A Computational Efficient Covariance Matrix Update and a (1+1)-CMA for Evolution Strategies

Christian Igel
Institut für Neuroinformatik
Ruhr-Universität Bochum
44780 Bochum, Germany
c.igel@ieee.org

Thorsten Suttorp
Institut für Neuroinformatik
Ruhr-Universität Bochum
44780 Bochum, Germany
thorsten.suttorp@rub.de

Nikolaus Hansen
Computational Science and Engineering Laboratory
Swiss Federal Institute of Technology (ETH) Zurich
Zurich, Switzerland
nikolaus.hansen@inf.ethz.ch

ABSTRACT

First, the covariance matrix adaptation (CMA) with rank-one update is introduced into the (1+1)-evolution strategy. An improved implementation of the 1/5-th success rule is proposed for step size adaptation, which replaces cumulative path length control. Second, an incremental Cholesky update for the covariance matrix is developed replacing the computational demanding and numerically involved decomposition of the covariance matrix. The Cholesky update can replace the decomposition only for the update without evolution path and reduces the computational effort from $O(n^3)$ to $O(n^2)$. The resulting (1+1)-Cholesky-CMA-ES is an elegant algorithm and the perhaps simplest evolution strategy with covariance matrix and step size adaptation. Simulations compare the introduced algorithms to previously published CMA versions.

Categories and Subject Descriptors

F.2.1 [**Numerical Algorithms and Problems**]: Computations on matrices; G.1.6 [**Optimization**]: Global Optimization; I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic methods

General Terms

Algorithms, Performance

Keywords

covariance matrix adaptation, rank-one update, Cholesky factors, evolution strategy

1. INTRODUCTION

Evolution strategies (ES) as well as evolutionary programming for real-valued optimization usually rely on Gaussian random variations. Appropriately adapting the covariance matrices of these mutations during optimization allows for learning and employing a variable metric for the search distribution. It is well known that such an automatic adaptation of the mutation distribution drastically improves the search performance on non-separable and/or badly scaled objective functions [12, 14, 9, 3, 10]. In general, altering the mutation distribution aims at making beneficial steps in the search space more likely. The so-called covariance matrix adaptation (CMA) explicitly employs this concept and has proven to be an effective adaptation mechanism for evolution strategies [9, 7, 10]. The CMA is applied with a $(\mu/\mu, \lambda)$-selection scheme and relies on an eigendecomposition to generate normally distributed random vectors according to the given covariance matrix, requiring $O(n^3)$ computations on n-dimensional objective functions.

This paper pursues two objectives. First, we introduce the CMA for the (1+1)-selection scheme, showing that complex (self-)adaptation of strategy parameters is not necessarily confined to non-elitist selection. We expect that this method (slightly) outperforms the $(\mu/\mu, \lambda)$-CMA-ES on unimodal non-noisy fitness functions. Second, we replace the eigendecomposition in the CMA by introducing an explicit update scheme for the Cholesky factors with a computational complexity of $O(n^2)$. Because the (1+1)-CMA-ES conducts the largest possible number of generations per function evaluation, the computational complexity of the eigendecomposition, which is naturally done every generation, is most relevant for this selection scheme.

In the following section, Gaussian mutations and the adaptation of the mutation distribution in ES are discussed. In Section 3 the elitist (1+1)-CMA-ES is described, and combined with the new update scheme in Section 4. The experiments for evaluating the performance of the (1+1)-CMA-ES, with and without evolution path, are presented in Section 5. Finally, the results are discussed.

2. GENERAL GAUSSIAN MUTATIONS IN EVOLUTION STRATEGIES

Evolution strategies (ES, [12, 14, 3, 10]) are one of the main branches of evolutionary algorithms and are particularly well suited for real-valued optimization. In their canonical form, the objective vector $\boldsymbol{x}_i^{(g+1)} \in \mathbb{R}^n$ of the ith offspring at generation g is created by

$$\boldsymbol{x}_i^{(g+1)} \leftarrow \underbrace{\boldsymbol{c}_i^{(g)}}_{\text{recombination}} + \underbrace{\boldsymbol{v}_i^{(g)}}_{\text{mutation}} .$$

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'06, July 8–12, 2006, Seattle, Washington, USA.
Copyright 2006 ACM 1-59593-186-4/06/0007 ...$5.00.

The vector $\boldsymbol{c}_i^{(g)}$ depends on the recombination scheme. For example, in case of weighted global intermediate recombination $\boldsymbol{c}_i^{(g)}$ is the weighted center of mass of the objective vectors in the current parent population. If no recombination is used, $\boldsymbol{c}_i^{(g)}$ is simply the objective vector of one of the parents. The mutation $\boldsymbol{v}_i^{(g)}$ is a realization of an n-dimensional random vector distributed according to a zero-mean Gaussian distribution with covariance matrix $\boldsymbol{C}_i^{(g)}$, that is,

$$\boldsymbol{v}_i^{(g)} \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{C}_i^{(g)}\right) \ .$$

The question arises how to sample the mutation distribution to generate $\boldsymbol{v}_i^{(g)}$. In general, this is done in two steps. First, the standard normal distribution is sampled to generate a realization of an n-dimensional normally distributed random vector $\boldsymbol{z}_i^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ with unit covariance matrix and zero mean. Second, this random vector is rotated and scaled by a linear transformation $\boldsymbol{A}_i^{(g)} \in \mathbb{R}^{n \times n}$ such that

$$\boldsymbol{A}_i^{(g)} \boldsymbol{z}_i^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_i^{(g)}) \text{ for } \boldsymbol{z}_i^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \ .$$

Thus, to sample a general multivariate normal distribution, the covariance matrix $\boldsymbol{C}_i^{(g)}$ is decomposed into Cholesky factors

$$\boldsymbol{C}_i^{(g)} = \boldsymbol{A}_i^{(g)} \boldsymbol{A}_i^{(g)^T} \ .$$

Every symmetric nonnegative definite matrix, such as covariance matrices, has Cholesky factors. This factorization is in general not unique, but under certain constraints (e.g., $\boldsymbol{A}_i^{(g)}$ lower triangular with nonnegative diagonal elements) a unique decomposition can be found. Computing Cholesky factors (by a triangular Cholesky, eigenvalue, or singular value decomposition algorithm) for a general covariance matrix requires $O(n^3)$ steps.

One of the decisive features of ES is that the covariance matrices are subject to adaptation. The general strategy is to alter the covariance matrices such that steps promising large fitness progress are sampled more often. There are typically two ways how the adaptation of the matrices is realized. First, the covariance matrix or its Cholesky factors can be parameterized, and these parameters can then be adapted. Either the parameterization or the adaptation rule has to ensure that the resulting matrices stay positive definite. This is done in self-adaptive ES (e.g., see [13]), where the parameters are changed by mutation. Here we consider a second way, where the covariance matrix is directly altered by additive updates of the form

$$\boldsymbol{C}^{(g+1)} = \alpha \boldsymbol{C}^{(g)} + \beta \boldsymbol{V}^{(g)} \ ,$$

where $\boldsymbol{V}^{(g)} \in \mathbb{R}^{n \times n}$ is positive definite and $\alpha, \beta \in \mathbb{R}_0^+$ are weighting factors (e.g., see [9, 7]). Let $\boldsymbol{v}^{(g)} \in \mathbb{R}$ be a mutation, that is, a step in the search space, promising large fitness progress. To increase the probability that $\boldsymbol{v}^{(g)}$ is sampled in the next iteration, the rank-one update

$$\boldsymbol{C}^{(g+1)} = \alpha \boldsymbol{C}^{(g)} + \beta \boldsymbol{v}^{(g)} \boldsymbol{v}^{(g)^T} \tag{1}$$

is appropriate. This update rule shifts the mutation distribution towards the line distribution $\mathcal{N}\left(\boldsymbol{0}, \boldsymbol{v}^{(g)} \boldsymbol{v}^{(g)^T}\right)$, which is the Gaussian distribution with the highest probability to generate $\boldsymbol{v}^{(g)}$ among all normal distributions with zero mean [9]. After the update, the new covariance matrix has to be decomposed into Cholesky factors to sample the distribution. If the covariance matrix updates occur frequently in the ES, say, at least once in every generation, the time needed for the factorization can dominate the computation time of the ES even for moderate n.

3. THE (1+1)-CMA-ES

To introduce the (1+1)-CMA-ES we combine the well known (1+1) selection scheme [12, 14, 3] with the covariance matrix adaptation (CMA) as proposed for the non-elitist (1,λ)- and (μ/μ,λ)-CMA-ES [8, 9]. The adaptation of the covariance matrix in the elitist (1+1)-CMA-ES is done using the rank-one update from the original, non-elitist (μ/μ,λ)-CMA-ES. However, the adaptation of the global step size has to be changed under elitist selection.

The application of the standard path length control to adapt the global step size is problematic in elitist selection, because the update of the evolution path (see [9] and below) would stall whenever the offspring is not successful. If in this case the evolution path is long, the step size might diverge. Therefore, the cumulative step size adaptation of the non-elitist CMA-ES is replaced by a success rule based step size control.

In the following we specify the algorithm, where we consider fitness functions $f : \mathbb{R}^n \to \mathbb{R}, \boldsymbol{x} \mapsto f(\boldsymbol{x})$ to be minimized. In the (1+1)-CMA-ES, parent and offspring encode candidate solution vectors $\boldsymbol{x}_{\text{parent}}, \boldsymbol{x}_{\text{offspring}} \in \mathbb{R}^n$. We keep track of an averaged success rate $\overline{p}_{\text{succ}} \in [0, 1]$, the global step size $\sigma \in \mathbb{R}_+$, an evolution path $\boldsymbol{p}_c \in \mathbb{R}^n$, and the covariance matrix $\boldsymbol{C} \in \mathbb{R}^{n \times n}$.

The algorithm is described in three routines. In the main part, `(1+1)-CMA-ES`, a new candidate solution is sampled and the parent solution $\boldsymbol{x}_{\text{parent}}$ is replaced depending on whether the new solution $\boldsymbol{x}_{\text{offspring}}$ is better than $\boldsymbol{x}_{\text{parent}}$.

Algorithm 1: (1+1)-CMA-ES

initialize $\boldsymbol{x}_{\text{parent}}$, σ, $\boldsymbol{C} = \boldsymbol{I}$, $\overline{p}_{\text{succ}} = p_{\text{succ}}^{\text{target}}$, $\boldsymbol{p}_c = \boldsymbol{0}$
repeat
  determine $\boldsymbol{A}$ such that $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{A}^T$
  $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
  $\boldsymbol{x}_{\text{offspring}} \leftarrow \boldsymbol{x}_{\text{parent}} + \sigma \boldsymbol{A}\boldsymbol{z}$
  `updateStepSize` $(\sigma, \lambda_{\text{succ}}, \overline{p}_{\text{succ}})$
  if $f(\boldsymbol{x}_{\text{offspring}}) \leq f(\boldsymbol{x}_{\text{parent}})$ **then**
    $\boldsymbol{x}_{\text{parent}} \leftarrow \boldsymbol{x}_{\text{offspring}}$
    `updateCov` $(\boldsymbol{C}, \boldsymbol{A}\boldsymbol{z}, \overline{p}_{\text{succ}}, \boldsymbol{p}_c)$
until stopping criterion is met

The variable

$$\lambda_{\text{succ}} \leftarrow \begin{cases} 1 & f(\boldsymbol{x}_{\text{offspring}}) \leq f(\boldsymbol{x}_{\text{parent}}) \\ 0 & \text{otherwise} \end{cases}$$

denotes whether the last mutation has been successful.

After sampling the new candidate solution, the step size is updated based on the success λ_{succ} with a learning rate c_p ($0 < c_p \leq 1$) using a target success rate $p_{\text{succ}}^{\text{target}}$.

Procedure `updateStepSize`$(\sigma, \lambda_{\text{succ}}, \overline{p}_{\text{succ}})$

$\overline{p}_{\text{succ}} \leftarrow (1 - c_p)\,\overline{p}_{\text{succ}} + c_p \lambda_{\text{succ}}$
$\sigma \leftarrow \sigma \cdot \exp\left(\frac{1}{d}\left(\overline{p}_{\text{succ}} - \frac{p_{\text{succ}}^{\text{target}}}{1 - p_{\text{succ}}^{\text{target}}}\left(1 - \overline{p}_{\text{succ}}\right)\right)\right)$

This procedure is rooted in the 1/5-success-rule proposed in [12] and generalizes and smoothes the implementation of the rule proposed in [10]. It implements the well-known

heuristic that the step size should be increased if the success rate (i.e., the fraction of offspring better than the parent) is high, and the step size should be decreased if the success rate is low. The rule is reflected in the argument to the exponential function. For $\overline{p}_{\text{succ}} > p_{\text{succ}}^{\text{target}}$ the argument is greater than zero and the step size increases; for $\overline{p}_{\text{succ}} < p_{\text{succ}}^{\text{target}}$ the argument is smaller than zero and the step size decreases; for $\overline{p}_{\text{succ}} = p_{\text{succ}}^{\text{target}}$ the argument becomes zero and no change of σ takes place.

The argument to the exponential function is always smaller than $1/d$ and larger than $-1/d$ if $p_{\text{succ}}^{\text{target}} < 0.5$ (a necessary condition for a functional success-based step size update rule). Therefore, the damping parameter d controls the rate of the step size adaptation.

If the new candidate solution is better than its parent, the covariance matrix is updated as in the original CMA-ES [9].

Procedure `updateCov`$(\boldsymbol{C}, \boldsymbol{y}, \overline{p}_{\text{succ}}, \boldsymbol{p}_c)$

1 **if** $\overline{p}_{\text{succ}} < p_{\text{thresh}}$ **then**
2 $\quad \boldsymbol{p}_c \leftarrow (1-c_c)\boldsymbol{p}_c + \sqrt{c_c(2-c_c)}\,\boldsymbol{y}$
3 $\quad \boldsymbol{C} \leftarrow (1-c_{\text{cov}})\boldsymbol{C} + c_{\text{cov}} \cdot \boldsymbol{p}_c \boldsymbol{p}_c^T$
4 **else**
5 $\quad \boldsymbol{p}_c \leftarrow (1-c_c)\boldsymbol{p}_c$
6 $\quad \boldsymbol{C} \leftarrow (1-c_{\text{cov}})\boldsymbol{C} + c_{\text{cov}} \cdot \left(\boldsymbol{p}_c \boldsymbol{p}_c^T + c_c(2-c_c)\boldsymbol{C}\right)$

The update of the evolution path $\boldsymbol{p}_c$ depends on the value of $\overline{p}_{\text{succ}}$. If the smoothed success rate $\overline{p}_{\text{succ}}$ is high, that is, above $p_{\text{thresh}} < 0.5$, the update of the evolution path $\boldsymbol{p}_c$ is stalled. This prevents a too fast increase of axes of $\boldsymbol{C}$ when the step size is far too small, for example, in a (close to) linear surrounding. If the smoothed success rate $\overline{p}_{\text{succ}}$ is low, the update of $\boldsymbol{p}_c$ is accomplished obeying an exponential smoothing. The constants c_c and c_{cov} $(0 \leq c_{\text{cov}} < c_c \leq 1)$ are learning rates for the evolution path and the covariance matrix, respectively. The factor $\sqrt{c_c(2-c_c)}$ normalizes the variance of $\boldsymbol{p}_c$ viewed as a random variable (see [9]). The evolution path $\boldsymbol{p}_c$ is used to update the covariance matrix. The new covariance matrix is a weighted mean of the old covariance matrix and the outer product $\boldsymbol{p}_c \boldsymbol{p}_c^T$. In the second case (line 5), the second summand in the update of $\boldsymbol{p}_c$ is missing and the length of $\boldsymbol{p}_c$ shrinks. Although of minor relevance, the term $c_c(2-c_c)\boldsymbol{C}$ (line 6) compensates for this shrinking in $\boldsymbol{C}$.

Table 1: Default parameters for the (1+1)-CMA Evolution Strategy.

Step size control:
$d = 1 + \frac{n}{2}$, $\quad p_{\text{succ}}^{\text{target}} = \frac{2}{11}$, $\quad c_p = \frac{1}{12}$
Covariance matrix adaptation:
$c_c = \frac{2}{n+2}$, $\quad c_{\text{cov}} = \frac{2}{n^2+6}$, $\quad p_{\text{thresh}} = 0.44$

Initial values are set to $\overline{p}_{\text{succ}} = p_{\text{succ}}^{\text{target}}$, $\boldsymbol{p}_c = \boldsymbol{0}$, and $\boldsymbol{C} = \boldsymbol{I}$, where $p_{\text{succ}}^{\text{target}}$ is given in Table 1. The initial candidate solution $\boldsymbol{x}_{\text{parent}} \in \mathbb{R}^n$ and the initial $\sigma \in \mathbb{R}_+$ must be chosen problem dependent. The optimum should presumably be within the cube $\boldsymbol{x}_{\text{parent}} \pm 2\,\sigma(1,\ldots,1)^T$.

The (external) strategy parameters of the (1+1)-CMA-ES are target success probability $p_{\text{succ}}^{\text{target}}$, step size damping d, success rate averaging parameter c_p, cumulation time horizon parameter c_c, and covariance matrix learning rate c_{cov}. Default values are given in Table 1. Most default values are derived from the precursor algorithms. They have been validated by simulations on simple test functions for different dimensions, where the parameters were varied in a large interval, even beyond their working range limits. The final parameter choice was made to stay away from these limits.

In particular, the parameters for the covariance matrix adaptation are similar to those for the standard non-elitist CMA-ES.

4. THE (1+1)-CHOLESKY-CMA-ES

4.1 Efficient Covariance Matrix Update

In general, each factorizing of a covariance matrix (compare line 3 in Algorithm 1) requires $O(n^3)$ operations. Thus, in an ES with additive covariance matrix adaptation the Cholesky factorization of the covariance matrix is the computationally dominating factor apart from the fitness function evaluations.

We propose not to factorize the covariance matrix, but to use an incremental rank-one update rule for the Cholesky factorization, and thereby to reduce the computational complexity to $O(n^2)$. The idea is never to compute the covariance matrix explicitly, but to operate on Cholesky factors only. We consider Cholesky factors that are general $n \times n$-matrices, and give a general update rule for ES. The proposed technique belongs to the well known rank-one updates and is frequently used in the domain of Kalman filtering [4].

Using $\boldsymbol{v}^{(g)} = \boldsymbol{A}^{(g)}\boldsymbol{z}^{(g)}$ with $\boldsymbol{z}^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ we can rewrite the rank-one update of the covariance matrix equation (1) as

$$\boldsymbol{C}^{(g+1)} = \alpha\boldsymbol{C}^{(g)} + \beta\boldsymbol{A}^{(g)}\boldsymbol{z}^{(g)}\left[\boldsymbol{A}^{(g)}\boldsymbol{z}^{(g)}\right]^T \quad . \tag{2}$$

We want to turn this update for $\boldsymbol{C}^{(g)}$ into an update for $\boldsymbol{A}^{(g)}$. Before we derive the new update scheme, we prove a useful lemma.

LEMMA 1. *Let $\boldsymbol{w} \in \mathbb{R}^n$ be a column vector. Then, for*

$$\varsigma = \frac{1}{\|\boldsymbol{w}\|^2}\left(\sqrt{1+\|\boldsymbol{w}\|^2} - 1\right)$$

the following equality holds

$$\boldsymbol{I} + \boldsymbol{w}\boldsymbol{w}^T = (\boldsymbol{I} + \varsigma\boldsymbol{w}\boldsymbol{w}^T)(\boldsymbol{I} + \varsigma\boldsymbol{w}\boldsymbol{w}^T) \quad .$$

PROOF. We have

$$\begin{aligned} & (\boldsymbol{I} + \varsigma\boldsymbol{w}\boldsymbol{w}^T)(\boldsymbol{I} + \varsigma\boldsymbol{w}\boldsymbol{w}^T) \\ = \;& \boldsymbol{I} + 2\varsigma\boldsymbol{w}\boldsymbol{w}^T + \varsigma^2\|w\|^2\boldsymbol{w}\boldsymbol{w}^T \\ = \;& \boldsymbol{I} + \left(2\varsigma + \varsigma^2\|\boldsymbol{w}\|^2\right)\boldsymbol{w}\boldsymbol{w}^T \\ = \;& \boldsymbol{I} + \boldsymbol{w}\boldsymbol{w}^T \quad . \end{aligned}$$

In order to check the last equality, it has to be verified that $\left(2\varsigma + \varsigma^2\|\boldsymbol{w}\|^2\right) = 1$, and this is true for ς as defined in the Lemma. □

Now, we present the update-rule for the Cholesky factors of the covariance matrix.

THEOREM 1. *Let $\boldsymbol{C}_t \in \mathbb{R}^{n\times n}$ be a symmetric nonnegative definite matrix with Cholesky factorization $\boldsymbol{C}_t = \boldsymbol{A}_t \boldsymbol{A}_t{}^T$. Assume further that $\boldsymbol{C}_t$ is updated using*

$$\boldsymbol{C}_{t+1} = \alpha \boldsymbol{C}_t + \beta \boldsymbol{v}_t \boldsymbol{v}_t^T,$$

with $\boldsymbol{v}_t = \boldsymbol{A}_t \boldsymbol{z}_t$ a column vector and $\alpha, \beta \in \mathbb{R}^+$. Then, the Cholesky factorization $\boldsymbol{A}_{t+1}\boldsymbol{A}_{t+1}{}^T$ of the matrix $\boldsymbol{C}_{t+1}$ is given by

$$\boldsymbol{A}_{t+1} = \sqrt{\alpha}\boldsymbol{A}_t + \frac{\sqrt{\alpha}}{\|\boldsymbol{z}_t\|^2}\left(\sqrt{1+\frac{\beta}{\alpha}\|\boldsymbol{z}_t\|^2} - 1\right)[\boldsymbol{A}_t \boldsymbol{z}_t]\,\boldsymbol{z}_t^T \ .$$

PROOF. The proof of this theorem is straightforward by calculating the updated covariance matrix

$$\begin{aligned}
\boldsymbol{C}_{t+1} &= \alpha \boldsymbol{C}_t + \beta \boldsymbol{v}_t \boldsymbol{v}_t^T \\
&= \alpha \boldsymbol{A}_t \boldsymbol{A}_t^T + \beta \boldsymbol{A}_t \boldsymbol{z}_t \boldsymbol{z}_t^T \boldsymbol{A}_t^T \\
&= \alpha \boldsymbol{A}_t \left(\boldsymbol{I} + \boldsymbol{z}_t \frac{\beta}{\alpha} \boldsymbol{z}_t^T\right) \boldsymbol{A}_t^T \\
&= \boldsymbol{A}_t \sqrt{\alpha}\left(\boldsymbol{I} + \varsigma \boldsymbol{z}_t \frac{\beta}{\alpha} \boldsymbol{z}_t^T\right)\left(\boldsymbol{I} + \varsigma \boldsymbol{z}_t \frac{\beta}{\alpha} \boldsymbol{z}_t^T\right)\sqrt{\alpha}\boldsymbol{A}_t^T \ ,
\end{aligned}$$

where $\varsigma = \frac{\alpha}{\beta}\frac{1}{\|\boldsymbol{z}_t\|^2}\left(\sqrt{1+\frac{\beta}{\alpha}\|\boldsymbol{z}_t\|^2} - 1\right)$. For the factorization in the last equality we used Lemma 1 with $\boldsymbol{w} = \sqrt{\frac{\beta}{\alpha}}\boldsymbol{z}_t$. It directly follows that

$$\begin{aligned}
\boldsymbol{A}_{t+1} &= \sqrt{\alpha}\boldsymbol{A}_t \left(\boldsymbol{I} + \varsigma \boldsymbol{z}_t \frac{\beta}{\alpha} \boldsymbol{z}_t^T\right) \\
&= \sqrt{\alpha}\boldsymbol{A}_t + \frac{\beta}{\sqrt{\alpha}}\varsigma \boldsymbol{A}_t \boldsymbol{z}_t \boldsymbol{z}_t^T \\
&= \sqrt{\alpha}\boldsymbol{A}_t + \frac{\sqrt{\alpha}}{\|\boldsymbol{z}_t\|^2}\left(\sqrt{1+\frac{\beta}{\alpha}\|\boldsymbol{z}_t\|^2} - 1\right)[\boldsymbol{A}_t \boldsymbol{z}_t]\,\boldsymbol{z}_t^T
\end{aligned}$$

computes the update of the Cholesky factor. □

The square brackets in the last equation indicate the order of computation, showing how to achieve a time complexity of $O(n^2)$. For an analogous rank-μ update (e.g., see [7, 6]) the time complexity can be reduced accordingly to $O(\mu n^2)$.

The new update rule ensures a positive-definite covariance matrix. In general, the numerical stability of the new update is likely to be better than an update requiring decompositions (e.g., see the discussion in [4, chapter 6]).

4.2 (1+1)-CMA-ES with Cholesky Update

Now we combine the (1+1)-CMA-ES with the proposed "Cholesky update". The new algorithm efficiently implements the (1+1)-CMA-ES for $c_c = 1$. That is, the evolution path is not considered, because it cannot be written in the form $\boldsymbol{p} = \boldsymbol{A}\boldsymbol{z}$ for equation (2) easily.

In contrast to the (1+1)-CMA-ES from Section 3, the computational complexity of a single generation is reduced from $O(n^3)$ to $O(n^2)$. A further advantage of the (1+1)-CMA-ES with Cholesky update, termed (1+1)-Cholesky-CMA-ES in the remainder of this article, is its simple implementation. It can be briefly described without hidden procedures such as the covariance matrix decomposition, which is necessary in the standard (1+1)-CMA-ES for sampling a multivariate normal distribution.

We describe the (1+1)-Cholesky-CMA-ES in three routines, similar to the description of the (1+1)-CMA-ES in the preceding section.

Algorithm 2: (1+1)-Cholesky-CMA-ES

```
initialize x_parent^(g), σ, A, p̄_succ ← p_succ^target
repeat
    z ~ N(0, I)
    x_offspring ← x_parent + σ A z
    updateStepSize(σ, λ_succ, p̄_succ)
    if f(x_offspring) ≤ f(x_parent) then
        x_parent ← x_offspring
        updateCholesky(A, z, p̄_succ)
until stopping criterion is met
```

The update of the step size is done as in the standard (1+1)-CMA-ES, and the same procedure `updateStepSize` is used.

In the (1+1)-Cholesky-CMA-ES the covariance matrix is never explicitly calculated. Instead, its Cholesky factors are directly updated using Theorem 1. This, together with the fact that the evolution path is not considered, leads to a very short and elegant formulation of the covariance update `updateCholesky`.

Procedure `updateCholesky`$(\boldsymbol{A}, \boldsymbol{z}, \overline{p}_{\text{succ}})$

```
if p̄_succ < p_thresh then
    A ← c_a A + (c_a / ||z||^2) ( sqrt(1 + (1 - c_a^2)||z||^2 / c_a^2) - 1 ) A z z^T
```

The constant c_a is defined by $c_a = \sqrt{1 - c_{\text{cov}}}$. The new algorithm inherits all invariance properties from the CMA-ES (e.g., invariance under strictly increasing transformations of the fitness function, under translation and rotation of the search space). The parameters should be chosen as in the standard (1+1)-CMA-ES (see Table 1).

5. EXPERIMENTS

We conducted two sets of experiments to evaluate the new algorithms empirically. The goal of the first experiments was to compare the performance of the (1+1)-CMA-ES with the $(1, \lambda)$-CMA-ES and the default $(\mu/\mu_{\text{W}},\lambda)$-CMA-ES. The second set of experiments was done to investigate the effect of not using the evolution path.

5.1 Experimental Setup

The performance w.r.t. the number of fitness function evaluations was tested on a family of quadratic functions

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} (a_i y_i)^2 \ \text{ with } \boldsymbol{y} = \boldsymbol{O}\boldsymbol{x} \ ,$$

where $\boldsymbol{O} \in \mathbb{R}^{n\times n}$ is an arbitrary rotation matrix, i.e. $\boldsymbol{O}$ is orthogonal. We considered the three functions f_{sphere}, where $a_i = 1$ for all $i = 1, \ldots, n$, f_{elli}, where $a_i = 1000^{\frac{i-1}{n-1}}$, and f_{tablet}, where $a_i = 1000$ for $i = 1$, and $a_i = 1$ otherwise. The tested dimensions were $n = 5, 10, 20, 30, \ldots, 80$.

Further, we looked at multimodal functions, namely the

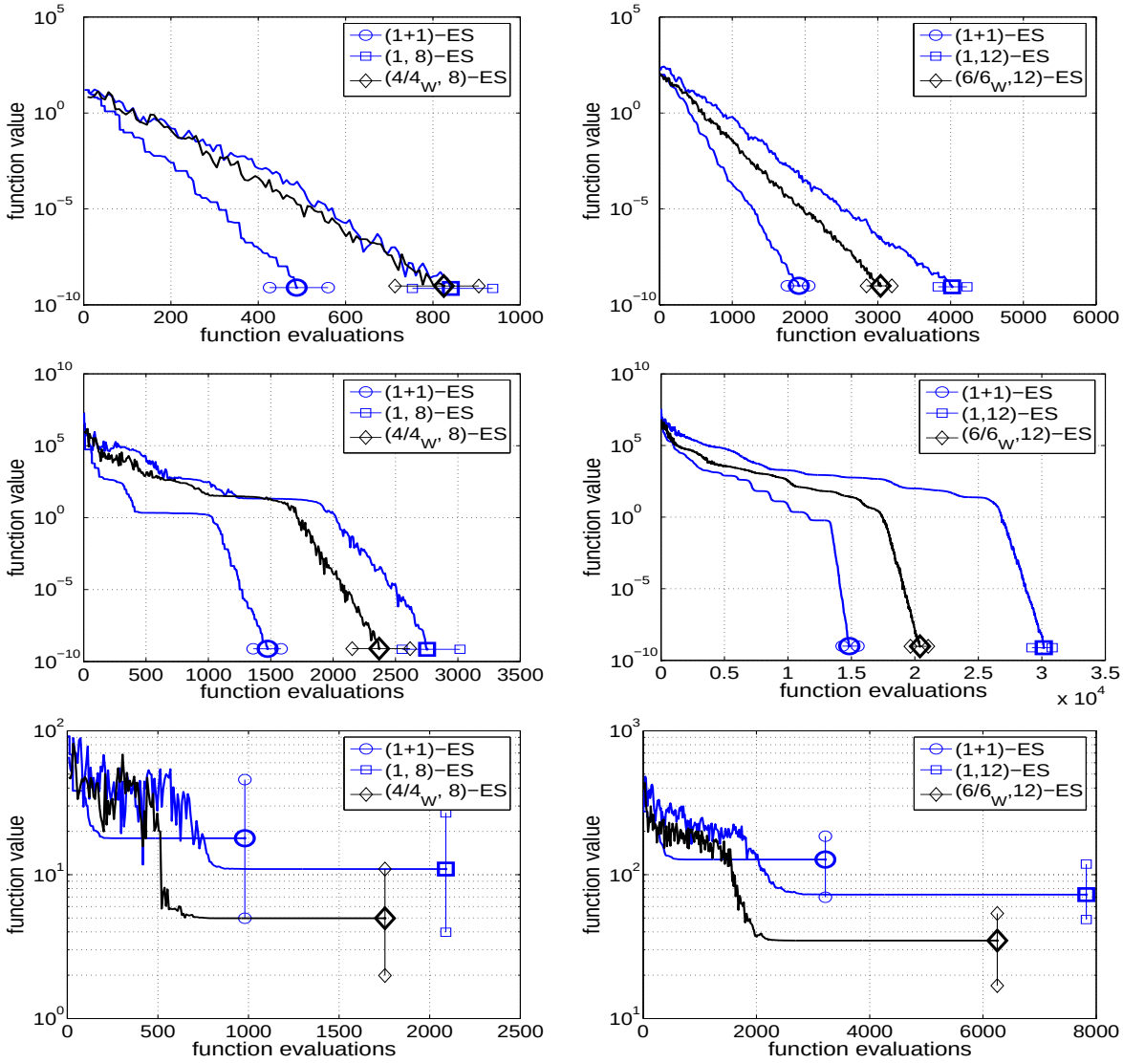


Figure 1: Simulations on functions f_{sphere} (above), f_{elli} (middle), and $f_{\text{Rastrigin}}$ (below), in 5D ($n = 5$, left) and 20D ($n = 20$, right). The plots show the median trial out of 51 trials for the (1+1)-, (1,λ)-, and (μ/μ_{W},λ)-CMA-ES. The error bars denote final values for the 3rd and the 49th trial (5%- and 95%-percentile).

generalized Ackley's function [1, 2]

$$f_{\text{Ackley}}(\boldsymbol{x}) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + \exp(1) \ ,$$

the generalized Rastrigin's function [15, 11]

$$f_{\text{Rastrigin}}(\boldsymbol{x}) = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi y_i)\right) \ ,$$

and Griewangk's function [15]

$$f_{\text{Griewangk}}(\boldsymbol{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) \ ,$$

with dimensions $n = 5, 20$.

We conducted 51 trials of each algorithm for each function and each dimension. Using 51 trials makes the report of median and 5%-quantile straightforward and more useful. In each of 51 scenarios, the initial candidate solution was chosen uniformly at random in the initial region $[-32.768, 32.768]$, $[-1, 5]$, $[-10, 600]$, and $\boldsymbol{O}^{-1}[-1, 5]^n$ for Ackley's function, Rastrigin's function, Griewangk's function, and the quadratic functions, respectively. In case of the quadratic fitness functions a new coordinate system $\boldsymbol{O}$ was randomly generated for each scenario. The initial global step size was set to 30, 3, 305, and 3 for f_{Ackley}, $f_{\text{Rastrigin}}$, $f_{\text{Griewangk}}$, and the quadratic functions, respectively. All results in this paper are invariant to the choice of $\boldsymbol{O}$.

5.2 Results

We compared the (1+1)-CMA-ES with the $(1, \lambda)$-CMA-ES and the default (μ/μ_{W},λ)-CMA-ES with weighted global

intermediate recombination and rank-μ-update of the covariance matrix. The former is elitist and has a success rule based step size adaptation. The comma strategies are non-elitist, use the cumulative step size adaptation (path length control), and the (μ/μ_{W},λ)-ES conducts weighted recombination of all $\mu = \lfloor \lambda/2 \rfloor$ parents.

First, to validate the step size adaptation we conducted experiments on a linear fitness function (the presented results are identical for any non-constant linear function). The step size increases linearly on the log-scale in all strategy variants, a minimal necessary demand on step size control [5]. The mean number of function evaluations needed to increase the step size by one order of magnitude is shown in Table 2 for the plus- and two comma-strategies. The success

Table 2: Mean number of function evaluations needed to increase the step size by a factor of ten on a linear fitness function, divided by $n/5$.

n	λ	(1+1)	(1,λ)	(μ/μ_{W},λ)
5	8	25	98	72
20	12	18	96	56

rule in the plus-strategy is up to five times faster than the path length control in the comma-strategies. This difference is an advantage in the linear environment, but should be usually of minor relevance.

Trials on f_{sphere}, f_{elli}, and $f_{\mathrm{Rastrigin}}$ are shown in Figure 1. On f_{sphere} and f_{elli} elitist (1+1) is roughly 1.5 times faster than (μ/μ_{W},λ) and two times faster than (1,λ). The final steepness of the function value graph on f_{elli} implies that the adaptation of the covariance matrix is successfully achieved as in the comma variants: after the mutation distribution has been adapted, the same performance as on the sphere function is achieved. On $f_{\mathrm{Rastrigin}}$ the standard (μ/μ_{W},λ) finds considerably (and significantly, Wilcoxon rank sum test, $p < 0.01$) better solutions than (1,λ), which in turn is significantly better than (1+1). Here, the performance of the plus-strategy can be considerably improved if the step size change rate is slowed down by increasing the damping d, but the performance of the (μ/μ_{W},λ) cannot be achieved.

In the second set of experiments, the (1+1)-Cholesky-CMA-ES, which requires $O(n)$ operations less per fitness function evaluation, is compared with the (1+1)-CMA-ES. On f_{sphere} it turns out that the difference in terms of fitness function values between the strategies is marginal, while the (1+1)-CMA-ES performs slightly faster (not shown). Figure 2 shows that the (1+1)-Cholesky-CMA-ES successfully adapts its covariance matrix to the underlying coordinate system on f_{elli}. As for the (1+1)-CMA-ES after the mutation distribution has been adapted, the same performance as on the sphere function is achieved. However, the comparison with the standard (1+1)-CMA-ES shows a considerable increase in the number of function evaluations needed. The standard (1+1)-CMA-ES performs better, because keeping track of the evolution path excels the adaptation on f_{elli}.

To quantify the performance loss by omitting the evolution path on f_{elli}, we measured the number of function evaluations to achieve a fitness value less than 10^{-10}. For every dimension, we determined the increase in the number of fitness evaluations by dividing the median of the number

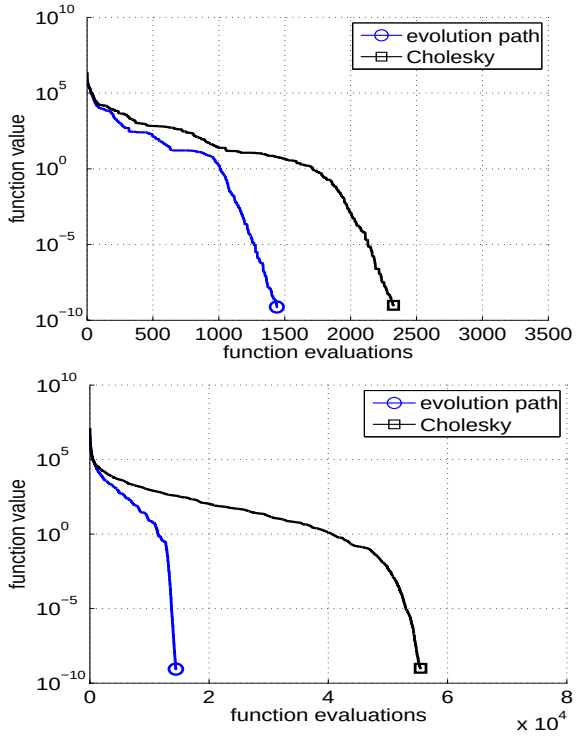


Figure 2: Course of the median fitness computed in each iteration over 51 trials of the (1+1)-CMA-ES with evolution path and the (1+1)-Cholesky-CMA-ES without evolution path on f_{elli} for dimensions $n = 5, 20$. The advantage of the evolution path becomes more pronounced with increasing dimension.

of evaluations to reach the threshold 10^{-10} using the (1+1)-Cholesky-CMA-ES by the corresponding median achieved by the (1+1)-CMA-ES. This ratio is plotted in Figure 3. The loss is less than linear in the dimensionality n, while we gain a factor of n in terms of computational efficiency by using the Cholesky update. Thus, in terms of computational efficiency of the covariance matrix update, the larger number of matrix updates due to omitting the evolution path is still compensated by the new update rule for large n. Of course, more generations mean more fitness evaluations. Thus, the (1+1)-CMA-ES with evolution path performs better than the (1+1)-Cholesky-CMA-ES in terms of objective function evaluations, which is usually the most relevant measure in practice.

Figure 4 shows trials on f_{tablet}. While for $n = 5$ the result is slightly in favour for the standard (1+1)-CMA-ES, for $n = 20$ the Cholesky variant slightly outperforms the standard variant in a later stage of the optimization. Obviously, on f_{tablet} the evolution path is of lesser importance.

On the multimodal test functions, no considerable or statistically significant performance differences can be found between the two algorithms, see Tables 3, 4 and Figure 5.

6. DISCUSSION

The proposed (1+1)-CMA-ES combines the covariance matrix adaptation and a newly introduced success rule based step size control with the (1+1) selection scheme. Step size

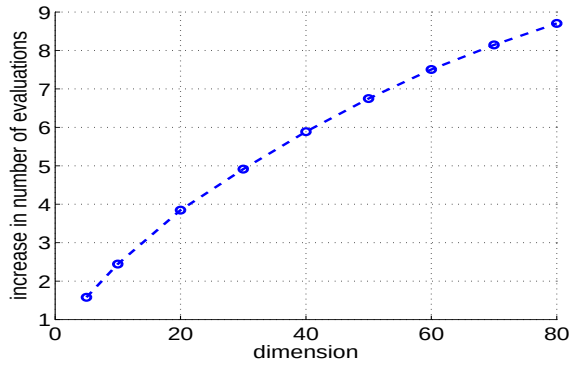


Figure 3: Number of function evaluations needed to reduce the fitness below 10^{-10} on f_{elli}. We computed for each iteration the median over 51 trials of the (1+1)-CMA-ES with evolution path and the (1+1)-Cholesky-CMA-ES without evolution path for varying problem dimension n. The plot depicts the quotient, that is how many times the (1+1)-CMA-ES is faster. The decreasing slope of the graph indicates that the gain in performance is less than linear.

Table 3: Comparison of the (1+1)-CMA-ES with evolution path and the (1+1)-Cholesky-CMA-ES without evolution path on Rastrigin's function $f_{\text{Rastrigin}}$. The table shows the values for median, the 3rd best, and the 49th best trial (i.e., median and 5%- and 95%-percentile) in generation 1000 for a total number of 51 trials.

n	with evolution path			without evolution path		
	median	3rd	49th	median	3rd	49th
5	11.4	11.4	29.5	11.4	11.4	29.5
20	63.7	45.6	100	63.8	45.6	118.1

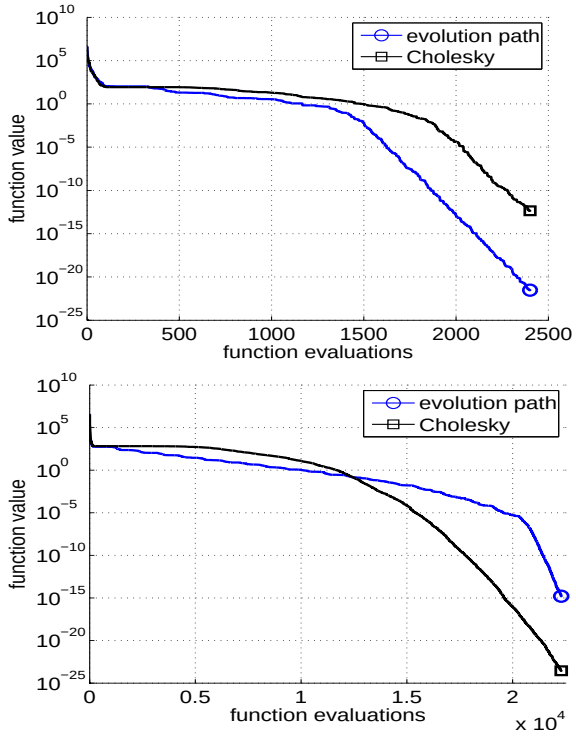


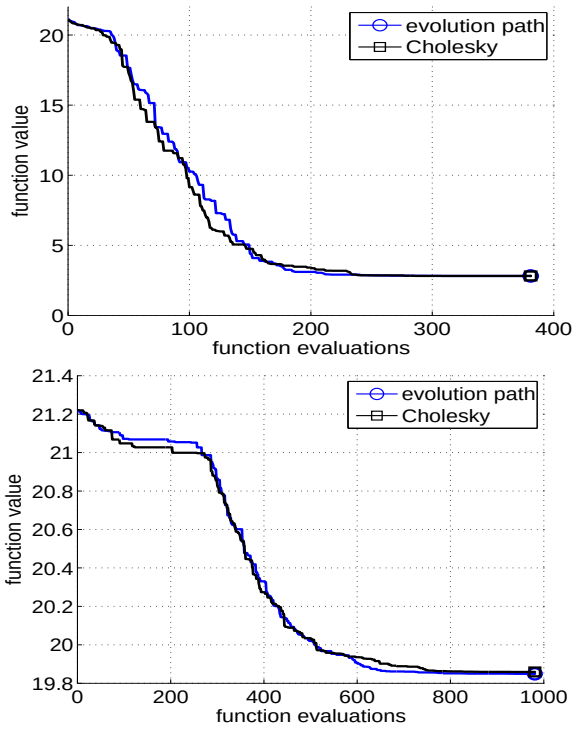
Figure 4: Course of the median fitness computed in each iteration over 51 trials of the (1+1)-CMA-ES with evolution path (thinner line) and the (1+1)-Cholesky-CMA-ES without evolution path on function f_{tablet}. The upper plot refers to $n = 5$, the lower to $n = 20$.

control and covariance matrix adaptation perform reliably within the (1+1)-ES. The result reveals that complex strategy parameter adaptation in evolution strategies is not confined to non-elitist selection. On unimodal functions the plus-selection together with the success rule based adaptation for the step size makes the (1+1)-CMA-ES about 1.5 times faster than the default $(\mu/\mu, \lambda)$-CMA-ES. In contrast, on multimodal functions the comma strategy with path length control is less susceptible to get trapped into local optima for two reasons. First, even a locally well evaluated individual is abandoned in the next generation; second, the path length control adapts larger step lengths, in particular within the recombinant strategy variant. Due to its performance deficiency on multimodal functions we would not generally recommend the (1+1)-CMA-ES as new default strategy variant.

The newly developed covariance matrix update rule reduces the computational complexity of the rank-one covariance matrix update from $O(n^3)$ to $O(n^2)$. This is a significant improvement on high dimensional, fast computable fitness functions. The new update rule is much simpler to implement (e.g., allowing for easy implementations in hardware and in low level programming languages) and is completely specified without any hidden, numerically involved procedures such as a singular value decomposition.

In contrast, using the more expensive eigenvalue or singular value decomposition has the advantage of making the eigenvalues of the covariance matrix available. This provides insights into the problem, allows for cumulative step size adaptation [8, 9], and makes explicit bounding of eigenvalues from below possible. In practice it is even not necessary to perform the eigenvalue decomposition in every generation, but only every τ generations. For $\tau = o(n)$ the new approach is still faster for large n while $\tau = \omega(n)$ is not advisable. Only for $\tau = \Theta(n)$ the computational complexity aligns with the Cholesky update rule. We believe that even then, aside from its simplicity, the new rule is still computationally faster, which will be asserted in further investigations.

The most severe drawback of the proposed covariance matrix update is its incompatibility with the concept of an evolution path, because the evolution path cannot be factorized appropriately. On some benchmark functions the evolution path is of minor relevance, but on some test functions it considerably improves the performance, and it rarely results in a usually small performance degradation. We cannot rule out that, based on the given results, an efficient factorization with evolution path might be found in future. Nevertheless, for the time being, we consider the (1+1)-Cholesky-CMA-



n	with evolution path			without evolution path		
	median	3rd	49th	median	3rd	49th
5	2.81	1.90 $\cdot 10^{-10}$	20.00	2.81	2.64 $\cdot 10^{-10}$	20.00
10	19.85	3.50	20.00	19.86	2.57	20.00

Figure 5: The plots show the median fitness computed in each iteration over 51 trials of the (1+1)-CMA-ES with evolution path and the (1+1)-Cholesky-CMA-ES without evolution path on Ackley's function f_{Ackley}. The upper plot refers to $n = 5$, the lower to $n = 20$. The graphs for the two algorithms can hardly be distinguished. The table shows the values for median, the 3rd best, and the 49th best trial in generation 1000.

ES as a theoretically elegant algorithm, the perhaps simplest evolution strategy with step size and covariance matrix adaptation, applicable to fast computable fitness functions.

7. REFERENCES

[1] D. H. Ackley. *A connectionist machine for genetic hillclimbing*. Kluwer, Boston, 1987.

[2] T. Bäck. *Evolutionary algorithms in theory and practice*. Oxford University Press, 1996.

[3] H.-G. Beyer and H.-P. Schwefel. Evolution strategies: A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[4] M. S. Grewal and A. P. Andrews. *Kalman Filtering: Theory and Practice*. Prentice-Hall, 1993.

[5] N. Hansen. An analysis of mutative σ-self-adaptation on linear fitness functions. *Evolutionary Computation*, 14(3), 2006.

Table 4: Comparison of the (1+1)-CMA-ES with evolution path and the (1+1)-Cholesky-CMA-ES without evolution path on Griewangk's function $f_{\text{Griewangk}}$. The table shows the values for median, the 3rd best, and the 49th best trial after 1000 and 2000 generation for a total number of 51 trials.

n	with evolution path			without evolution path		
	median	3rd	49th	median	3rd	49th
			after 1000 evaluations			
5	0.34	0.049	1.10	0.27	0.071	1.02
10	0.23	0.090	0.63	0.33	0.081	0.67
			after 2000 evaluations			
10	0.017	2.90 $\cdot 10^{-6}$	0.050	0.012	3.00 $\cdot 10^{-6}$	0.042

[6] N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao, E. Burke, J. A. Lozano, J. Smith, J. J. Merelo-Guervós, J. A. Bullinaria, J. Rowe, P. Tiňo, A. Kabán, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature (PPSN VIII)*, volume 3242 of *LNCS*, pages 282–291. Springer-Verlag, 2004.

[7] 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.

[8] 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)*, pages 312–317, 1996.

[9] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[10] S. Kern, S. 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.

[11] H. Mühlenbein, D. Schomisch, and J. Born. The Parallel Genetic Algorithm as Function Optimizer. *Parallel Computing*, 17(6-7):619–632, 1991.

[12] I. Rechenberg. *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Frommann-Holzboog, 1973.

[13] G. Rudolph. On correlated mutations in evolution strategies. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature 2 (PPSN II)*, pages 105–114. Elsevier, 1992.

[14] H.-P. Schwefel. *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology Series. John Wiley & Sons, 1995.

[15] A. Törn and A. Zilinskas. *Global Optimization*, volume 350 of *LNCS*. Springer-Verlag, 1989.