The CMA Evolution Strategy: A Comparing Review

Nikolaus Hansen

CoLab Computational Laboratory, ETH Zürich
ICoS Institute of Computational Science, ETH Zürich
nikolaus.hansen@inf.ethz.ch

Summary. Derived from the concept of self-adaptation in evolution strategies, the CMA (Covariance Matrix Adaptation) adapts the covariance matrix of a multi-variate normal search distribution. The CMA was originally designed to perform well with small populations. In this review, the argument starts out with large population sizes, reflecting recent extensions of the CMA algorithm. Commonalities and differences to continuous Estimation of Distribution Algorithms are analyzed. The aspects of reliability of the estimation, overall step size control, and independence from the coordinate system (invariance) become particularly important in small populations sizes. Consequently, performing the adaptation task with small populations is more intricate.

Nomenclature

Abbreviations

CMA Covariance Matrix Adaptation

EDA Estimation of Distribution Algorithm

EMNA Estimation of Multivariate Normal Algorithm

ES Evolution Strategy

$(\mu/\mu_{\{\mathrm{I,W}\}},\lambda)$-ES, evolution strategy with μ parents, with recombination of all μ parents, either Intermediate or Weighted, and λ offspring.

$\mathrm{OP} : \mathbb{R}^n \to \mathbb{R}^{n\times n}, \mathbf{x} \mapsto \mathbf{x}\mathbf{x}^{\mathrm{T}}$, denotes the outer product of a vector with itself, which is a matrix of rank one with eigenvector $\mathbf{x}$ and eigenvalue $\|\mathbf{x}\|^2$.

RHS Right Hand Side.

Greek symbols

$\lambda \geq 2$, population size, sample size, number of offspring.

$\mu \leq \lambda$ parent number, number of selected search points in the population.

N. Hansen: *The CMA Evolution Strategy: A Comparing Review*, StudFuzz **192**, 75–102 (2006)
www.springerlink.com

μ_{cov}, parameter for weighting between rank-one and rank-μ update, see (22).

$\mu_{\mathrm{eff}} = \left(\sum_{i=1}^{\mu} w_i^2\right)^{-1}$, the variance effective selection mass, see (5).

$\sigma^{(g)} \in \mathbb{R}_+$, step size.

Latin symbols

$\mathbf{B} \in \mathbb{R}^n$, an orthogonal matrix. Columns of $\mathbf{B}$ are eigenvectors of $\mathbf{C}$ with unit length and correspond to the diagonal elements of $\mathbf{D}$.

$\mathbf{C}^{(g)} \in \mathbb{R}^{n \times n}$, covariance matrix at generation g.

c_{ii}, diagonal elements of $\mathbf{C}$.

$c_{\mathrm{c}} \leq 1$, learning rate for cumulation for the rank-one update of the covariance matrix, see (17) and (33).

$c_{\mathrm{cov}} \leq 1$, learning rate for the covariance matrix update, see (11), (21), (22), and (34).

$c_\sigma < 1$, learning rate for the cumulation for the step size control, see (23) and (31).

$\mathbf{D} \in \mathbb{R}^n$, a diagonal matrix. The diagonal elements of $\mathbf{D}$ are square roots of eigenvalues of $\mathbf{C}$ and correspond to the respective columns of $\mathbf{B}$.

d_{ii}, diagonal elements of $\mathbf{D}$.

$d_\sigma \approx 1$, damping parameter for step size update, see (24), (28), and (32).

E Expectation value

$f : \mathbb{R}^n \to \mathbb{R}, \mathbf{x} \mapsto f(\mathbf{x})$, objective function (fitness function) to be minimized.

$f_{\mathrm{sphere}} : \mathbb{R}^n \to \mathbb{R}, \mathbf{x} \mapsto f_{\mathrm{sphere}}(\mathbf{x}) = \|\mathbf{x}\|^2 = \sum_{i=1}^{n} x_i^2$.

$g \in \mathbb{N}$, generation counter, iteration number.

$\mathbf{I} \in \mathbb{R}^{n \times n}$, Identity matrix, unity matrix.

$\mathbf{m}^{(g)} \in \mathbb{R}^n$, mean value of the search distribution at generation g.

$n \in \mathbb{N}_{>0}$, search space dimension, see f.

$\mathcal{N}(\mathbf{0}, \mathbf{I})$, multi-variate normal distribution with zero mean and unity covariance matrix. A vector distributed according to $\mathcal{N}(\mathbf{0}, \mathbf{I})$ has independent, $(0, 1)$-normally distributed components.

$\mathcal{N}(\mathbf{m}, \mathbf{C}) \sim \mathbf{m} + \mathcal{N}(\mathbf{0}, \mathbf{C})$, multi-variate normal distribution with mean $\mathbf{m} \in \mathbb{R}^n$ and covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$. The matrix $\mathbf{C}$ is symmetric and positive definite.

$\mathbf{p} \in \mathbb{R}^n$, evolution path, a sequence of successive (normalized) steps, the strategy takes over a number of generations.

w_i, where $i = 1, \ldots, \mu$, recombination weights, see also (3).

$\mathbf{x}_k^{(g+1)} \in \mathbb{R}^n$, k-th offspring from generation $g+1$. We refer to $\mathbf{x}^{(g+1)}$, as search point, or object parameters/variables, commonly used synonyms are candidate solution, or design variables.

$\mathbf{x}_{i:\lambda}^{(g+1)}$, i-th best individual out of $\mathbf{x}_1^{(g+1)}, \ldots, \mathbf{x}_\lambda^{(g+1)}$.

1 Introduction

We assume a search scenario, where we want to minimize an objective function $f : \mathbb{R}^n \to \mathbb{R}, \mathbf{x} \mapsto f(\mathbf{x})$.[1] The only accessible information on f are function values of evaluated search points. Our performance measure is the number of function evaluations needed to reach a certain function value. Many continuous domain evolutionary algorithms use a normal distribution to sample new search points. In this chapter, we focus on algorithms with a multi-variate normal search distribution, where the covariance matrix of the distribution *is not restricted to a priori*, e.g., not a diagonal matrix. Estimation of Distribution Algorithms (EDAs) falling into this class, include the Estimation of Multi-variate Normal Algorithm (EMNA), the Estimation of Gaussian Network Algorithm (EGNA) [15, 16], and the Iterated Density Estimation Evolutionary Algorithm (IDEA) [4]. Evolution Strategies (ESs) falling into this class include a $(\mu/\mu_\mathrm{I}, \lambda)$-ES[2] with self-adaptation of correlated mutations [19], and the ES with Covariance Matrix Adaptation (CMA) [10]. Originally, the CMA was interpreted as *derandomized self-adaptation* [12]: in contrast to the original self-adaptation, where changes of the distribution parameters obey their own stochastics, in the CMA, changes of the distribution parameters are *deterministically* linked to the object parameter variations. In this chapter, we will review the CMA from a different perspective revealing the close relationship to EDAs like the EMNA.

The Multi-variate Normal Distribution

Any normal distribution, $\mathcal{N}(\mathbf{m}, \mathbf{C})$, is uniquely determined by its mean $\mathbf{m} \in \mathbb{R}^n$ and its symmetric and positive definite covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$. Covariance matrices have an appealing geometrical interpretation: they can be uniquely identified with the (hyper-)ellipsoid $\{\mathbf{x} \in \mathbb{R}^n \,|\, \mathbf{x}^\mathrm{T}\mathbf{C}^{-1}\mathbf{x} = 1\}$, as shown in Fig. 1. The ellipsoid is a surface of equal density of the distribution. The principal axes of the ellipsoid correspond to the eigenvectors of $\mathbf{C}$, the squared axes lengths correspond to the eigenvalues. The eigendecomposition is denoted by $\mathbf{C} = \mathbf{B}\left(\mathbf{D}\right)^2\mathbf{B}^\mathrm{T}$, where columns of $\mathbf{B}$ are eigenvectors of $\mathbf{C}$ with unit length ($\mathbf{B}$ is orthogonal), and the squared diagonal elements of the diagonal matrix $\mathbf{D}$ are the corresponding eigenvalues.

The normal distribution $\mathcal{N}(\mathbf{m}, \mathbf{C})$ can be written in different forms.

$$\mathcal{N}(\mathbf{m}, \mathbf{C}) \sim \mathbf{m} + \mathcal{N}(\mathbf{0}, \mathbf{C}) \sim \mathbf{m} + \mathbf{B}\mathbf{D}\mathbf{B}^\mathrm{T}\mathcal{N}(\mathbf{0}, \mathbf{I}) \sim \mathbf{m} + \mathbf{B}\mathbf{D}\mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (1)$$

[1] In fact, the image needs not to be $\mathbb{R}$. Any totally ordered set is sufficient.

[2] $(\mu/\mu_\mathrm{I}, \lambda)$ refers to the non-elitist selection scheme with μ parents, Intermediate recombination of all μ parents, and λ offspring.

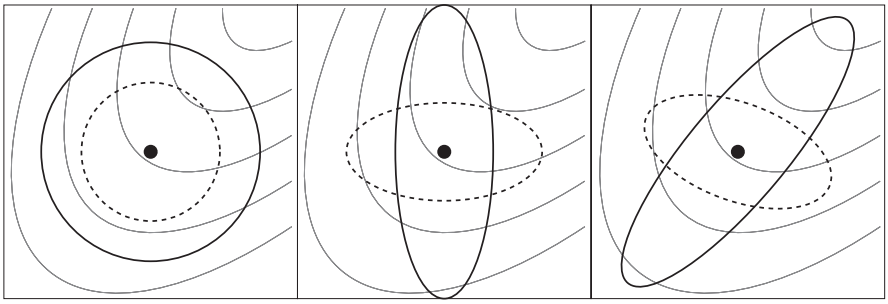


Fig. 1. Six ellipsoids, depicting one-σ lines of equal density of six different normal distributions, where $\sigma \in \mathbb{R}_+$, $\mathbf{D}$ is a diagonal matrix, and $\mathbf{C}$ is a positive definite full covariance matrix. Thin lines depict exemplary objective function contour lines

where "$\sim$" denotes equality in distribution and $\mathbf{I}$ denotes the identity matrix. If $\mathbf{D} = \sigma\mathbf{I}$, where $\sigma \in \mathbb{R}_+$, $\mathbf{C} = \sigma^2\mathbf{I}$ and the ellipsoid is isotropic (Fig. 1, left). If $\mathbf{B} = \mathbf{I}$, the ellipsoid is axis parallel oriented (middle). In the coordinate system given by $\mathbf{B}$, the distribution $\mathcal{N}(\mathbf{0}, \mathbf{C})$ is uncorrelated.

Objective

The objective of covariance matrix adaptation is, loosely speaking, to fit the search distribution to the contour lines of the objective function f to be minimized. In Fig. 1 the solid-line distribution in the right figure follows the objective function contour most suitably, and it is easy to foresee that it will help to approach the optimum the most. On convex-quadratic objective functions, setting the covariance matrix of the search distribution to the inverse Hessian matrix is equivalent to rescaling the ellipsoid function into a spherical one. We assume that the optimal covariance matrix equals the inverse Hessian matrix, up to a constant factor.[3] Consequently, the adaptation mechanism should aim to *approximate the inverse Hessian matrix*. Choosing a covariance matrix or choosing a respective affine linear transformation of the search space is equivalent [7].

Basic Equation

In the CMA evolution strategy, a population of new search points is generated by sampling a multi-variate normal distribution. The basic equation for sampling the search points, for generation number $g = 0, 1, 2, \ldots$, reads[4]

$$\mathbf{x}_k^{(g+1)} \sim \mathcal{N}\left(\mathbf{m}^{(g)}, \left(\sigma^{(g)}\right)^2 \mathbf{C}^{(g)}\right) \qquad \text{for } k = 1, \ldots, \lambda \tag{2}$$

[3] Even though there is good intuition and strong empirical evidence for this statement, a rigorous proof is missing.

[4] Framed equations belong to the final algorithm of a CMA evolution strategy.

where

$\sim$ denotes the same distribution on the left and right side.

$\mathcal{N}(\mathbf{m}^{(g)}, (\sigma^{(g)})^2\mathbf{C}^{(g)}) \sim \mathbf{m}^{(g)} + \sigma^{(g)}\mathcal{N}(\mathbf{0}, \mathbf{C}^{(g)}) \sim \mathbf{m}^{(g)} + \sigma^{(g)}\mathbf{B}^{(g)}\mathbf{D}^{(g)}\mathcal{N}(\mathbf{0}, \mathbf{I})$ is the multi-variate normal search distribution.

$\mathbf{x}_k^{(g+1)} \in \mathbb{R}^n$, k-th offspring (search point) from generation $g+1$.

$\mathbf{m}^{(g)} \in \mathbb{R}^n$, mean value of the search distribution at generation g.

$\sigma^{(g)} \in \mathbb{R}_+$, "overall" standard deviation, step size, at generation g.

$\mathbf{C}^{(g)} \in \mathbb{R}^{n\times n}$, covariance matrix at generation g.

$\lambda \geq 2$, population size, sample size, number of offspring.

To define the complete iteration step, the remaining question is, how to calculate $\mathbf{m}^{(g+1)}$, $\mathbf{C}^{(g+1)}$, and $\sigma^{(g+1)}$ for the next generation $g+1$. The next three sections will answer these questions, respectively.

2 Selection and Recombination: Choosing the Mean

The new mean $\mathbf{m}^{(g+1)}$ of the search distribution is a *weighted average of μ selected points* from the sample $\mathbf{x}_1^{(g+1)}, \ldots, \mathbf{x}_\lambda^{(g+1)}$:

$$\mathbf{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \, \mathbf{x}_{i:\lambda}^{(g+1)} \tag{3}$$

$$\sum_{i=1}^{\mu} w_i = 1, \qquad w_i > 0 \quad \text{for } \; i = 1, \ldots, \mu \tag{4}$$

where

$\mu \leq \lambda$ is the parent population size, i.e. the number of selected points.

$w_{i=1\ldots\mu} \in \mathbb{R}_+$, positive weight coefficients for recombination, where $w_1 \geq w_2 \geq \cdots \geq w_\mu > 0$. Setting $w_i = 1/\mu$, (3) calculates the mean value of μ selected points.

$\mathbf{x}_{i:\lambda}^{(g+1)}$, i-th best individual out of $\mathbf{x}_1^{(g+1)}, \ldots, \mathbf{x}_\lambda^{(g+1)}$ from (2). The index $i:\lambda$ denotes the index of the i-th ranked individual and $f(\mathbf{x}_{1:\lambda}^{(g+1)}) \leq f(\mathbf{x}_{2:\lambda}^{(g+1)}) \leq \cdots \leq f(\mathbf{x}_{\lambda:\lambda}^{(g+1)})$, where f is the objective function to be minimized.

Equation (3) implements *recombination* by taking a weighted sum of μ individuals, *and selection* by choosing $\mu < \lambda$ and/or assigning different weights w_i.

The measure

$$\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} w_i^2 \right)^{-1} \tag{5}$$

can be paraphrased as *variance effective selection mass*. From the definition of w_i we derive $1 \leq \mu_{\text{eff}} \leq \mu$, and $\mu_{\text{eff}} = \mu$ for equal recombination weights, i.e. $w_i = 1/\mu$ for all $i = 1 \ldots \mu$. Usually, $\mu_{\text{eff}} \approx \lambda/4$ indicates a reasonable setting of w_i. A typical setting would be $w_i \propto \mu - i + 1$, and $\mu \approx \lambda/2$.

3 Adapting the Covariance Matrix

In this section, the update of the covariance matrix, $\mathbf{C}$, is derived. We will start out estimating the covariance matrix from a single population of one generation (Sect. 3.1). For small populations this estimation is unreliable and an adaptation procedure has to be invented (Sect. 3.2). The adaptation procedure takes into account more than one generation and can be further enhanced by exploiting dependencies between successive steps (Sect. 3.3).

3.1 Estimating the Covariance Matrix

For the moment we assume that the population contains enough information to reliably estimate a covariance matrix from the population.[5] For the sake of convenience we assume $\sigma^{(g)} = 1$ in this section. For $\sigma^{(g)} \neq 1$ the discussion holds except for a constant factor.

Referring to (2), we can (re-)estimate the original covariance matrix $\mathbf{C}^{(g)}$ from the sample population, $\mathbf{x}_1^{(g+1)} \ldots \mathbf{x}_\lambda^{(g+1)}$, by

$$\mathbf{C}_{\text{emp}}^{(g+1)} = \frac{1}{\lambda - 1} \sum_{i=1}^{\lambda} \left(\mathbf{x}_i^{(g+1)} - \frac{1}{\lambda} \sum_{j=1}^{\lambda} \mathbf{x}_j^{(g+1)} \right) \left(\mathbf{x}_i^{(g+1)} - \frac{1}{\lambda} \sum_{j=1}^{\lambda} \mathbf{x}_j^{(g+1)} \right)^{\mathrm{T}} . \tag{6}$$

The empirical covariance matrix $\mathbf{C}_{\text{emp}}^{(g+1)}$ is an unbiased estimator of $\mathbf{C}^{(g)}$: assuming the $\mathbf{x}_i^{(g+1)}, i = 1 \ldots \lambda$, to be random variables (rather than a realized sample), we have that $\mathsf{E}\left[\mathbf{C}_{\text{emp}}^{(g+1)} \,\middle|\, \mathbf{C}^{(g)}\right] = \mathbf{C}^{(g)}$. Consider now a slightly different approach to get an estimator for $\mathbf{C}^{(g)}$.

[5] To re-estimate the covariance matrix, $\mathbf{C}$, from a $\mathcal{N}(\mathbf{0}, \mathbf{I})$ distributed sample such that $\mathtt{cond}(\mathbf{C}) < 10$ a sample size $\lambda \geq 4n$ is needed. The condition number of the matrix $\mathbf{C}$ is defined via the Euclidean norm: $\mathtt{cond}(\mathbf{C}) \stackrel{\text{def}}{=} \|\mathbf{C}\| \times \|\mathbf{C}^{-1}\|$, where $\|\mathbf{C}\| = \sup_{\|\mathbf{x}\|=1} \|\mathbf{C}\mathbf{x}\|$. For the covariance matrix $\mathbf{C}$ holds $\mathtt{cond}(\mathbf{C}) = \frac{\lambda_{\max}}{\lambda_{\min}} \geq 1$, where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalue of $\mathbf{C}$.

$$\mathbf{C}_{\lambda}^{(g+1)} = \frac{1}{\lambda}\sum_{i=1}^{\lambda}\left(\mathbf{x}_i^{(g+1)} - \mathbf{m}^{(g)}\right)\left(\mathbf{x}_i^{(g+1)} - \mathbf{m}^{(g)}\right)^{\mathrm{T}} \tag{7}$$

The matrix $\mathbf{C}_{\lambda}^{(g+1)}$ is an unbiased maximum likelihood estimator of $\mathbf{C}^{(g)}$. The remarkable difference between (6) and (7) is the reference mean value. For $\mathbf{C}_{\mathrm{emp}}^{(g+1)}$ it is the mean of the *actually realized* sample. For $\mathbf{C}_{\lambda}^{(g+1)}$ it is the true mean value of the distribution, $\mathbf{m}^{(g)}$ (see (2)). Therefore, the estimators $\mathbf{C}_{\mathrm{emp}}^{(g+1)}$ and $\mathbf{C}_{\lambda}^{(g+1)}$ can be interpreted differently: while $\mathbf{C}_{\mathrm{emp}}^{(g+1)}$ estimates the distribution variance *within the sampled points*, $\mathbf{C}_{\lambda}^{(g+1)}$ estimates variances of sampled *steps*, $\mathbf{x}_i^{(g+1)} - \mathbf{m}^{(g)}$. For the CMA the second approach is chosen.

Equation (7) re-estimates *the original* covariance matrix. To "estimate" a "better" covariance matrix (7) is modified and the same, *weighted selection* mechanism as in (3) is used [8].

$$\mathbf{C}_{\mu}^{(g+1)} = \sum_{i=1}^{\mu} w_i \left(\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}\right)\left(\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}\right)^{\mathrm{T}} \tag{8}$$

The matrix $\mathbf{C}_{\mu}^{(g+1)}$ is an estimator for the distribution of *selected steps*, just as $\mathbf{C}_{\lambda}^{(g+1)}$ is an estimator of the original distribution of steps before selection. Sampling from $\mathbf{C}_{\mu}^{(g+1)}$ tends to reproduce selected, i.e. *successful* steps, giving a justification for what a "better" covariance matrix means.

We compare (8) with the EMNA$_{global}$ approach [15, 16], where

$$\mathbf{C}_{\mathrm{EMNA}_{global}}^{(g+1)} = \frac{1}{\mu}\sum_{i=1}^{\mu}\left(\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g+1)}\right)\left(\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g+1)}\right)^{\mathrm{T}} , \tag{9}$$

and $\mathbf{m}^{(g+1)} = \frac{1}{\mu}\sum_{i=1}^{\mu}\mathbf{x}_{i:\lambda}^{(g+1)}$. The subtle difference is, again, the choice of the reference mean value.[6] Equation (8) estimates selected steps while in (9) the variance within the selected population is estimated. Equation (8) always reveals larger variances than (9), because the reference mean value in (9) is the minimizer for the variances. Moreover, in most conceivable selection situations (9) decreases the variances.

Figure 2 demonstrates the estimation results on *a linear* objective function for $\lambda = 150$, $\mu = 50$, and $w_i = 1/\mu$. While (8) increases the expected variance in direction of the gradient (where the selection takes place, here the diagonal), given ordinary settings for parent number μ and recombination weights $w_1, \ldots, w_n$, (9) decreases this variance! Therefore, (9) is highly susceptible to premature convergence, in particular with small parent populations, where the population cannot be expected to bracket the optimum at any time. However, for large values of μ in large populations with large initial variances, the impact of the different reference mean value can be marginal.

[6] Taking a weighted sum, $\sum_{i=1}^{\mu} w_i \ldots$, instead of the mean, $\frac{1}{\mu}\sum_{i=1}^{\mu}\ldots$, is an appealing, but less important, difference.

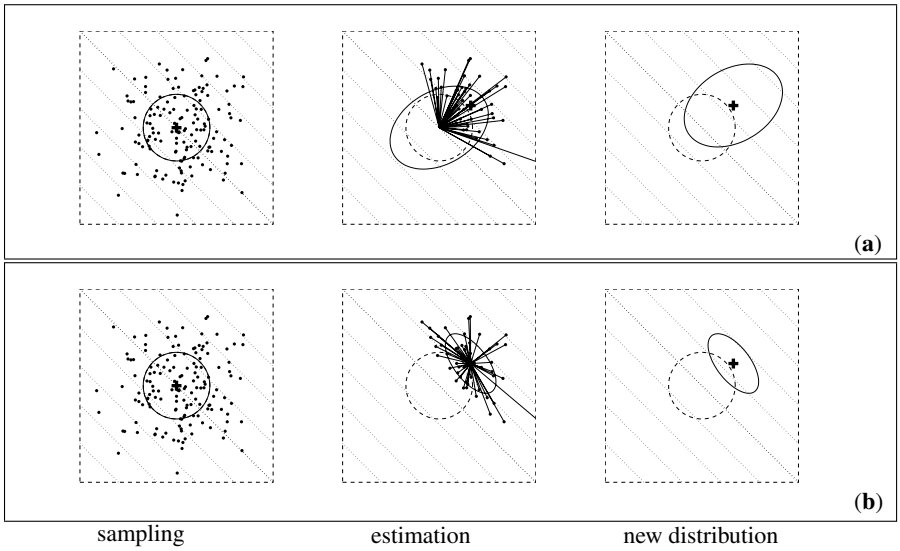


Fig. 2. Estimation of the covariance matrix on $f_{\text{linear}}(\mathbf{x}) = -\sum_{i=1}^{2} x_i$ to be minimized. Contour lines (*dotted*) indicate that the strategy should move toward the upper right corner. (**a**) Estimation of $\mathbf{C}_{\mu}^{(g+1)}$ according to (8), where $w_i = 1/\mu$; (**b**) estimation of $\mathbf{C}_{\text{EMNA}_{global}}^{(g+1)}$ according to (9). *Left*: sample of $\lambda = 150\ \mathcal{N}(\mathbf{0}, \mathbf{I})$ distributed points. *Middle*: the $\mu = 50$ selected points (*dots*) determining the entries for the estimation equation (*solid straight lines*), and the estimated covariance matrix (*ellipsoid*). *Right*: search distribution of the next generation. Given $w_i = 1/\mu$, (**a**) *increases* the expected variance in gradient direction for all $\mu < \lambda/2$, while (**b**) *decreases* this variance for any $\mu < \lambda$

To ensure $\mathbf{C}_{\mu}^{(g+1)}$ is a *reliable* estimator implementing (2), (3), and (8), the variance effective selection mass μ_{eff} (cf. (5)) must be large enough: to get condition numbers smaller than ten for $\mathbf{C}_{\mu}^{(g)}$ on $f_{\text{sphere}}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$, to our experience, $\mu_{\text{eff}} \approx 10n$ is needed. The next step is to circumvent this restriction on μ_{eff}.

3.2 Rank-μ-Update

To achieve *fast* search (opposite to more *robust* or more *global* search), e.g. competitive performance on f_{sphere}, the population size λ must be small. Because $\mu_{\text{eff}} \approx \lambda/4$ also μ_{eff} must be small and we may assume, e.g., $\mu_{\text{eff}} \leq 1 + \ln n$. Then, it is not possible to get a *reliable* estimator for a good covariance matrix from (8) alone. As a remedy, information from previous generations is added. For example, after a sufficient number of generations, the mean of the estimated covariance matrices from all generations,

$$\mathbf{C}^{(g+1)} = \frac{1}{g+1} \sum_{i=0}^{g} \frac{1}{{\sigma^{(i)}}^2} \mathbf{C}_{\mu}^{(i+1)} \tag{10}$$

becomes a reliable estimator for the selected steps. To make $\mathbf{C}_\mu^{(g)}$ from different generations comparable, the different $\sigma^{(i)}$ are incorporated. (Assuming $\sigma^{(i)} = 1$, (10) resembles the covariance matrix from EMNA$_i$ [16].)

In (10), all generation steps have the same weight. To assign recent generations a higher weight, exponential smoothing is introduced. Choosing $\mathbf{C}^{(0)} = \mathbf{I}$ to be the unity matrix and a learning rate $0 < c_{\text{cov}} \leq 1$, then $\mathbf{C}^{(g+1)}$ reads

$$\begin{aligned}\mathbf{C}^{(g+1)} &= (1 - c_{\text{cov}})\mathbf{C}^{(g)} + c_{\text{cov}} \frac{1}{{\sigma^{(g)}}^2} \mathbf{C}_\mu^{(g+1)} \\ &= (1 - c_{\text{cov}})\mathbf{C}^{(g)} + c_{\text{cov}} \sum_{i=1}^{\mu} w_i \, \text{OP}\left(\frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \right) \end{aligned} \tag{11}$$

where

$c_{\text{cov}} \leq 1$ learning rate for updating the covariance matrix. For $c_{\text{cov}} = 1$, no prior information is retained and $\mathbf{C}^{(g+1)} = \frac{1}{{\sigma^{(g)}}^2} \mathbf{C}_\mu^{(g+1)}$. For $c_{\text{cov}} = 0$, no learning takes place and $\mathbf{C}^{(g+1)} = \mathbf{C}^{(0)}$.

$\text{OP} : \mathbb{R}^n \to \mathbb{R}^{n \times n}, \mathbf{x} \mapsto \mathbf{x}\mathbf{x}^{\text{T}}$, denotes the outer product of a vector with itself.

This covariance matrix update is called rank-μ-update [9], because the sum of outer products in (11) is of rank $\min(\mu, n)$ (with probability one). Note that this sum can even consist of a single term, if $\mu = 1$.

The factor $1/c_{\text{cov}}$ can be interpreted as the *backward time horizon*. Because (11) expands to the weighted sum

$$\mathbf{C}^{(g+1)} = (1 - c_{\text{cov}})^{g+1}\mathbf{C}^{(0)} + c_{\text{cov}} \sum_{i=0}^{g} (1 - c_{\text{cov}})^{g-i} \frac{1}{{\sigma^{(i)}}^2} \, \mathbf{C}_\mu^{(i+1)} \quad , \tag{12}$$

the backward time horizon, Δg, where about 63% of the overall weight is summed up, is defined by

$$c_{\text{cov}} \sum_{i=g+1-\Delta g}^{g} (1 - c_{\text{cov}})^{g-i} \approx 0.63 \approx 1 - \frac{1}{\text{e}} \quad . \tag{13}$$

Resolving the sum yields

$$(1 - c_{\text{cov}})^{\Delta g} \approx \frac{1}{\text{e}} \quad , \tag{14}$$

and resolving for Δg, using the Taylor approximation for $\ln$, yields

$$\Delta g \approx \frac{1}{c_{\text{cov}}} \quad . \tag{15}$$

That is, approximately 37% of the information in $\mathbf{C}^{(g+1)}$ is older than $1/c_{\text{cov}}$ generations, and, according to (14), the original weight is reduced by a factor of 0.37 after approximately $1/c_{\text{cov}}$ generations.

The choice of c_{cov} is crucial. Small values lead to slow learning, too large values lead to a failure, because the covariance matrix degenerates. Fortunately, a good setting seems to be largely independent of the function to be optimized.[7] A first order approximation for a good choice is $c_{\mathrm{cov}} \approx \mu_{\mathrm{eff}}/n^2$. Therefore, the characteristic time horizon for (11) is roughly n^2/μ_{eff}.

Even for the learning rate $c_{\mathrm{cov}} = 1$, adapting the covariance matrix cannot be accomplished within one generation. The effect of the original sample distribution does not vanish until a sufficient number of generations. Assuming fixed search costs (number of function evaluations), a small population size λ allows a larger number of generations and therefore usually leads to a faster adaptation of the covariance matrix.

3.3 Cumulation: Utilizing the Evolution Path

We have used the selected steps, $(\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)})/\sigma^{(g)}$, to update the covariance matrix in (11). Because $\mathrm{OP}(\mathbf{x}) = \mathbf{x}\mathbf{x}^{\mathrm{T}} = \mathrm{OP}(-\mathbf{x})$, *the sign of the steps in* (11) *is irrelevant* – that is, the sign information is not used for calculating $\mathbf{C}^{(g+1)}$. To exploit this information, the so-called *evolution path* is introduced [10, 12].

We call a sequence of successive steps, the strategy takes over a number of generations, an evolution path. An evolution path can be expressed by a sum of consecutive steps. This summation is referred to as cumulation. To construct an evolution path, the step size σ is disregarded. For example, an evolution path of three steps can be constructed by the sum

$$\frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} + \frac{\mathbf{m}^{(g)} - \mathbf{m}^{(g-1)}}{\sigma^{(g-1)}} + \frac{\mathbf{m}^{(g-1)} - \mathbf{m}^{(g-2)}}{\sigma^{(g-2)}} \quad . \tag{16}$$

Again, we use exponential smoothing as in (11), to construct the evolution path, $\mathbf{p}_{\mathrm{c}} \in \mathbb{R}^n$, starting with $\mathbf{p}_{\mathrm{c}}^{(0)} = \mathbf{0}$.

$$\mathbf{p}_{\mathrm{c}}^{(g+1)} = (1 - c_{\mathrm{c}})\mathbf{p}_{\mathrm{c}}^{(g)} + \sqrt{c_{\mathrm{c}}(2 - c_{\mathrm{c}})\mu_{\mathrm{eff}}}\ \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \tag{17}$$

where

$\mathbf{p}_{\mathrm{c}}^{(g)} \in \mathbb{R}^n$, evolution path at generation g.

$c_{\mathrm{c}} \leq 1$. Again, $1/c_{\mathrm{c}}$ is the backward time horizon of the evolution path $\mathbf{p}_{\mathrm{c}}$ (compare (15)). A time horizon between $\sqrt{n}$ and n is reasonable.

The factor $\sqrt{c_{\mathrm{c}}(2 - c_{\mathrm{c}})\mu_{\mathrm{eff}}}$ is a normalization constant for $\mathbf{p}_{\mathrm{c}}^{(g)}$. For $c_{\mathrm{c}} = 1$ and $\mu_{\mathrm{eff}} = 1$, the factor reduces to one, and $\mathbf{p}_{\mathrm{c}}^{(g+1)} = (\mathbf{x}_{1:\lambda}^{(g+1)} - \mathbf{m}^{(g)})/\sigma^{(g)}$. The factor is chosen, such that

[7] We use the sphere model $f_{\mathrm{sphere}}(\mathbf{x}) = \sum_i x_i^2$ to empirically find a good setting for the parameter c_{cov}, dependent on n and μ_{eff}. The setting found was applicable to any non-noisy objective function we had tried so far.

$$\mathbf{p}_{\mathrm{c}}^{(g+1)} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}) \tag{18}$$

if

$$\mathbf{p}_{\mathrm{c}}^{(g)} \sim \frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}) \quad \text{for all } i = 1, \ldots, \mu \quad . \tag{19}$$

To derive (18) from (19) and (17) remark that

$$(1 - c_{\mathrm{c}})^2 + \sqrt{c_{\mathrm{c}}(2 - c_{\mathrm{c}})}^2 = 1 \quad \text{and} \quad \sum_{i=1}^{\mu} w_i \mathcal{N}_i(\mathbf{0}, \mathbf{C}) \sim \frac{1}{\sqrt{\mu_{\mathrm{eff}}}} \mathcal{N}(\mathbf{0}, \mathbf{C}) \quad . \tag{20}$$

The (rank-one) update of the covariance matrix $\mathbf{C}^{(g)}$ via the evolution path $\mathbf{p}_{\mathrm{c}}^{(g+1)}$ reads [10]

$$\mathbf{C}^{(g+1)} = (1 - c_{\mathrm{cov}})\mathbf{C}^{(g)} + c_{\mathrm{cov}} \mathbf{p}_{\mathrm{c}}^{(g+1)} \mathbf{p}_{\mathrm{c}}^{(g+1)\,\mathrm{T}} \quad . \tag{21}$$

An empirically validated choice for the learning rate in (21) is $c_{\mathrm{cov}} \approx 2/n^2$. For $c_{\mathrm{c}} = 1$ and $\mu = 1$, (21) and (11) are identical.

Using the evolution path for the update of $\mathbf{C}$ is a significant improvement of (11) for small μ_{eff}, because correlations between consecutive steps are exploited. The leading signs of steps, and the dependencies between consecutive steps, play a significant role in the resulting evolution path $\mathbf{p}_{\mathrm{c}}^{(g+1)}$. For $c_{\mathrm{c}} \approx 3/n$ the number of function evaluations needed to adapt a nearly optimal covariance matrix on cigar-like objective functions becomes $\mathcal{O}(n)$.

As a last step, we combine (11) and (21).

3.4 Combining Rank-μ-Update and Cumulation

The final CMA update of the covariance matrix combines (11) and (21), where μ_{cov} determines their relative weighting.

$$\mathbf{C}^{(g+1)} = (1 - c_{\mathrm{cov}})\mathbf{C}^{(g)} + \frac{c_{\mathrm{cov}}}{\mu_{\mathrm{cov}}} \underbrace{\mathbf{p}_{\mathrm{c}}^{(g+1)} \mathbf{p}_{\mathrm{c}}^{(g+1)\,\mathrm{T}}}_{\text{rank-one update}} + c_{\mathrm{cov}} \left(1 - \frac{1}{\mu_{\mathrm{cov}}}\right) \times \underbrace{\sum_{i=1}^{\mu} w_i \left(\frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}\right) \left(\frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}\right)^{\mathrm{T}}}_{\text{rank-}\mu\text{ update}} \tag{22}$$

where

$\mu_{\mathrm{cov}} \geq 1$. Choosing $\mu_{\mathrm{cov}} = \mu_{\mathrm{eff}}$ is most appropriate.

$c_{\mathrm{cov}} \approx \min(\mu_{\mathrm{cov}}, \mu_{\mathrm{eff}}, n^2)/n^2$.

Equation (22) reduces to (11) for $\mu_{\mathrm{cov}} \to \infty$ and to (21) for $\mu_{\mathrm{cov}} = 1$. The equation combines the advantages of (11) and (21). On the one hand, the information within the population of one generation is used efficiently by the rank-μ update. On the other hand, information of correlations between generations is exploited by using the evolution path for the rank-one update. The former is important in large populations, the latter is particularly important in small populations.

4 Step Size Control

We know two reasons to introduce a step size control in addition to the adaptation rule of (22) for $\mathbf{C}^{(g)}$.

1. The optimal overall step length cannot be well approximated by (22), in particular if μ_{eff} is chosen larger than one. For example, on $f_{\mathrm{sphere}}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$, the optimal step size σ equals approximately $\mu \sqrt{f_{\mathrm{sphere}}(\mathbf{x})}/n$, given $\mathbf{C}^{(g)} \approx \mathbf{I}$ and $\mu_{\mathrm{eff}} = \mu \ll n$ [2, 17]. This dependency on μ cannot be realized by (11), and is also not well approximated by (22).
2. The largest reliable learning rate for the covariance matrix update in (22) is too slow to achieve competitive change rates for the overall step length. To achieve optimal performance on f_{sphere} with an evolution strategy, the overall step length must decrease by a factor of approximately $\exp(0.202) \approx 1.22$ within n function evaluations, as can be derived from progress formulas [2, p. 229]. That is, the time horizon for the step length change must be proportional to n or shorter. From the learning rate c_{cov} in (22) it follows that the adaptation is too slow to perform competitively on f_{sphere} whenever $\mu_{\mathrm{eff}} \ll n$. This can be validated by simulations even for moderate dimensions, say, $n \geq 10$ and small μ_{eff}, say, $\mu_{\mathrm{eff}} \leq 1 + \ln n$.

To control the step size $\sigma^{(g)}$ we utilize an evolution path, i.e. a sum of successive steps (see page 84). The method is denoted *cumulative path length control*, cumulative step size control, or *cumulative step size adaptation*. The length of an evolution path is exploited, based on the following reasoning.

- If the evolution path is long, the single steps are pointing to similar directions. Loosely speaking, they are correlated. Because the steps are similar, the same distance can be covered by fewer but longer steps in the same directions – consequently the step size should be increased.
- If the evolution path is short, single steps cancel each other out. Loosely speaking, they are anti-correlated. If steps annihilate each other, the step size should be decreased.
- In the desired situation, the steps are approximately perpendicular in expectation and therefore uncorrelated.

To define "long" and "short", we compare the length of the evolution path with its *expected length under random selection*.[8] Under random selection consecutive steps are independent and therefore uncorrelated. If selection biases the evolution path to be longer then expected, σ will be increased, and, vice versa. If selection biases the evolution path to be shorter than expected, σ will be decreased. In the ideal situation, selection does not bias the length of the evolution path at all.

Because, in general, the expected length of the evolution path $\mathbf{p}_{\mathrm{c}}^{(g+1)}$ from (17) depends on its direction (compare (18)), a conjugate evolution path is constructed:

$$\mathbf{p}_{\sigma}^{(g+1)} = (1-c_{\sigma})\mathbf{p}_{\sigma}^{(g)} + \sqrt{c_{\sigma}(2-c_{\sigma})\mu_{\mathrm{eff}}}\ \mathbf{C}^{(g)^{-\frac{1}{2}}}\ \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \tag{23}$$

where

$\mathbf{p}_{\sigma}^{(g)} \in \mathbb{R}^n$ is the conjugate evolution path at generation g.

$c_{\sigma} < 1$. Again, $1/c_{\sigma}$ is the backward time horizon of the evolution path (compare (15)). For small μ_{eff}, a time horizon between $\sqrt{n}$ and n is reasonable.

$\sqrt{c_{\sigma}(2-c_{\sigma})\mu_{\mathrm{eff}}}$ is a normalization constant, see (17).

$\mathbf{C}^{(g)^{-\frac{1}{2}}} \stackrel{\mathrm{def}}{=} \mathbf{B}^{(g)}\mathbf{D}^{(g)^{-1}}\mathbf{B}^{(g)^{\mathrm{T}}}$, where $\mathbf{C}^{(g)} = \mathbf{B}^{(g)}\left(\mathbf{D}^{(g)}\right)^2\mathbf{B}^{(g)^{\mathrm{T}}}$ is an eigendecomposition of $\mathbf{C}^{(g)}$, where $\mathbf{B}^{(g)}$ is an orthonormal basis of eigenvectors, and the diagonal elements of the diagonal matrix $\mathbf{D}^{(g)}$ are square roots of the corresponding positive eigenvalues.

For $\mathbf{C}^{(g)} = \mathbf{I}$, (23) replicates (17), because $\mathbf{C}^{(g)^{-\frac{1}{2}}} = \mathbf{I}$ then. The transformation $\mathbf{C}^{(g)^{-\frac{1}{2}}}$ re-scales the step $\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}$ within the coordinate system given by $\mathbf{B}^{(g)}$. The single factors of the transformation $\mathbf{C}^{(g)^{-\frac{1}{2}}} = \mathbf{B}^{(g)}\mathbf{D}^{(g)^{-1}}\mathbf{B}^{(g)^{\mathrm{T}}}$ can be read as follows (from right to left):

$\mathbf{B}^{(g)^{\mathrm{T}}}$ rotates the space such that the columns of $\mathbf{B}^{(g)}$, i.e. the principle axes of the distribution $\mathcal{N}(\mathbf{0}, \mathbf{C}^{(g)})$, rotate into the coordinate axes. Elements of the resulting vector relate to projections onto the corresponding eigenvectors.

$\mathbf{D}^{(g)^{-1}}$ applies a (re-)scaling such that all axes become equally sized.

$\mathbf{B}^{(g)}$ rotates the result back into the original coordinate system. This last transformation ensures that directions of consecutive steps are comparable.

Consequently, the transformation $\mathbf{C}^{(g)^{-\frac{1}{2}}}$ makes the expected length of $\mathbf{p}_{\sigma}^{(g+1)}$ independent of its direction, and for any sequence of realized covariance matrices $\mathbf{C}^{(g)}_{g=0,1,2,\ldots}$ we have under random selection $\mathbf{p}_{\sigma}^{(g+1)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, given $\mathbf{p}_{\sigma}^{(0)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ [6].

To update $\sigma^{(g)}$, we "compare" $\|\mathbf{p}_{\sigma}^{(g+1)}\|$ with its expected length $\mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|$, that is

[8] Random selection means that the index $i:\lambda$ (compare (3)) is independent of the value of $\mathbf{x}_{i:\lambda}^{(g+1)}$ for all $i = 1, \ldots, \lambda$, e.g. $i:\lambda = i$.

$$\ln \sigma^{(g+1)} = \ln \sigma^{(g)} + \frac{c_\sigma}{d_\sigma} \left(\frac{\|\mathbf{p}_\sigma^{(g+1)}\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|} - 1 \right) \quad , \tag{24}$$

where

$d_\sigma \approx 1$, damping parameter, scales the change magnitude of $\ln \sigma^{(g)}$. The factor c_σ / d_σ is based on in-depth investigations of the algorithm [6].

$\mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\| = \sqrt{2}\,\Gamma(\frac{n+1}{2})/\Gamma(\frac{n}{2}) \approx \sqrt{n} + \mathcal{O}(1/n)$, expectation of the Euclidean norm of a $\mathcal{N}(\mathbf{0}, \mathbf{I})$ distributed random vector.

For $\|\mathbf{p}_\sigma^{(g+1)}\| = \mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|$ the second summand in (24) is zero, and $\sigma^{(g)}$ is unchanged, while $\sigma^{(g)}$ is increased for $\|\mathbf{p}_\sigma^{(g+1)}\| > \mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|$, and $\sigma^{(g)}$ is decreased for $\|\mathbf{p}_\sigma^{(g+1)}\| < \mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|$. The step size change is unbiased on the log scale, because $\mathsf{E}\left[\ln \sigma^{(g+1)} \,\middle|\, \sigma^{(g)} \right] = \ln \sigma^{(g)}$ for $\mathbf{p}_\sigma^{(g+1)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The role of unbiasedness is discussed in Sect. 6.

We show that successive steps taken by $\mathbf{m}^{(g)}$ are approximately ${\mathbf{C}^{(g)}}^{-1}$-conjugate. Equations (23) and (24) adapt σ such that the length of $\mathbf{p}_\sigma^{(g+1)}$ equals approximately $\mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|$. Starting from $(\mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|)^2 \approx \|\mathbf{p}_\sigma^{(g+1)}\|^2 = {\mathbf{p}_\sigma^{(g+1)}}^{\mathrm{T}} \mathbf{p}_\sigma^{(g+1)} = \mathrm{RHS}^{\mathrm{T}}\mathrm{RHS}$ of (23) and assuming that the expected squared length of ${\mathbf{C}^{(g)}}^{-\frac{1}{2}}(\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)})$ is unchanged by selection we get

$${\mathbf{p}_\sigma^{(g)}}^{\mathrm{T}} {\mathbf{C}^{(g)}}^{-\frac{1}{2}} (\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}) \approx 0 \quad , \tag{25}$$

and

$$\left({\mathbf{C}^{(g)}}^{\frac{1}{2}} \mathbf{p}_\sigma^{(g)} \right)^{\mathrm{T}} {\mathbf{C}^{(g)}}^{-1} \left(\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)} \right) \approx 0 \quad . \tag{26}$$

Given $1/c_{\mathrm{cov}} \gg 1$ and (25) we assume that ${\mathbf{p}_\sigma^{(g-1)}}^{\mathrm{T}} {\mathbf{C}^{(g)}}^{-\frac{1}{2}} (\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}) \approx 0$ and derive

$$\left(\mathbf{m}^{(g)} - \mathbf{m}^{(g-1)} \right)^{\mathrm{T}} {\mathbf{C}^{(g)}}^{-1} \left(\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)} \right) \approx 0 \quad . \tag{27}$$

That is, consecutive steps taken by the distribution mean become approximately ${\mathbf{C}^{(g)}}^{-1}$-conjugate.

Because $\sigma^{(g)} > 0$, (24) is equivalent to

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

The length of the evolution path is an intuitive and empirically well validated goodness measure for the overall step length. For $\mu_{\mathrm{eff}} > 1$ it is the best measure to our knowledge. Nevertheless, it fails to adapt nearly optimal step sizes on very noisy objective functions [3].

5 Simulations

The complete algorithm of the CMA evolution strategy is summarized in Appendix A, where all (default) parameter settings are given. We show single simulation runs of the CMA-ES on the test functions from Table 1, where $n = 8$.[9] All functions are convex-quadratic and can be written in the form $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{H}\mathbf{x}$, where $\mathbf{H}$ is the positive definite Hessian matrix. For each function we run an *axis parallel* version and a *randomly oriented* version. In the axis parallel version the Hessian is diagonal, because we choose $\mathbf{O} = \mathbf{I}$ (see Table 1). For the randomly oriented version each column of $\mathbf{O}$ is uniformly distributed on the unit hypersphere [12], fixed for each run. The matrix $\mathbf{O}$ defines the coordinate system where the Hessian is diagonal. On f_{sphere}, instead of $\mathbf{O}$, we set $\mathbf{B}^{(0)}$ to an arbitrary orthogonal matrix in the "randomly oriented" version. Furthermore, the diagonal elements of $\mathbf{D}^{(0)}$ are set to $d_{ii} = 10^{-3+3\frac{i-1}{n-1}}$ and $\mathbf{C}^{(0)} = \mathbf{B}^{(0)}\mathbf{D}^{(0)}\mathbf{D}^{(0)}{\mathbf{B}^{(0)}}^{\mathrm{T}}$. That is, the condition number of $\mathbf{C}^{(0)}$ equals to 10^6 and $\mathbf{C}$ has to become spherical (condition number one) during the adaptation (see Fig. 3). Further settings and initial values for the CMA-ES are according to Fig. 7 and Table 2 in Appendix A.

Table 1. Convex-quadratic test functions. $\mathbf{y} = \mathbf{O}\mathbf{x}$, where $\mathbf{O}$ is an orthogonal matrix

Function	$\mathtt{cond}(\mathbf{H})$	f_{stop}	Initial interval
$f_{\mathrm{sphere}}(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{n} x_i^2$	1	10^{-9}	$[0.1, 0.3]^n$
$f_{\mathrm{elli}}(\mathbf{x}) = \frac{1}{2}\sum_{i=1}^{n} 10^{6\frac{i-1}{n-1}} y_i^2$	10^6	10^{-9}	$[0.1, 0.3]^n$
$f_{\mathrm{cigtab}}(\mathbf{x}) = \frac{1}{2}\left(y_1^2 + 10^4\sum_{i=2}^{n-1} y_i^2 + 10^8 y_n^2\right)$	10^8	10^{-9}	$[5, 25]^n$
$f_{\mathrm{twoax}}(\mathbf{x}) = \frac{1}{2}\left(\sum_{i=1}^{\lfloor n/2\rfloor} y_i^2 + 10^6\sum_{i=\lfloor n/2\rfloor+1}^{n} y_i^2\right)$	10^6	10^{-9}	$[5, 25]^n$

By tracking eigenvalues and variances of the covariance matrix we can pursue, whether the objective of the covariance matrix adaptation is achieved, to approximate the inverse Hessian matrix of the objective function up to a constant factor. Eigenvalues of the Hessian correspond to the coefficients in Table 1 ($\{1, \ldots, 1\}$ for f_{sphere}, $\{10^{6\frac{i-1}{n-1}} \mid i = 1, \ldots, n\}$ for f_{elli}, $\{1, 10^4, 10^8\}$ for f_{cigtab}, and $\{1, 10^6\}$ for f_{twoax}).

The runs are shown in Fig. 3–6. The bottom figures show the square root of the eigenvalues of the covariance matrix, that is the lengths of the principal axes of the distribution ellipsoid, corresponding to diagonal elements, d_{ii}, of $\mathbf{D}$. After about 3500, 3500, 4000, and 5000 function evaluations, respectively, the adaptation has taken place and the axes lengths d_{ii} reflect the square root of the inverse eigenvalues of the Hessian, properly. Notice the striking connection between the matching of the lengths of the axes and the slope of the function value graph. Apart from effects of

[9] For exhaustive investigations of the CMA-ES on larger test function sets see [6, 8, 9, 11, 12] and for scale-up investigation up to $n = 320$ see [12].

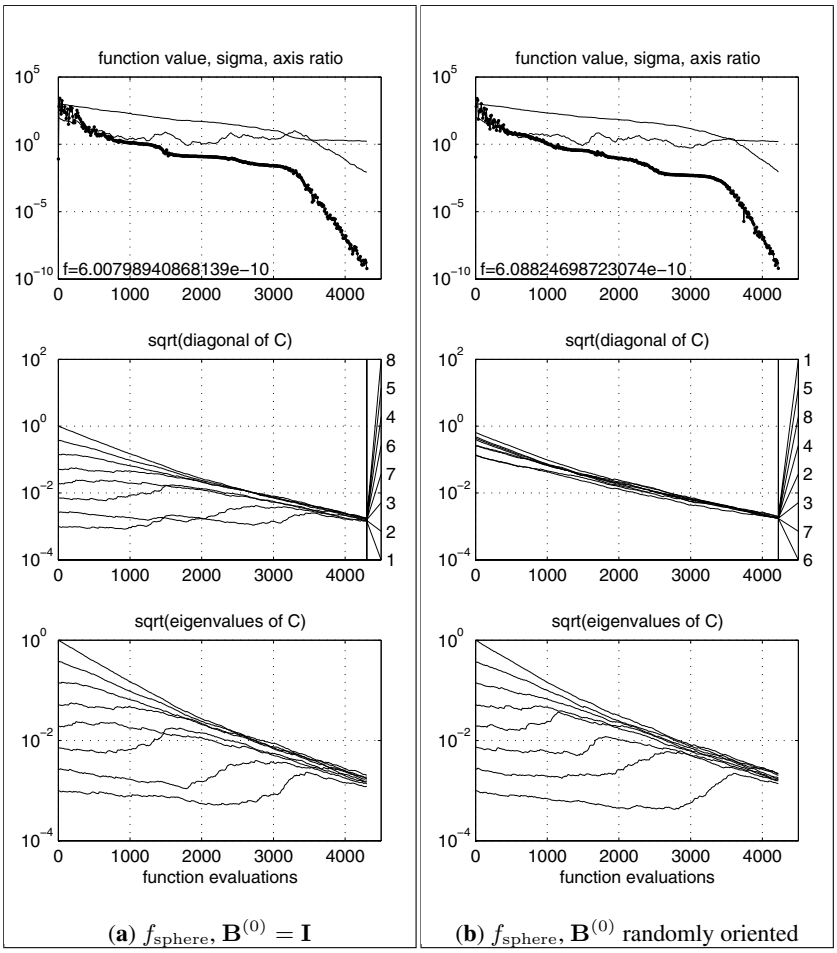
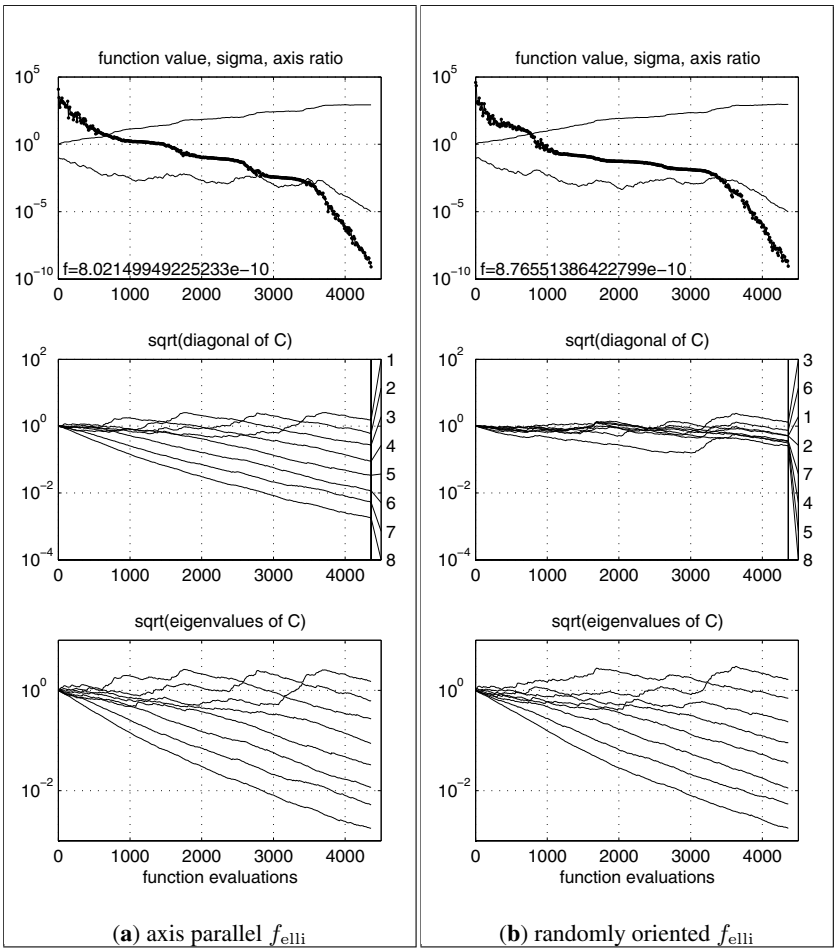


Fig. 3. Two runs on f_{sphere}, *where the initial covariance matrix,* $\mathbf{C}^{(0)}$*, is not spherical Above*: function value (*thick line*), σ (lower graph), $\sqrt{\mathtt{cond}(\mathbf{C})}$ (upper graph). *Middle*: $\sqrt{\mathtt{diag}(\mathbf{C})}$, index annotated. *Below*: square root of the eigenvalues of $\mathbf{C}$, i.e. $\mathtt{diag}(\mathbf{D}) = [d_{11}, \ldots, d_{nn}]$, versus number of function evaluations

different $\mathbf{x}^{(0)}$ and different random seeds, the upper and lower figures are equivalent for the axis parallel (**a**) and the randomly oriented version (**b**).

On axis parallel functions, the principal axes of the search distribution should become axis parallel after the adaptation has taken place. The middle figures show the square root of the diagonal elements of the covariance matrix, $\sqrt{c_{ii}}$. The elements $\sqrt{c_{ii}}$ align to the principal axes lengths d_{ii} in the left figures. That means, the search ellipsoid becomes axis parallel oriented (apart from subspaces of equal eigenvalues, where the final orientation is irrelevant). The final ordering of the $\sqrt{c_{ii}}$ reflects the ordering of the coefficients in the objective function. In contrast, the ordering of the

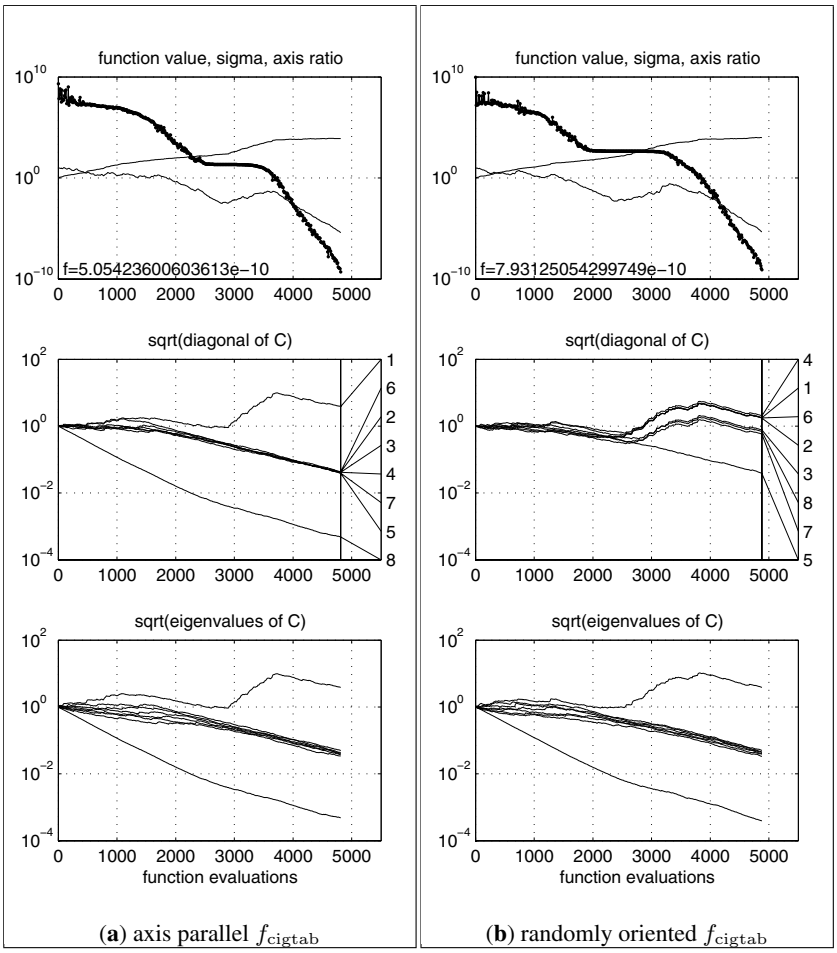


(**a**) axis parallel f_{elli} (**b**) randomly oriented f_{elli}

Fig. 4. Two runs on f_{elli} *Above*: function value (*thick line*), σ (lower graph), $\sqrt{\mathtt{cond}(\mathbf{C})}$ (upper graph). *Middle*: $\sqrt{\mathtt{diag}(\mathbf{C})}$, index annotated. *Below*: square root of the eigenvalues of $\mathbf{C}$, i.e. $\mathtt{diag}(\mathbf{D}) = [d_{11}, \ldots, d_{nn}]$, versus number of function evaluations

$\sqrt{c_{ii}}$ on the randomly oriented functions is arbitrary. The course of $\sqrt{c_{ii}}$ depends on the given coordinate system and therefore is remarkably different between (**a**) and (**b**). After the adaptation has taken place, in all cases the optimum is approached as fast as with an isotropic search distribution on f_{sphere}.

All the data give clear evidence that the inverse Hessian is well approximated. A measure for "well" can be derived from the runs on f_{sphere} (Fig. 3): the final condition number of $\mathbf{C}$ is smaller than five.



(**a**) axis parallel f_{cigtab} (**b**) randomly oriented f_{cigtab}

Fig. 5. Two runs on f_{cigtab} *Above*: function value (*thick line*), σ (lower graph), $\sqrt{\mathtt{cond}(\mathbf{C})}$ (upper graph). *Middle*: $\sqrt{\mathtt{diag}(\mathbf{C})}$, index annotated. *Below*: square root of the eigenvalues of $\mathbf{C}$, i.e. $\mathtt{diag}(\mathbf{D}) = [d_{11}, \ldots, d_{nn}]$, versus number of function evaluations

6 Discussion

In effect, the CMA-ES transforms any ellipsoid function into a spherical function. It is highly competitive on a considerable number of test functions [6, 8, 9, 11, 12] and was successfully applied to real world problems.[10] We discuss a few basic design principles.

[10] To our knowledge a few dozen successful applications have been published up to now, see http://www.icos.ethz.ch/software/evolutionary_computation/cmaapplications.pdf

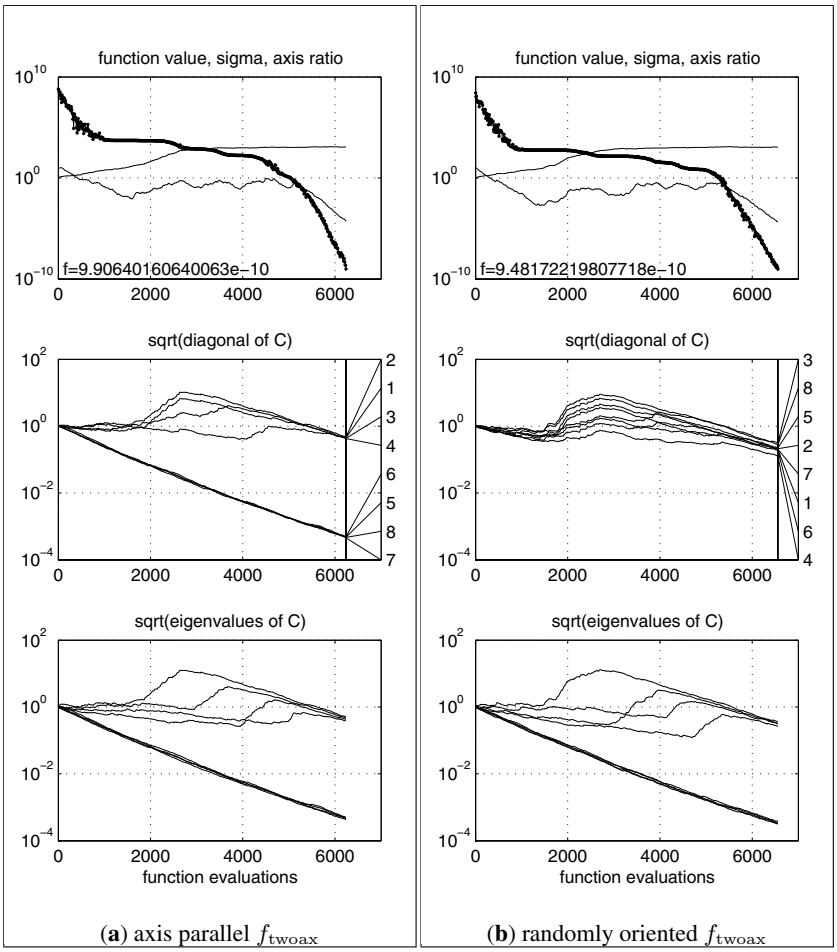


Fig. 6. Two runs on f_{twoax} *Above*: function value (*thick line*), σ (lower graph), $\sqrt{\mathtt{cond}(\mathbf{C})}$ (upper graph). *Middle*: $\sqrt{\mathtt{diag}(\mathbf{C})}$, index annotated. *Below*: square root of the eigenvalues of $\mathbf{C}$, i.e. $\mathtt{diag}(\mathbf{D}) = [d_{11}, \ldots, d_{nn}]$, versus number of function evaluations

Change Rates

A great deal of differences between continuous domain EDAs with multiple dependencies and the CMA-ES can be found in the change rates of distribution parameters. We refer to a change rate as the expected parameter change *per sampled search point*, given a certain selection situation. The CMA-ES separately controls change rates for the mean value of the distribution, $\mathbf{m}$, the covariance matrix, $\mathbf{C}$, and the step size, σ.

- The change rate for the mean value $\mathbf{m}$, given a fixed sample distribution, is determined by the parent number and the recombination weights. The larger μ_{eff}, the

smaller the possible change rate of $\mathbf{m}$ is. This is consistent with most EDAs. Interestingly, an explicit control parameter for the change rate for $\mathbf{m}$ is proposed in the Stochastic Hill Climbing with Learning by Vectors of Normal Distributions [18] and in the Population Based Incremental Learning for continuous domain (PBILc) [20], and even an *adaptive* control parameter is proposed in [21].

- The change rate of the covariance matrix $\mathbf{C}$ is explicitly controlled by the learning rate c_{cov} and detached from parent number and population size. The learning rate reflects the model complexity. An incremental update of distribution parameters from the selected population, similar to CMA, was already proposed in Population Based Incremental Learning (PBIL) [1] and expanded to continuous domain [5, 18, 20]. In contrast to CMA, these algorithms do not consider covariances. In EMNA$_a$ [15], both, mean and covariances are incrementally updated, but the change rates are equal for $\mathbf{m}$ and $\mathbf{C}$.
- The change rate of the step size σ is independent from the change rate of $\mathbf{C}$. The chosen time constant ensures a fast change of the overall step length in particular with small population sizes.

Invariance

Invariance properties of a search algorithm denote identical behavior on a set of objective functions. Invariances are highly desirable: they imply uniform performance on classes of functions and therefore allow for generalization of empirical results. Translation invariance should be taken for granted in continuous domain optimization. Further invariances to linear transformations of the search space are desirable. The CMA-ES and the EMNA approaches exhibit the following invariances.

- Invariance against order preserving (i.e. strictly monotonic) transformations of the objective function value. The algorithms only depend on *the ranking* of function values.
- Invariance against angle preserving transformations of the search space (rotation, reflection, and translation) if the initial search point(s) are transformed accordingly.
- Scale invariance if the initial scaling, e.g. $\sigma^{(0)}$, and the initial search point(s) are chosen accordingly.
- Invariance against any invertible linear transformation of the search space, $\mathbf{A}$, if the initial covariance matrix $\mathbf{C}^{(0)} = \mathbf{A}^{-1} \left(\mathbf{A}^{-1}\right)^{\mathrm{T}}$, and the initial search point(s) are transformed accordingly.

In our opinion, invariance should be a fundamental design criterion for any search algorithm.

Stationarity

An important design criterion for a stochastic search procedure is *unbiasedness* of variations of object and strategy parameters [13, 12]. Consider random selection, e.g. the objective function $f(\mathbf{x}) = \mathtt{rand}$ to be independent of $\mathbf{x}$. The population mean is unbiased if its expected value remains unchanged in the next generation, that

is $\mathsf{E}\left[\mathbf{m}^{(g+1)} \,\middle|\, \mathbf{m}^{(g)}\right] = \mathbf{m}^{(g)}$. For the population mean stationarity under random selection is a rather intuitive concept. In the CMA-ES, stationarity is respected for all parameters in (2). The distribution mean $\mathbf{m}$, the covariance matrix $\mathbf{C}$, and $\ln \sigma$ are unbiased. Unbiasedness of $\ln \sigma$ does not imply that σ is unbiased. Actually, under random selection, $\mathsf{E}\left[\sigma^{(g+1)} \,\middle|\, \sigma^{(g)}\right] > \sigma^{(g)}$, compare (24).[11]

For variances (or step sizes) a bias toward increase or decrease will entail the danger of divergence or premature convergence, respectively, whenever the selection pressure is low. Nevertheless, on noisy problems a properly controlled bias toward increase, even on the log scale, can be beneficial.

7 Summary and Conclusion

We have compared the CMA evolution strategy with EDAs that estimate the complete covariance matrix of a multi-variate normal search distribution. We summarize identified key points.

- Estimation principle: Most EDAs estimate the distribution parameters from a set of *selected points*. The CMA estimates them from a set of *selected steps*. Using steps is much less prone to premature convergence and supports explorative search behavior.
- Step size control: Methods to estimate or adapt the covariance matrix do not achieve good overall step lengths. In EDAs, step size control is usually absent, making a potential increase of the overall step length almost impossible. In the CMA-ES, the adaptation of the covariance matrix is complemented with step size control. The adjustment of the step size is based on a *different adaptation principle*. Cumulative path length control often adapts nearly optimal step sizes usually leading to considerably larger step lengths. This improves convergence speed *and* global search capabilities at the same time.
- Population size, adaptation, and change rates: Choosing the population size λ is always a compromise. Small λ lead to faster convergence, and large λ help to avoid local optima. To achieve a fast learning scheme for a covariance matrix
  1. the population size λ must be comparatively small (see end of Sect. 3.2) and
  2. an adaptation procedure must be established, where parameters are updated rather than estimated from scratch in every generation.

  Appropriate time constants for change rates of the population mean, of the covariance matrix, and of the overall step length are essential for competitive performance. In the CMA-ES, learning rates can be adjusted independently and only the change rate of the population mean is (indirectly) associated with the population size λ (via μ_{eff}). Determining different change rates for different parameters by adjusting learning rates is an open issue in EDAs.

[11] Alternatively, if (28) would have been designed to be unbiased for $\sigma^{(g+1)}$, this would presumably imply $\mathsf{E}\left[\ln \sigma^{(g+1)} \,\middle|\, \sigma^{(g)}\right] < \ln \sigma^{(g)}$, to our opinion a less desirable possibility.

- Invariances: To generalize empirical performance results, invariance properties are invaluable. Many EDAs use the given coordinate system to estimate the distribution, and are consequently not invariant to rotations of the search space. The CMA-ES is invariant under search space rotation and exhibits further invariances. Admittedly, a rotation invariant method cannot exploit separability of the objective function efficiently.[12]

Based on these key points the CMA can improve the performance on ill-conditioned and/or non-separable problems by orders of magnitude, leaving the performance on simple problems unchanged. In conclusion, the CMA evolution strategy is a state-of-the-art continuous domain evolutionary algorithm which is widely applicable and quasi parameter free.

Acknowledgments

The author wishes to gratefully thank Anne Auger, Christian Igel, Stefan Kern, and Fabrice Marchal for the many valuable comments on the manuscript.

References

1. S. Baluja and R. Caruana. Removing the genetics from standard genetic algorithm. In A. Prieditis and S. Russell, editors, *Proceedings of the International Conference on Machine Learning*, pp. 38–46. Morgan Kaufmann, 1995. 94
2. H.G. Beyer. *The Theory of Evolution Strategies*. Springer, 2001. 86
3. H.G. Beyer and D. Arnold. Qualms regarding the optimality of cumulative path length control in CSA/CMA-evolution strategies. *Evolutionary Computation*, 11(1):19–28, 2003. 88
4. P.A.N. Bosman and D. Thierens. Expanding from discrete to continuous estimation of distribution algorithms: The IDEA. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI. Lecture Notes in Computer Science 1917*, pp. 767–776, 2000. 77
5. M. Gallagher and M. Frean. Population-based continuous optimization and probabilistic modeling. Technical Report MG-1-2001, echnical report, School of Information Technology and Electrical Engineering, University of Queensland, 2001. 94
6. N. Hansen. *Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie*. Mensch und Buch Verlag, 1998. 87, 88, 89, 92
7. N. Hansen. Invariance, self-adaptation and correlated mutations in evolution strategies. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J.J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature - PPSN VI*, pp. 355–364. Springer, 2000. 78
8. N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In Xin Yao et al., editor, *Parallel Problem Solving from Nature - PPSN VIII*, pp. 282–291. Springer, 2004. 81, 89, 92, 99

[12] An n-dimensional *separable* problem can be solved by solving n 1-dimensional problems separately.

9. N. Hansen, S.D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003. 83, 89, 92
10. N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the 1996 IEEE Conference on Evolutionary Computation (ICEC '96)*, pp. 312–317, 1996. 77, 84, 85
11. N. Hansen and A. Ostermeier. Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu_I, \lambda)$-CMA-ES. In *Proceedings of the 5*th *European Congresson Intelligent Techniques and Soft Computing*, pp. 650–654, 1997. 89, 92
12. N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. 77, 84, 89, 92, 94, 99
13. K. Deb and H.G. Beyer. On self-adaptive features in real-parameter evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 5(3):250–270, 2001. 94
14. S. Kern, S.D. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos. Learning probability distributions in continuous evolutionary algorithms – a comparative review. *Natural Computing*, 3:77–112, 2004. 100
15. P. Larrañaga. A review on estimation of distribution algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*, pp. 80–90. Kluwer Academic Publishers, 2002. 77, 81, 94
16. P. Larrañaga, J. A. Lozano, and E. Bengoetxea. Estimation of distribution algorithms based on multivariate normal and Gaussian networks. Technical Report KZAA-IK-1-01, Dept. of Computer Science and Artificial Intelligence, University of the Basque Country, 2001. 77, 81, 83
17. I. Rechenberg. *Evolutionsstrategie '94*. Frommann-Holzboog, Stuttgart, Germany, 1994. 86
18. S. Rudlof and M. Köppen. Stochastic hill climbing by vectors of normal distributions. In *Proceedings of the First Online Workshop on Soft Computing (WSC1)*, 1996. Nagoya, Japan. 94
19. H.-P. Schwefel. *Evolution and Optimum Seeking*. John Wiley & Sons, Inc., 1995. 77
20. M. Sebag and A. Ducoulombier. Extending population-based incremental learning to continuos search spaces. In *Parallel Problem Solving from Nature - PPSN V*, pp. 418–427. Springer-Verlag, 1998. Berlin. 94
21. B. Yuan and M. Gallagher. Playing in continuous spaces: Some analysis and extension of population-based incremental learning. In Sarkar et al., editor, *Proc. Congress on Evolutionary Computation (CEC)*, pp. 443–450, 2003. 94

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

Figure 7 outlines the complete algorithm, summarizing (2), (3), (17), (22), (23), and (28). Symbols used are:

$\mathbf{x}_k^{(g+1)} \in \mathbb{R}^n$, for $k = 1, \ldots, \lambda$. Sample of λ search points of generation $g+1$.

$\mathcal{N}(\mathbf{m}, \mathbf{C})$, multi-variate normal distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{C}$.

$\mathbf{x}_{i:\lambda}^{(g+1)}$, i-th best point out of $\mathbf{x}_1^{(g+1)}, \ldots, \mathbf{x}_\lambda^{(g+1)}$ from (29). The index $i:\lambda$ denotes the index of the i-th ranked point and $f(\mathbf{x}_{1:\lambda}^{(g+1)}) \leq f(\mathbf{x}_{2:\lambda}^{(g+1)}) \leq \cdots \leq f(\mathbf{x}_{\lambda:\lambda}^{(g+1)})$.

Set parameters

Set parameters λ, μ, $w_{i=1\ldots\mu}$, c_σ, d_σ, c_c, μ_cov, and c_cov to their default values according to Table 2.

Initialization

Set evolution paths $\mathbf{p}_\sigma^{(0)} = \mathbf{0}$, $\mathbf{p}_\mathrm{c}^{(0)} = \mathbf{0}$, and covariance matrix $\mathbf{C}^{(0)} = \mathbf{I}$.
Choose step size $\sigma^{(0)} \in \mathbb{R}_+$ and distribution mean $\mathbf{m}^{(0)} \in \mathbb{R}^n$ problem dependent.[1]

For generation $g = 0, 1, 2, \ldots$, until stopping criterion met:

Sample new population of search points

$$\mathbf{x}_k^{(g+1)} \sim \mathcal{N}\left(\mathbf{m}^{(g)}, \left(\sigma^{(g)}\right)^2 \mathbf{C}^{(g)}\right) \qquad \text{for } k = 1, \ldots, \lambda \tag{29}$$

Selection and recombination

$$\mathbf{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i \, \mathbf{x}_{i:\lambda}^{(g+1)}, \quad \sum_{i=1}^{\mu} w_i = 1, \; w_i > 0 \tag{30}$$

Step size control

$$\mathbf{p}_\sigma^{(g+1)} = (1 - c_\sigma)\mathbf{p}_\sigma^{(g)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_\mathrm{eff}}\; \mathbf{C}^{(g)^{-\frac{1}{2}}} \; \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \tag{31}$$

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^{(g+1)}\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|} - 1\right)\right) \tag{32}$$

Covariance matrix adaptation

$$\mathbf{p}_\mathrm{c}^{(g+1)} = (1 - c_\mathrm{c})\mathbf{p}_\mathrm{c}^{(g)} + h_\sigma^{(g+1)} \sqrt{c_\mathrm{c}(2 - c_\mathrm{c})\mu_\mathrm{eff}} \; \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}} \tag{33}$$

$$\begin{aligned}\mathbf{C}^{(g+1)} = {} & (1 - c_\mathrm{cov})\mathbf{C}^{(g)} + \frac{c_\mathrm{cov}}{\mu_\mathrm{cov}}\left(\mathbf{p}_\mathrm{c}^{(g+1)}\mathbf{p}_\mathrm{c}^{(g+1)^\mathrm{T}} + \delta(h_\sigma^{(g+1)})\mathbf{C}^{(g)}\right) \\ & + c_\mathrm{cov}\left(1 - \frac{1}{\mu_\mathrm{cov}}\right)\sum_{i=1}^{\mu} w_i \, \mathrm{OP}\left(\frac{\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}\right)\end{aligned} \tag{34}$$

[1] The optimum should presumably be within the cube $\mathbf{m}^{(0)} \pm 2\sigma^{(0)}(1, \ldots, 1)^\mathrm{T}$. If the optimum is expected to be in $[0, 1]^n$ (initial interval) we may choose the initial search point, $\mathbf{m}^{(0)}$, uniformly randomly in $[0, 1]^n$, and $\sigma^{(0)} = 0.5$. Different search intervals Δs_i for different variables can be reflected by a different initialization of $\mathbf{C}$, in that the diagonal elements of $\mathbf{C}^{(0)}$ obey $c_{ii}^{(0)} = (\Delta s_i)^2$.

Fig. 7. The $(\mu_\mathrm{W}, \lambda)$-CMA evolution strategy. Symbols: see text

$\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} w_i^2\right)^{-1}$ is the variance effective selection mass. It holds $1 \leq \mu_{\text{eff}} \leq \mu$.

$\mathbf{C}^{(g)^{-\frac{1}{2}}} \stackrel{\text{def}}{=} \mathbf{B}^{(g)}\mathbf{D}^{(g)^{-1}}\mathbf{B}^{(g)^{\mathrm{T}}}$, where $\mathbf{C}^{(g)} = \mathbf{B}^{(g)}\mathbf{D}^{(g)}\mathbf{D}^{(g)}\mathbf{B}^{(g)^{\mathrm{T}}}$ is an eigendecomposition of the symmetric, positive definite covariance matrix $\mathbf{C}^{(g)}$. Columns of $\mathbf{B}^{(g)}$ are an orthonormal basis of eigenvectors, $\mathbf{B}^{(g)^{\mathrm{T}}}\mathbf{B}^{(g)} = \mathbf{B}^{(g)}\mathbf{B}^{(g)^{\mathrm{T}}} = \mathbf{I}$. Diagonal elements of the diagonal matrix $\mathbf{D}^{(g)}$ are square roots of the corresponding positive eigenvalues. The matrix $\mathbf{D}^{(g)}$ can be inverted by inverting its diagonal elements.

$\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\| = \sqrt{2}\,\Gamma(\frac{n+1}{2})/\Gamma(\frac{n}{2}) \approx \sqrt{n}\left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right)$.

$h_\sigma^{(g+1)} = \begin{cases} 1 \text{ if } \frac{\|\mathbf{p}_\sigma^{(g+1)}\|}{\sqrt{1-(1-c_\sigma)^{2(g+1)}}} < (1.5 + \frac{1}{n-0.5})\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\| \\ 0 \text{ otherwise} \end{cases}$

the Heaviside function $h_\sigma^{(g+1)}$ stalls the update of $\mathbf{p}_{\text{c}}^{(g)}$ in (17) if $\|\mathbf{p}_\sigma^{(g+1)}\|$ is large. This prevents a too fast increase of axes of $\mathbf{C}$ in a linear surrounding, i.e. when the step size is far too small. This is useful when the initial step size chosen is far too small or when the objective function changes in time.

$\delta(h_\sigma^{(g+1)}) = (1 - h_\sigma^{(g+1)})c_{\text{c}}(2 - c_{\text{c}}) \leq 1$ is of minor relevance and can be set to 0. In the (unusual) case of $h_\sigma^{(g+1)} = 0$, it substitutes for the second term from (33) in (34).

$\text{OP} : \mathbb{R}^n \to \mathbb{R}^{n\times n}, \mathbf{x} \mapsto \mathbf{x}\mathbf{x}^{\mathrm{T}}$, denotes the outer product of a vector with itself.

Default Parameters

The (external) strategy parameters are λ, μ, $w_{i=1\ldots\mu}$, c_σ, d_σ, c_{c}, μ_{cov}, and c_{cov}. Default strategy parameters values are given in Table 2. An in-depth discussion of most parameters is given in [12]. The default parameters of (37)–(39) are in particular chosen to be a robust setting and therefore, to our experience, applicable to a wide range of functions to be optimized. We do not recommend changing this setting. In contrast, the population size λ in (35) can be increased by the user.[13] If the λ-dependent default values for μ and w_i are used, the population size λ has a significant influence on the global search performance [8]. Increasing λ usually improves the global search capabilities and the robustness of the CMA-ES, at the price of a reduced convergence speed. The convergence speed decreases at most linearly with λ.

Implementation

We discuss a few implementational issues.

[13] Decreasing λ is not recommended. Too small values regularly have strong adverse effects on the performance.

Table 2. Default Strategy Parameters, where $\mu_{\text{eff}} = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \geq 1$ and $\sum_{i=1}^{\mu} w_i = 1$

Selection and Recombination:

$$\lambda = 4 + \lfloor 3 \ln n \rfloor, \quad \mu = \lfloor \lambda/2 \rfloor \,, \tag{35}$$

$$w_i = \frac{\ln(\mu + 1) - \ln i}{\sum_{j=1}^{\mu} (\ln(\mu + 1) - \ln j)} \quad \text{for } i = 1, \ldots, \mu \,, \tag{36}$$

Step size control:

$$c_\sigma = \frac{\mu_{\text{eff}} + 2}{n + \mu_{\text{eff}} + 3}, \quad d_\sigma = 1 + 2 \max\left(0, \sqrt{\frac{\mu_{\text{eff}} - 1}{n + 1}} - 1\right) + c_\sigma \tag{37}$$

Covariance matrix adaptation:

$$c_{\text{c}} = \frac{4}{n + 4}, \quad \mu_{\text{cov}} = \mu_{\text{eff}} \tag{38}$$

$$c_{\text{cov}} = \frac{1}{\mu_{\text{cov}}} \frac{2}{(n + \sqrt{2})^2} + \left(1 - \frac{1}{\mu_{\text{cov}}}\right) \min\left(1, \frac{2\mu_{\text{eff}} - 1}{(n + 2)^2 + \mu_{\text{eff}}}\right) \tag{39}$$

Multi-variate normal distribution: The distribution $\mathcal{N}(\mathbf{m}^{(g)}, {\sigma^{(g)}}^2 \mathbf{C}^{(g)})$ in (29) is distributed as $\mathbf{m}^{(g)} + \sigma^{(g)} \mathbf{B}^{(g)} \mathbf{D}^{(g)} \mathcal{N}(\mathbf{0}, \mathbf{I})$ (see ${\mathbf{C}^{(g)}}^{-\frac{1}{2}}$ above for the definitions). This can be used to generate the random vector on the computer, because $\mathcal{N}(\mathbf{0}, \mathbf{I})$ is a vector with independent, $(0, 1)$-normally distributed components that can be easily sampled on a computer.

Strategy internal numerical effort: In practice, the re-calculation of $\mathbf{B}^{(g)}$, $\mathbf{D}^{(g)}$, and ${\mathbf{C}^{(g)}}^{-\frac{1}{2}}$ does not need to be done until $\max(1, \lfloor 1/(10 n c_{\text{cov}}) \rfloor)$ generations. For reasonable c_{cov} values, this reduces the numerical effort due to the eigendecomposition from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ per generated search point. On a Pentium 4, 2.5 GHz processor the overall strategy internal time consumption is roughly $4(n + 2)^2 \times 10^{-8}$ seconds per function evaluation [14].

Flat fitness: In the case of equal function values for several individuals in the population, it is feasible to increase the step size (see lines 92–96 in the source code below).

Constraints: A simple, and occasionally sufficient, way to handle any type of boundaries and constraints is re-sampling unfeasible $\mathbf{x}_k^{(g+1)}$ until they become feasible.

B MATLAB Code

```
001 function xmin=purecmaes
002  % CMA-ES: Evolution Strategy with Covariance Matrix Adaptation for
003  % nonlinear function minimization.
004  %
005  % This code is an excerpt from cmaes.m and implements the key parts
```

```
% of the algorithm. It is intendend to be used for READING and
% UNDERSTANDING the basic flow and all details of the CMA *algorithm*.
% Computational efficiency is sometimes disregarded.

% --------------------  Initialization --------------------------------

% User defined input parameters (need to be edited)
strfitnessfct = 'felli'; % name of objective/fitness function
N = 10;                  % number of objective variables/problem dimension
xmean = rand(N,1);       % objective variables initial point
sigma = 0.5;             % coordinate wise standard deviation (step size)
stopfitness = 1e-10;  % stop if fitness < stopfitness (minimization)
stopeval = 1e3*N^2;   % stop after stopeval number of function evaluations

 % Strategy parameter setting: Selection
 lambda = 4+floor(3*log(N));  % population size, offspring number
 mu = floor(lambda/2);        % number of parents/points for recombination
 weights = log(mu+1)-log(1:mu)'; % muXone array for weighted recombination
 % lambda=12; mu=3; weights = ones(mu,1); % uncomment for (3_I,12)-ES
 weights = weights/sum(weights);     % normalize recombination weights array
 mueff=sum(weights)^2/sum(weights.^2); % variance-effective size of mu

 % Strategy parameter setting: Adaptation
 cc = 4/(N+4);    % time constant for cumulation for covariance matrix
 cs = (mueff+2)/(N+mueff+3); % t-const for cumulation for sigma control
 mucov = mueff;   % size of mu used for calculating learning rate ccov
 ccov = (1/mucov) * 2/(N+1.4)^2 + (1-1/mucov) * ...  % learning rate for
        ((2*mueff-1)/((N+2)^2+2*mueff));              % covariance matrix
 damps = 1 + 2*max(0, sqrt((mueff-1)/(N+1))-1) + cs; % damping for sigma

 % Initialize dynamic (internal) strategy parameters and constants
 pc = zeros(N,1); ps = zeros(N,1);   % evolution paths for C and sigma
 B = eye(N);                         % B defines the coordinate system
 D = eye(N);                         % diagonal matrix D defines the scaling
 C = B*D*(B*D)';                     % covariance matrix
 eigeneval = 0;                      % B and D updated at counteval == 0
 chiN=N^0.5*(1-1/(4*N)+1/(21*N^2));  % expectation of
                                     %   ||N(0,I)|| == norm(randn(N,1))

 % -------------------- Generation Loop --------------------------------

 counteval = 0;  % the next 40 lines contain the 20 lines of interesting code
 while counteval < stopeval

   % Generate and evaluate lambda offspring
   for k=1:lambda,
     arz(:,k) = randn(N,1);  % standard normally distributed vector
     arx(:,k) = xmean + sigma * (B*D * arz(:,k));   % add mutation    % Eq. 29
     arfitness(k) = feval(strfitnessfct, arx(:,k)); % objective function call
     counteval = counteval+1;
   end

   % Sort by fitness and compute weighted mean into xmean
   [arfitness, arindex] = sort(arfitness); % minimization
   xmean = arx(:,arindex(1:mu))*weights;   % recombination            % Eq. 30
   zmean = arz(:,arindex(1:mu))*weights;   % == sigma^-1*D^-1*B'*(xmean-xold)

   % Cumulation: Update evolution paths
   ps = (1-cs)*ps + (sqrt(cs*(2-cs)*mueff)) * (B * zmean);            % Eq. 31
```

```
065     hsig = norm(ps)/sqrt(1-(1-cs)^(2*counteval/lambda))/chiN < 1.5+1/(N-0.5);
066     pc = (1-cc)*pc + hsig * sqrt(cc*(2-cc)*mueff) * (B*D*zmean);       % Eq. 33
067
068     % Adapt covariance matrix C
069     C = (1-ccov) * C ...                    % regard old matrix         % Eq. 34
070          + ccov * (1/mucov) * (pc*pc' ...  % plus rank one update
071                                + (1-hsig) * cc*(2-cc) * C) ...
072          + ccov * (1-1/mucov) ...           % plus rank mu update
073            * (B*D*arz(:,arindex(1:mu))) ...
074            *  diag(weights) * (B*D*arz(:,arindex(1:mu)))';
075
076     % Adapt step size sigma

077     sigma = sigma * exp((cs/damps)*(norm(ps)/chiN - 1));               % Eq. 32
078
079     % Update B and D from C
080     if counteval - eigeneval > lambda/ccov/N/10  % to achieve O(N^2)
081       eigeneval = counteval;
082       C=triu(C)+triu(C,1)'; % enforce symmetry
083       [B,D] = eig(C);       % eigen decomposition, B==normalized eigenvectors
084       D = diag(sqrt(diag(D))); % D contains standard deviations now
085     end
086
087     % Break, if fitness is good enough
088     if arfitness(1) <= stopfitness
089       break;
090     end
091
092     % Escape flat fitness
093     if arfitness(1) == arfitness(min(1+floor(lambda/2), 2+ceil(lambda/4)))
094       sigma = sigma * exp(0.2+cs/damps);
095       disp('escape flat fitness');
096     end
097
098     disp([num2str(counteval) ': ' num2str(arfitness(1))]);
099
100   end % while, end generation loop
101
102   % -------------------- Ending Message --------------------------------
103
104   disp([num2str(counteval) ': ' num2str(arfitness(1))]);
105   xmin = arx(:, arindex(1)); % Return best point of last generation.
106                              % Notice that xmean is expected to be even
107                              % better.
108
109 % ---------------------------------------------------------------
110 function f=felli(x)
111   N = size(x,1); if N < 2 error('dimension must be greater one'); end
112   f=1e6.^((0:N-1)/(N-1)) * x.^2;  % condition number 1e6
```