)$-normally distributed.

In the beginning of Section 3, we formulated four fundamental demands on an algorithm that adapt a general linear encoding in ESs. Now we pursue the question whether the CMA-ES can satisfy these demands.

The invariance demand is met because the algorithm is formulated inherently independent of the coordinate system: All results of CMA-ES, PATH-ES, and MUT-ES are independent of the basis $\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n$ actually chosen (see Section 6), i.e., valid for any orthonormal coordinate system. That means, these strategies are, in particular, independent of rotation and reflection of the search space.

The performance demand is also satisfied. The $(2_{\mathrm{I}}, 9)$-CMA-ES performs on f_{sphere} between 1.5 (n large) and 3 ($n = 5$) times slower than the $(1, 5)$-ES with isotropic mutation distribution and optimal step size. The applicability of the CMA algorithm is independent of any reasonable choice of μ and λ.

The stationarity demand is respected because, under random selection, expectation of $\boldsymbol{C}^{(g+1)}$ equals $\boldsymbol{C}^{(g)}$, and expectation of $\log \sigma^{(g+1)}$ equals $\log \sigma^{(g)}$ (Hansen, 1998).

Simulation runs on functions 3–5 in Section 7.2 will show that the adaptation demand is met as well.

In Section 7.3, performance results on the unimodal test functions 1–9 are presented, and scaling properties are discussed. In Section 7.4, global search properties are evaluated.

7.1 CPU-Times

We think strategy internal time consumption is of minor relevance. It seems needless to say that results strongly depend on implementational skills and the programming language chosen. Nevertheless, to complete the picture, we summarize CPU-times taken by strategy internal operations of different $(2_{\mathrm{I}}, 10)$-ESs in Figure 7. The strategies were coded in MATLAB and run uncompiled on a pentium 400 MHz processor. Differences between MUT-ES and PATH-ES are presumably due to the more sophisticated recombination operator in MUT-ES, which allows two parent recombination independent of the parent number μ. MATLAB offers relatively efficient routines for the most time consuming computations in the CMA-ES. Therefore, even for $n = 320$, the CMA-ES needs only 50ms CPU-time per generated offspring. This time can still be reduced doing the update of $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$ every $n/10$ generations instead of every $\sqrt{n}$ generations. No efficient MATLAB routines are available for the rotation procedure in the CORR-ES. To make CPU-times comparable, this procedure was reimplemented in C and called from MATLAB gaining a remarkable speed up. CORR-ES is still seven to ten times slower than CMA-ES. This may be due to a trivial implementational matter and does not make any difference for many non-linear, non-separable search problems.

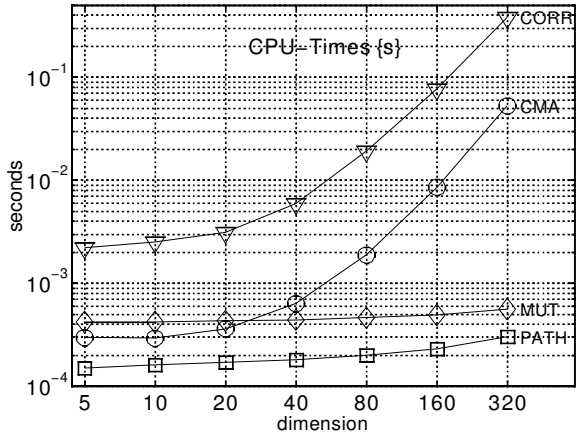

Figure 7: Strategy internal CPU-times per function evaluation (i.e., per generated offspring) of different $(2_{\mathrm{I}}, 10)$-ESs in seconds. The strategies were coded in MATLAB (compare text) and run on a pentium 400 MHz processor.

7.2 Testing Adaptation

In this section, single simulation runs on the convex-quadratic functions 3–5 are shown in comparison with runs on f_{sphere}, revealing whether the adaptation demand can be met.

We start with a discussion of the $(15/2_{\mathrm{I}}, 100)$-CORR-ES. In Figure 8, with respect to the stop criterion, the best, middle, and worst out of 17 runs on f_{cigar}, f_{tablet}, and f_{elli} are shown, where $n = 5$.[21] Respective results for the $(15_{\mathrm{I}}, 100)$-CMA-ES are shown for comparison. This selection scheme is not recommended for the CMA-ES, where $n = 5$ (see Section 5.1, in particular Table 1 and 2). The recommended $(4_{\mathrm{W}}, 8)$-CMA-ES performs roughly ten times faster (compare also Figure 10).

First, we compare the results between the left and right column in the figure. On the left, the axis-parallel, and therefore completely separable, versions of the functions are used ($\boldsymbol{o}_i = \boldsymbol{e}_i$ and $\boldsymbol{y} = \boldsymbol{x}$, compare Section 6). Only axis-parallel mutation ellipsoids are necessary to meet the adaptation demand. On the right, the basis $\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n$ is randomly chosen anew for each run. The CORR-ES performs roughly ten to forty times slower on the non-axis-parallel oriented versions. In accordance with previous results (Hansen et al., 1995; Holzheuer, 1996; Hansen, 2000), the CORR-ES strongly depends on the orientation of the given coordinate system and largely exploits the separability of the problem. In contrast, the CMA-ES performs identically on the left and right.

Only on the axis-parallel versions of f_{elli} and f_{tablet} does the CORR-ES partly meet the adaptation demand. At times, after an adaptation phase, progress rates are similar to those on f_{sphere}. On the non-axis-parallel functions, the progress rates are worse by a factor between 100 (best run on f_{elli}) and 6000 (worst run on f_{cigar}) compared to those on f_{sphere}. As Rudolph (1992) pointed out, the search problem on the strategy parameters is multimodal. Even after a supposed adaptation phase, on most test functions, long phases with different progress rates can be observed. This suggests the hypothesis that the ratio between the effect of the mutation (of the angles) and the width of the

[21] With 17 runs, one would expect roughly 95% of any simulations to end up between the shown best and worst run.

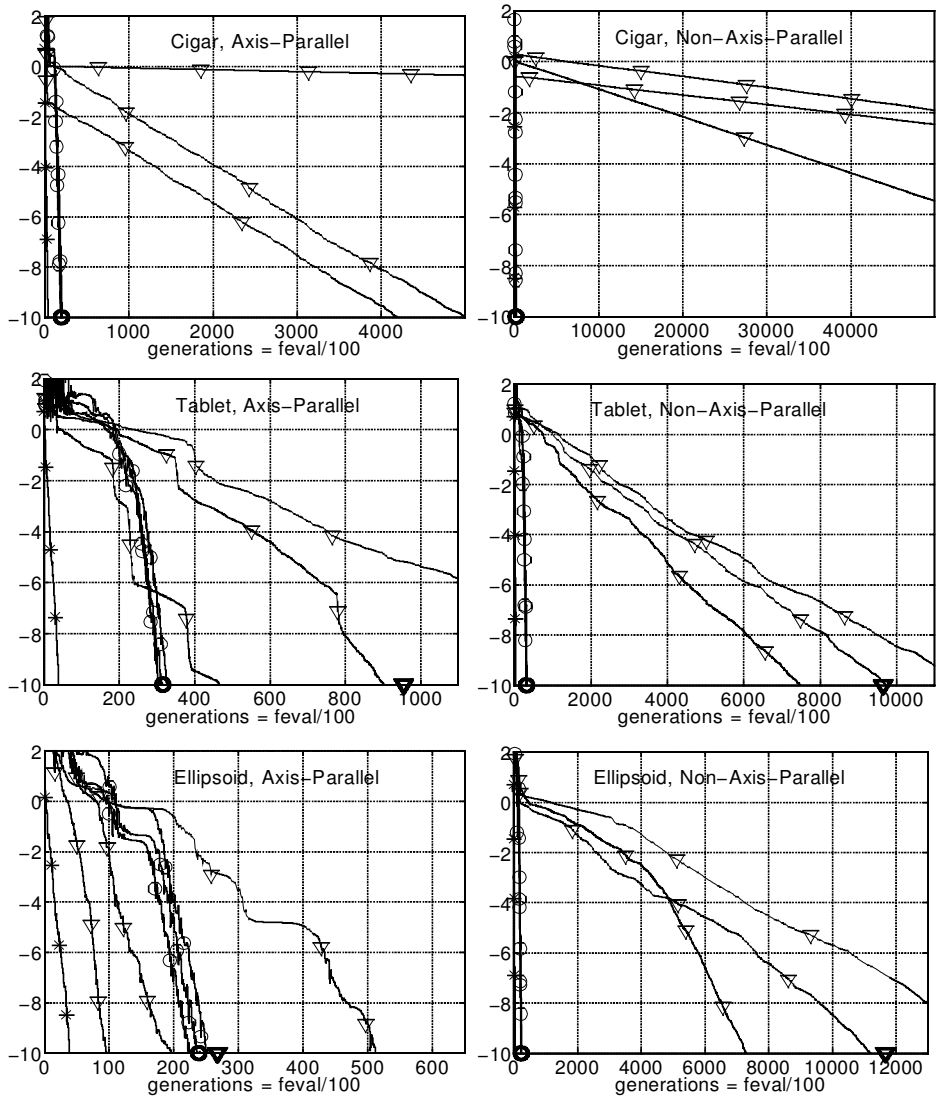

Figure 8: $\log_{10}$(function value) vs. generation number with $(15/2_{\mathrm{I}}, 100)$-CORR-ES ($\bigtriangledown$), where $n = 5$. The best, middle, and worst out of 17 runs are shown. Bold symbols at the lower edge indicate the mean generation number to reach $f^{\mathrm{stop}} = 10^{-10}$. In the left column, $\boldsymbol{o}_i = \boldsymbol{e}_i$, that is, the functions are axis-parallel oriented and completely separable. The range of the abscissa between left and right columns is enlarged by a factor 20 for f_{elli} (lower row) and 10 otherwise. For comparison are shown a single run on f_{sphere} ($*$) and respective simulations with the $(15_{\mathrm{I}}, 100)$-CMA-ES ($\bigcirc$) that do *not* reflect the real performance of the CMA-ES. The default $(4_{\mathrm{W}}, 8)$-CMA-ES performs roughly ten times faster. The CORR-ES does not meet the adaptation demand. Progress rates on the non-separable functions are worse by a factor between 100 and 6000 compared to those on f_{sphere}.

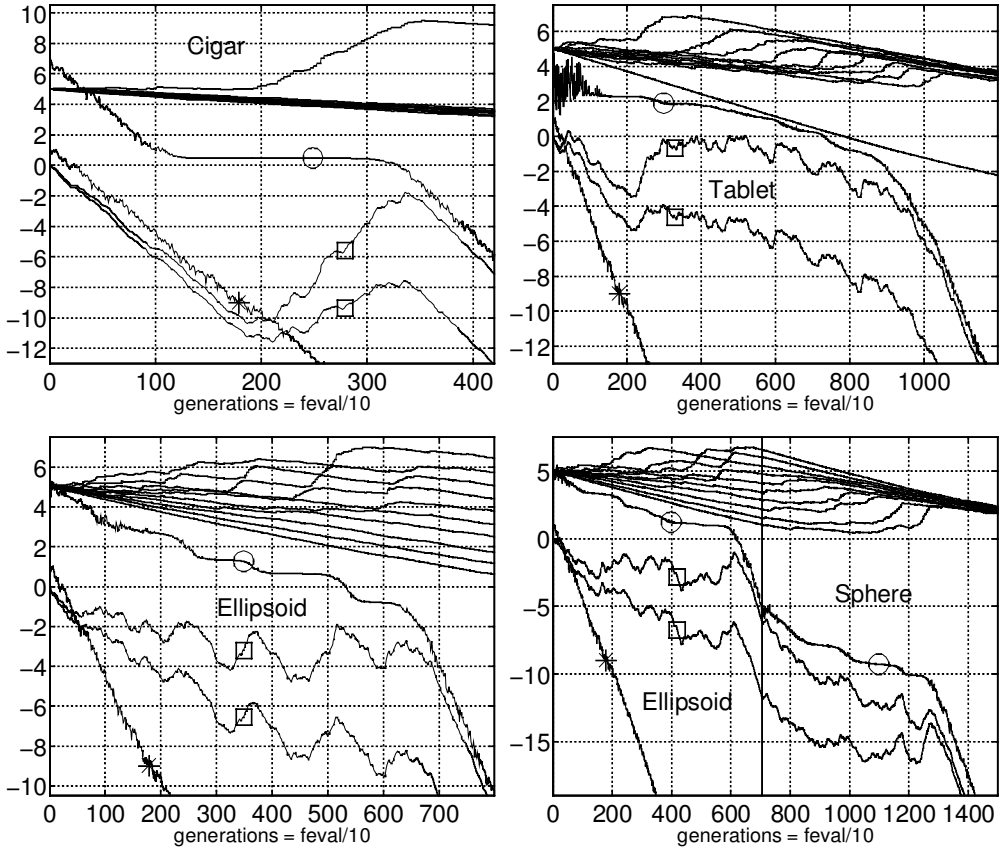

Figure 9: Simulation runs with the $(2_I, 10)$-CMA-ES, where $n = 10$. ○: $\log_{10}$(function value). □: $\log_{10}$(smallest and largest variance of the mutation distribution), ∗: $\log_{10}$(function value) from a run on f_{sphere}. The upper ten graphs (without symbol) are the variances of the principal axes of the mutation distribution ellipsoid, sorted, and multiplied with $10^5/{\sigma^{(g)}}^2$ on a logarithmic scale. Lower right: Simulation run on f_{elli} until function value 10^{-5} is reached and afterwards on const $\cdot\, f_{\text{sphere}}$. The CMA-ES meets the adaptation demand. After an adaptation phase, in all cases, the progress rates are identical with those on f_{sphere}.

respective optimum differs significantly between different optima. Therefore, different progress rates can be observed.

Finally, we stress reliability and replicability of the results with the CORR-ES. First, the initialization of the angles is of major importance for the results. Initializing the angles with zero, which is a global optimum for the axis-parallel functions, is almost ten times slower on the axis-parallel f_{elli}. Also, initializing all individuals with different angles is still considerably worse than the results shown. Second, the implemented order of applied rotations influences the results especially on f_{cigar} and f_{tablet}. Using, e.g., the reverse rotation order, results are significantly worse.

We continue with a discussion of the $(2_I, 10)$-CMA-ES. Due to a small variance between different simulation runs (compare Figures 8 and 10), it is sufficient here to look at single runs on different functions. Figure 9 shows simulation runs on f_{cigar}

(upper left) and f_{tablet} (upper right), where $n = 10$, compared to a run on f_{sphere}. The adaptation process can be clearly observed through the variances of the principle axes of the mutation distribution, where global step size is disregarded (upper ten graphs). When the variances correspond to the scaling in the objective function, progress rates identical to those on f_{sphere} are realized (notice again that the basis $\boldsymbol{o}_1, \ldots, \boldsymbol{o}_n$ is chosen randomly here). The shorter adaptation time on f_{cigar} is due to the cumulation, which detects the parallel correlations of single steps very efficiently.

In the lower left of Figure 9, a simulation run on f_{elli} is shown. Similar to f_{tablet}, but more pronounced, local adaptation occurs repeatedly on this function. Progress increases and decreases a few times together with the step size. When the adaptation is completed, the variances are evenly spread in a range of 10^6. In the lower right of Figure 9, the objective function is switched from f_{elli} to (some multiple of) f_{sphere}, after reaching function value 10^{-5}. The mutation distribution adapts from an ellipsoid-like shape back to the isotropic distribution. Adaptation time and graphs of function value and step size are very similar to those on f_{elli}. In fact, from a theoretical point of view, the algorithm must show exactly the same behavior in both adaptation cases (apart from stochastic influences). Again, after the adaptation phase, progress rates identical to those on f_{sphere} are achieved in both cases.

Concluding these results, the adaptation demand is satisfied by the CMA-ES: Any convex-quadratic function is rescaled into the sphere function.

7.3 Testing Convergence Speed and Scaling

In this section, we investigate the number of function evaluations to reach f^{stop} on test functions 1–9, where $n = 5; 20; 80; 320$. The $(2_{\text{I}}, 10)$-CMA-ES is compared to the $(2_{\text{I}}, 10)$-PATH-ES, where only global step size adaptation takes place and which usually (slightly) outperforms the $(2_{\text{I}}, 10)$-MUT-ES. Depending on CPU-time resources, up to 50 simulation runs are evaluated. Figure 10 shows the results on the convex-quadratic test functions 1–5 (first row). On f_{sphere}, both strategies perform almost identically, while on f_{Schwefel}, the results are still comparable (upper left). The difference on f_{sphere} is due to the decreasing variance of the covariance matrix supporting the adaptation of the step size σ that is somewhat too slow (see above discussion of damping parameter d_σ in Section 5.1). If the axis ratio between the longest and shortest axis is $1 : 1000$, as on f_{cigar}, f_{tablet}, and f_{elli} (upper right), CMA-ES outperforms PATH-ES by a factor between 7 (f_{tablet}, $n = 320$) and almost 60000 (f_{cigar}). Only on f_{tablet} for $n > 30$ does the factor fall below 100.

The lower row of Figure 10 shows the results on functions 6–9. The CMA-ES works quite well even on different kinds of ridge-like topologies. On f_{Rosen}, the CMA-ES is 7–80 times faster than PATH-ES; on f_{parabR}, the factor becomes 2000 (lower left). Results for the PATH-ES on f_{sharpR} and f_{diffpow} are omitted (lower right). On the latter, f^{stop} is reached after more than 10^{10} function evaluations ($n = 5$). On the former, the PATH-ES does not reach f^{stop} because step size converges to zero (as with mutative step size control). Setting the minimum step size to 10^{-10}, far more than 10^{10} function evaluations are needed to reach f^{stop}.

We take an interesting look at the scaling properties with n (compare the sloping grids in the figure). As with simple ESs in general, PATH-ES scales linearly on most functions. There are two exceptions. First, on f_{Schwefel} the PATH-ES scales quadratically. The axis ratio between longest and shortest principal axis of this function increases with increasing n. That is, not only problem dimension but also "problem difficulty" increases. Therefore, a simple ES scales worse than linear. This favors CMA-ES

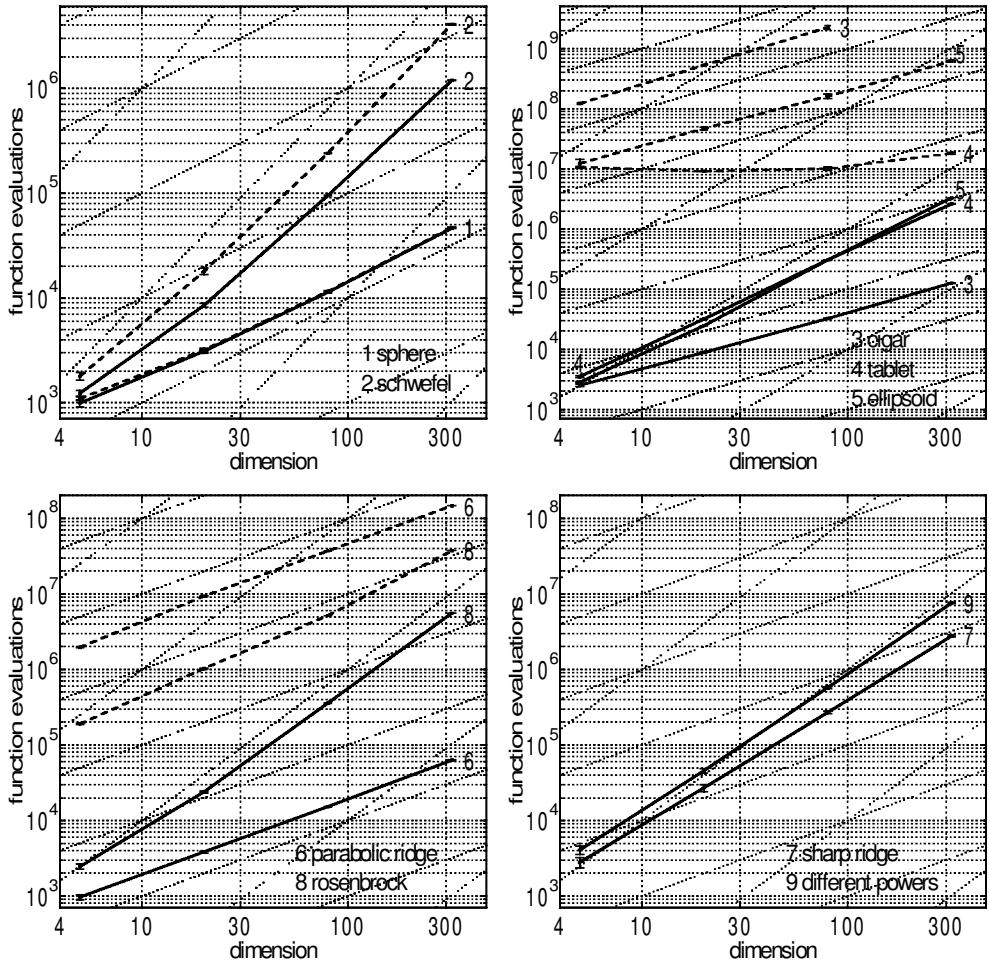

Figure 10: Function evaluations to reach f^{stop} with the $(2_{\text{I}}, 10)$-CMA-ES (solid graphs) and the $(2_{\text{I}}, 10)$-PATH-ES (dashed graphs) vs. dimension n on the test functions 1–9, where $n = 5; 20; 80; 320$. Shown are mean values and standard deviations. Sloping dotted lines indicate $\text{const} \cdot n = [10; 100; 1000; \ldots] \cdot n$ and $\text{const} \cdot n^2 = [1/100; 1; 100; \ldots] \cdot n^2$ function evaluations. For the PATH-ES the missing result on f_{cigar} ($n = 320$) could not be obtained in reasonable time (upper right) and results on f_{sharpR} and f_{diffpow} are far above the shown area (lower right, compare text). On f_{sphere} (upper left), both strategies perform almost identically.

scaling slightly better than PATH-ES on f_{Schwefel}. Second and more surprisingly, performance of the PATH-ES on f_{tablet} is almost independent of n. This effect is due to the cumulative path length control that adapts here better (i.e., larger) step sizes with higher dimension.

In general, we expected the CMA-ES to scale quadratically with n. The number of free strategy parameters to be adapted is $(n^2 + n)/2$. f_{Rosen} and f_{diffpow} meet our expectations quite well. On f_{sphere}, no adaptation is necessary, and CMA-ES scales linearly with n, as to be expected of any ES.

In contrast, the perfect linear scaling on f_{cigar} and f_{parabR}, even though desirable, comes as a surprise to us. On both functions one long axis has to be adapted. We found the cumulation to be responsible for this excellent scaling. Without cumulation, the scaling on f_{cigar} is similar to the scaling on f_{tablet}. As mentioned above, the cumulation especially supports the adaptation of long axes. On f_{tablet}, f_{sharpR}, f_{elli}, and f_{Schwefel}, increasingly ordered, the scaling is between $n^{1.6}$ and $n^{1.8}$. Where the CMA-ES scales worse than the PATH-ES, performances align with $n \to \infty$ because the progress surpasses the adaptation procedure with $n \to \infty$.

In summary, the CMA-ES substantially outperforms the PATH-ES in dimensions up to 320 on all tested functions—apart from f_{sphere}, where both strategies perform almost identically. The CMA-ES always scales between n and n^2: Exactly linearly for the adaptation of long axes and if no adaptation takes place, nearly quadratically if a continuously changing topology demands persistent adaptation.

7.4 Testing Global Search Performance

In evolutionary computation, the aspect of global search is often emphasized. In contrast, we interpret ESs—not taking into account the initial search phase—as local search strategies. The population occupies a comparatively small area, and the horizon beyond this area is limited by the actual step size: Steps larger than some multiple of the distribution variance do virtually not appear.[22] In our opinion, even the so-called premature convergence is often due to the lack of local convergence speed. When developing adaptation mechanisms, we mainly strove to address local convergence speed and did not consider global search performance.

Consequently, even though the general judgment of global search performance is problematic, it is sometimes argued that adaptation to the local topology of the objective function spoils global search properties of an algorithm. With respect to the CMA-ES, we discuss this objection now: There is good reason, and some evidence, that even the opposite is a more appropriate point of view: The local adaptation mechanism of the CMA-ES improves global search properties.

We compare different $(2_{\text{I}}, 10)$-ESs on the generalized Rastrigin function, where $n = 20$. Increasing population size improves the performance on this function. In contrast, the differences between the strategies compared are not affected. Note that a smaller initial step size worsens the performance as interpreted in Section 6. The CMA-ES is compared to the MUT-ES, where mutative control of only global step size takes place. PATH-ES performs similar to MUT-ES on the investigated Rastrigin functions.

Figure 11 shows 30 simulation runs on f_{Rast} with the CMA-ES (left) and the MUT-ES (right). Behavior of both strategies is very similar. They get trapped into local minima with function values between 30 and 100 within about 2000 and 3000 function evaluations.

Figure 12 shows 30 simulation runs on the scaled Rastrigin function f_{Rast10} that should be regarded as a more realistic multimodal test problem than f_{Rast}. Even though the mis-scaling between longest and shortest axis is only of a factor ten, the CMA-ES (left) and the MUT-ES (right) perform quite differently here. Function values obtained with MUT-ES are worse by a factor of 20 than those obtained with CMA-ES. In addition, these values are finally reached after about $5 \cdot 10^5$ function evaluations compared to

[22] In a pure mathematical sense, this is, of course, wrong. But, in contrast to any theoretical consideration, in practical applications, the finite time horizon is too short to wait for those events to occur. Even if a distribution is chosen that facilitates large steps more often, due to a search space volume phenomenon, these steps will virtually never produce better points if n exceeds, say, ten.

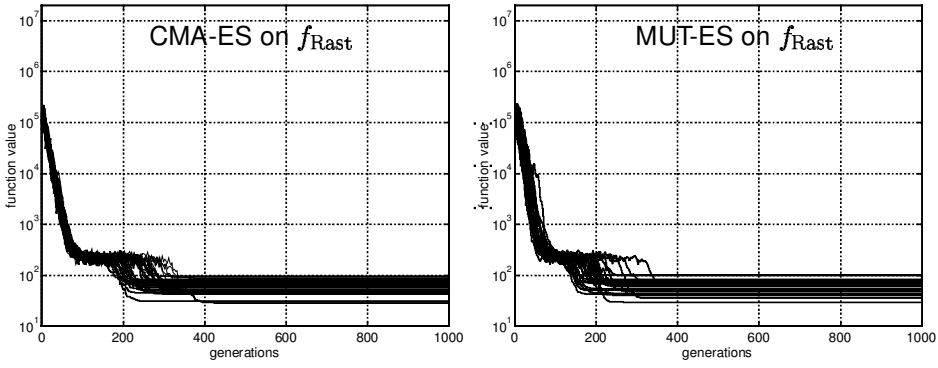

Figure 11: 30 simulation runs on the generalized Rastrigin function f_{Rast} with the $(2_{\mathrm{I}}, 10)$-CMA-ES (left) and the $(2_{\mathrm{I}}, 10)$-MUT-ES (right), where $n = 20$. Both strategies perform very similar.

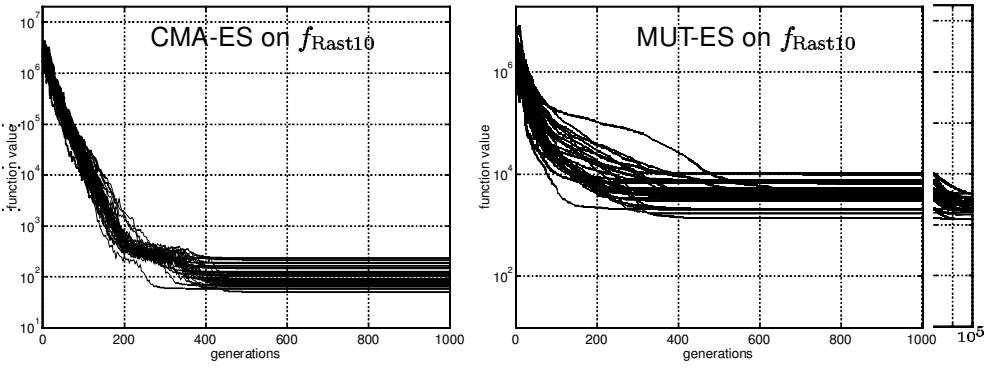

Figure 12: 30 simulation runs on the scaled Rastrigin function f_{Rast10}, maximal axis ratio $1 : 10$, with the $(2_{\mathrm{I}}, 10)$-CMA-ES (left) and the $(2_{\mathrm{I}}, 10)$-MUT-ES (right), where $n = 20$. Function values reached with MUT-ES are worse by a factor 20.

$5 \cdot 10^3$ function evaluations with the CMA-ES.

Figure 13 shows 30 simulation runs on f_{Rast1000}. The variance of function values reached with the CMA-ES (left) is larger than on f_{Rast} and f_{Rast10}. Nevertheless, about half of the simulation runs end up in local optima with function values less than 100. This takes about $3 \cdot 10^4$ function evaluations, i.e., about ten times longer than on f_{Rast} or f_{Rast10}. The adaptation time is now a decisive factor. With the MUT-ES, obtained function values are worse by a factor 10000 than those obtained with the CMA-ES, and simulation roughly needs ten times the number of function evaluations (strictly speaking, even after 10^6 function evaluations, final function values are not yet reached).

We found very similar results to those presented here on the Rastrigin functions in earlier investigation on a more complex, constrained multimodal test problem (EVOTECH-7, 1997).

Even though at first glance these results are surprising, there is a simple explanation why global search properties are improved. While the shape of the mutation distribution becomes suitably adapted to the topology of the objective function, the step size is adjusted much larger than without adaptation. A larger step size improves

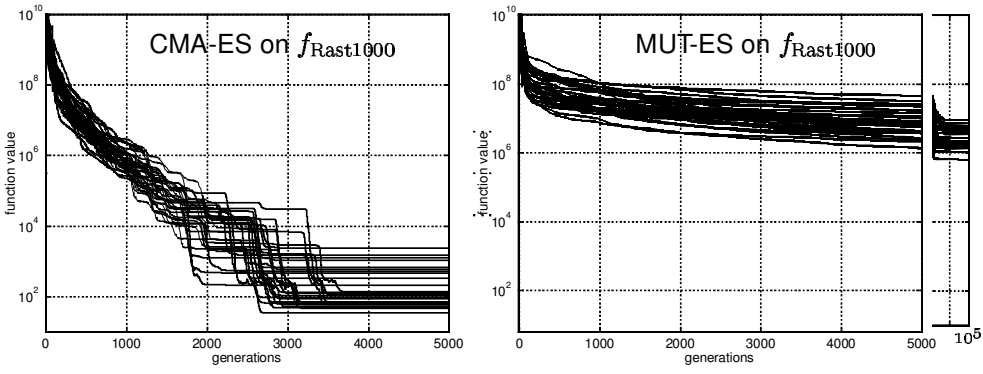

Figure 13: 30 simulation runs on the scaled Rastrigin function f_{Rast1000}, maximal axis ratio $1:1000$, with the $(2_{\text{I}}, 10)$-CMA-ES (left) and the $(2_{\text{I}}, 10)$-MUT-ES (right), where $n = 20$. Function values reached with the MUT-ES are worse by a factor 10000.

the global search performance of a local search procedure. The effect of the distribution adaptation on the step size can be clearly observed in Figure 9. Variances in *all* directions continually increase with the ongoing adaptation to the topology on f_{cigar} between generations 200 and 330 (shown are the smallest and the largest variance) and several times on f_{elli}.

Concluding this section, we observe the local adaptation in the CMA-ES to come along with an increasing step size that is more likely to improve than to spoil global search performance of a local search algorithm.

8 Conclusion

For the application to real-world search problems, the evolution strategy (ES) is a relevant search strategy if neither derivatives of the objective function are at hand, nor differentiability and numerical accuracy can be assumed. If the search problem is expected to be significantly non-separable, an ES with covariance matrix adaptation (CMA-ES), as put forward in this paper, is preferable to any ES with only global or individual step size adaptation.

We believe that there are principle limitations to the possibilities of self-adaptation in evolution strategies: To reliably adapt a significant change of the mutation distribution shape, at least roughly $10n$ function evaluations have to be done (where n is the problem dimension). For a real-world search problem, it seems unrealistic to expect the adaptation to improve strategy behavior before, say, $30n$ function evaluations (apart from global step size adaptation). A complete adaptation can take even $100n^2$ function evaluations (compare Section 7.3). Therefore, to get always the most out of adaptation, CPU resources should allow roughly between 100 and $200(n+3)^2$ function evaluations.

One main reason for the general robustness of ESs is that the selection scheme is solely based on a ranking of the population. The CMA-ES preserves this selection related robustness because no additional selection information is used. Furthermore, the CMA-ES preserves all invariance properties against transformations of the search space and of the objective function value, which facilitates any simple $(1, \lambda)$-ES with isotropic mutation distribution. Apart from initialization of object and strategy parameters, the CMA-ES yields an additional invariance against any linear transformation

of the search space—in contrast to any evolution strategy to our knowledge (besides Hansen et al. (1995) and Hansen and Ostermeier (1999)). Invariance properties are of major importance for the evaluation of any search strategy (compare Section 6).

The step from an ES with isotropic mutation distribution to an ES facilitating the adaptation of arbitrary normal mutation distributions, if successfully taken, can be compared with the step from a simple deterministic gradient strategy to a quasi-Newton method. The former follows the local gradient, which in a certain sense, also does an ES with isotropic mutation distribution. The latter approximates the inverse Hessian matrix in an iterative sequence without acquiring additional information on the search space. This is exactly what the CMA does for evolution strategies.

In simulations, the CMA-ES reliably approximates the inverse Hessian matrix of different objective functions. In addition, there are reported successful applications of the CMA-ES to real-world search problems (Alvers, 1998; Holste, 1998; Meyer, 1998; Lutz and Wagner, 1998a; Lutz and Wagner, 1998b; Olhofer et al., 2000; Bergener et al., 2001; Cerveri et al., 2001; Igel and von Seelen, 2001; Igel et al., 2001). Consequently, comparable to quasi-Newton methods, we expect this algorithm, or at least some quite similar method, based on its superior performance to become state-of-the-art for the application of ESs to real-world search problems.

Acknowledgments

This work was supported by the *Deutsche Forschungsgemeinschaft* under grant Re 215/12-1 and the *Bundesministerium für Bildung und Forschung* under grant 01 IB 404 A. We gratefully thank Iván Santibáñez-Koref for many helpful discussions and persistent support of our work. In addition, we thank Christian Igel and all responding users of the CMA-ES who gave us helpful suggestions from many different points of view.

A CMA-ES in MATLAB

```
% CMA-ES for non-linear function minimization
% See also http://www.bionik.tu-berlin.de/user/niko
function xmin=cmaes
  % Set dimension, fitness fct, stop criteria, start values...
  N=10; strfitnessfct = 'cigar';
  maxeval = 300*(N+2)^2; stopfitness = 1e-10; % stop criteria
  xmeanw = ones(N, 1); % object parameter start point (weighted mean)
  sigma = 1.0; minsigma = 1e-15; % step size, minimal step size

  % Parameter setting: selection,
  lambda = 4 + floor(3*log(N)); mu = floor(lambda/2);
  arweights = log((lambda+1)/2) - log(1:mu)'; % for recombination
  % parameter setting: adaptation
  cc = 4/(N+4); ccov = 2/(N+2^0.5)^2;
  cs = 4/(N+4); damp = 1/cs + 1;

  % Initialize dynamic strategy parameters and constants
  B = eye(N); D = eye(N); BD = B*D; C = BD*transpose(BD);
  pc = zeros(N,1); ps = zeros(N,1);
  cw = sum(arweights)/norm(arweights);
  chiN = N^0.5*(1-1/(4*N)+1/(21*N^2));

  % Generation loop
  counteval = 0; arfitness(1) = 2*abs(stopfitness)+1;
  while arfitness(1) > stopfitness & counteval < maxeval
```

```
    % Generate and evaluate lambda offspring
    for k=1:lambda
      % repeat the next two lines until arx(:,k) is feasible
      arz(:,k) = randn(N,1);
      arx(:,k) = xmeanw + sigma * (BD * arz(:,k));          % Eq.(13)
      arfitness(k) = feval(strfitnessfct, arx(:,k));
      counteval = counteval+1;
    end

    % Sort by fitness and compute weighted mean
    [arfitness, arindex] = sort(arfitness); % minimization
    xmeanw = arx(:,arindex(1:mu))*arweights/sum(arweights);
    zmeanw = arz(:,arindex(1:mu))*arweights/sum(arweights);

    % Adapt covariance matrix
    pc = (1-cc)*pc + (sqrt(cc*(2-cc))*cw) * (BD*zmeanw);    % Eq.(14)
    C = (1-ccov)*C + ccov*pc*transpose(pc);                 % Eq.(15)
    % adapt sigma
    ps = (1-cs)*ps + (sqrt(cs*(2-cs))*cw) * (B*zmeanw);     % Eq.(16)
    sigma = sigma * exp((norm(ps)-chiN)/chiN/damp);         % Eq.(17)

    % Update B and D from C
    if mod(counteval/lambda, N/10) < 1
      C=triu(C)+transpose(triu(C,1)); % enforce symmetry
      [B,D] = eig(C);
      % limit condition of C to 1e14 + 1
      if max(diag(D)) > 1e14*min(diag(D))
        tmp = max(diag(D))/1e14 - min(diag(D));
        C = C + tmp*eye(N); D = D + tmp*eye(N);
      end
      D = diag(sqrt(diag(D))); % D contains standard
    deviations now
      BD = B*D; % for speed up only
    end % if mod

    % Adjust minimal step size
    if sigma*min(diag(D)) < minsigma ...
          | arfitness(1) == arfitness(min(mu+1,lambda)) ...
          | xmeanw == xmeanw ...
             + 0.2*sigma*BD(:,1+floor(mod(counteval/lambda,N)))
      sigma = 1.4*sigma;
    end
  end % while, end generation loop

  disp([num2str(counteval) ': ' num2str(arfitness(1))]);
  xmin = arx(:, arindex(1)); % return best point of last generation

function f=cigar(x)
  f = x(1)^2 + 1e6*sum(x(2:end).^2);
```

References

Alvers, M. (1998). *Zur Anwendung von Optimierungsstrategien auf Potentialfeldmodelle.* Berliner geowissenschaftliche Abhandlungen, Reihe B: Geophysik. Selbstverlag Fachbereich Geowissenschaften, Freie Universität Berlin, Germany.

Arnold, D. (2000). Personal communication.

Bäck, T. and Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23.

Bäck, T., Hammel, U., and Schwefel, H.-P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):3–17.

Bergener, T., Bruckhoff, C., and Igel, C. (2001). Parameter optimization for visual obstacle detection using a derandomized evolution strategy. In Blanc-Talon, J. and Popesc, D., editors, *Imaging and Vision Systems: Theory, Assessment and Applications*, Advances in Computation: Theory and Practice. NOVA Science Books, Huntington, New York.

Beyer, H.-G. (1995). Toward a theory of evolution strategies: On the benefit of sex - the $(\mu/\mu, \lambda)$-theory. *Evolutionary Computation*, 3(1):81–110.

Beyer, H.-G. (1996a). On the asymptotic behavior of multirecombinant evolution strategies. In Voigt, H.-M. et al., editors, *Proceedings of PPSN IV, Parallel Problem Solving from Nature*, pages 122–133, Springer, Berlin, Germany.

Beyer, H.-G. (1996b). Toward a theory of evolution strategies: Self-adaptation. *Evolutionary Computation*, 3(3):311–347.

Beyer, H.-G. (1998). Mutate large, but inherit small! In Eiben, A. et al., editors, *Proceedings of PPSN V, Parallel Problem Solving from Nature*, pages 109–118, Springer, Berlin, Germany.

Beyer, H.-G. and Deb, K. (2000). On the desired behaviors of self-adaptive evolutionary algorithms. In Schoenauer, M. et al., editors, *Proceedings of PPSN VI, Parallel Problem Solving from Nature*, pages 59–68, Springer, Berlin, Germany.

Cerveri, P., Pedotti, A., and Borghese, N. (2001). Enhanced evolution strategies: A novel approach to stereo-camera calibration. *IEEE Transactions on Evolutionary Computation*, in press.

EVOTECH-7 (1997). Evotech—Einsatz der Evolutionsstrategie in Wissenschaft und Technik, 7. Zwischenbericht. Interim report of the Fachgebiet Bionik und Evolutionstechnik der Technischen Universität Berlin under grant 01 IB 404 A of the Bundesminister für Bildung, Wissenschaft, Forschung und Technologie.

Ghozeil, A. and Fogel, D. B. (1996). A preliminary investigation into directed mutations in evolutionary algorithms. In Voigt, H.-M. et al., editors, *Proceedings of PPSN IV, Parallel Problem Solving from Nature*, pages 329–335, Springer, Berlin, Germany.

Hansen, N. (1998). *Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie. Eine Untersuchung zur entstochastisierten, koordinatensystemunabhängigen Adaptation der Mutationsverteilung*. Mensch und Buch Verlag, Berlin, Germany. ISBN 3-933346-29-0.

Hansen, N. (2000). Invariance, self-adaptation and correlated mutations in evolution strategies. In Schoenauer, M. et al., editors, *Proceedings of PPSN VI, Parallel Problem Solving from Nature*, pages 355–364, Springer, Berlin, Germany.

Hansen, N. and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 312–317, IEEE Press, Piscataway, New Jersey.

Hansen, N. and Ostermeier, A. (1997). Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu_I, \lambda)$-CMA-ES. In Zimmermann, H.-J., editor, *Proceedings of EUFIT'97, Fifth European Congress on Intelligent Techniques and Soft Computing*, pages 650–654, Verlag Mainz, Aachen, Germany.

Hansen, N., Ostermeier, A., and Gawelczyk, A. (1995). On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. In Eshelman, L., editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 57–64, Morgan Kaufmann, San Francisco, California.

Herdy, M. (1993). The number of offspring as strategy parameter in hierarchically organized evolution strategies. *SIGBIO Newsletter*, 13(2):2–7.

Hildebrand, L., Reusch, B., and Fathi, M. (1999). Directed mutation—a new self adaptation for evolution strategies. In Angeline, P., editor, *Proceedings of the 1999 Congress on Evolutionary Computation CEC99*, pages 1550–1557, IEEE Press, Piscataway, New Jersey.

Holste, D. (1998). Modellkalibrierung am Beispiel von Kläranlagenmodellen. In Hafner, S., editor, *Industrielle Anwendungen Evolutionärer Algorithmen*, chapter 4, pages 37–44, Oldenbourg Verlag, München, Germany.

Holzheuer, C. (1996). Analyse der Adaptation von Verteilungsparametern in der Evolutionsstrategie. Diploma thesis, Fachgebiet Bionik und Evolutionstechnik der Technischen Universität Berlin, Berlin, Germany.

Igel, C. and von Seelen, W. (2001). Design of a field model for early vision: A case study of evolutionary algorithms in neuroscience. In *28th Goettingen Neurobiology Conference*. In press.

Igel, C., Erlhagen, W., and Jancke, D. (2001). Optimization of neural fields models. *Neurocomputing*, 36(1–4):225–233.

Lutz, T. and Wagner, S. (1998a). Drag reduction and shape optimization of airship bodies. *Journal of Aircraft*, 35(3):345–351.

Lutz, T. and Wagner, S. (1998b). Numerical shape optimization of natural laminar flow bodies. In *Proceedings of 21st ICAS Congress*, International Council of the Aeronautical Sciences and the American Institute of Aeronautics, Paper No. ICAS-98-2,9,4.

Meyer, M. (1998). Parameteroptimierung dynamischer Systeme mit der Evolutionsstrategie. Diploma thesis, Fachgebiet Bionik und Evolutionstechnik der Technischen Universität Berlin, Berlin, Germany.

Olhofer, M., Arima, T., Sonoda, T., and Sendhoff, B. (2000). Optimisation of a stator blade used in a transonic compressor cascade with evolution strategies. In Parmee, I., editor, *Adaptive Computing in Design and Manufacture (ACDM)*, pages 45–54. Springer Verlag, Berlin, Germany.

Ostermeier, A. (1992). An evolution strategy with momentum adaptation of the random number distribution. In Männer, R. and Manderick, B., editors, *Parallel Problem Solving from Nature, 2*, pages 197–206, North Holland, Amsterdam, The Netherlands.

Ostermeier, A. (1997). *Schrittweitenadaptation in der Evolutionsstrategie mit einem entstochastisierten Ansatz*. Ph.D. thesis, Technische Universität Berlin, Berlin, Germany.

Ostermeier, A. and Hansen, N. (1999). An evolution strategy with coordinate system invariant adaptation of arbitrary normal mutation distributions within the concept of mutative strategy parameter control. In Banzhaf, W. et al., editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-99*, pages 902–909, Morgan Kaufmann, San Francisco, California.

Ostermeier, A., Gawelczyk, A., and Hansen, N. (1994a). A derandomized approach to self-adaptation of evolution strategies. *Evolutionary Computation*, 2(4):369–380.

Ostermeier, A., Gawelczyk, A., and Hansen, N. (1994b). Step-size adaptation based on non-local use of selection information. In Davidor, Y. et al., editors, *Proceedings of PPSN IV, Parallel Problem Solving from Nature*, pages 189–198, Springer, Berlin, Germany.

Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. Second Edition. Cambridge University Press, Cambridge, England.

Rechenberg, I. (1973). *Evolutionsstrategie, Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stuttgart, Germany.

Rechenberg, I. (1994). *Evolutionsstrategie'94*. Frommann-Holzboog, Stuttgart, Germany.

Rechenberg, I. (1998). Personal communication.

Rudolph, G. (1992). On correlated mutations in evolution strategies. In Männer, R. and Manderick, B., editors, *Parallel Problem Solving from Nature, 2*, pages 105–114, North-Holland, Amsterdam, The Netherlands.

Schwefel, H.-P. (1981). *Numerical Optimization of Computer Models*. Wiley, Chichester, England.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology Series. John Wiley and Sons, New York, New York.

Whitley, D., Mathias, K., Rana, S., and Dzubera, J. (1996). Evaluating evolutionary algorithms. *Artificial Intelligence*, 85:245–276.