A CMA-ES with Multiplicative Covariance Matrix Updates

Oswin Krause
Department of Computer Science
University of Copenhagen
Copenhagen,Denmark
oswin.krause@di.ku.dk

Tobias Glasmachers
Institut für Neuroinformatik
Ruhr-Universität Bochum
Bochum, Germany
tobias.glasmachers@ini.rub.de

ABSTRACT

Covariance matrix adaptation (CMA) mechanisms are core building blocks of modern evolution strategies. Despite sharing a common principle, the exact implementation of CMA varies considerably between different algorithms. In this paper, we investigate the benefits of an exponential parametrization of the covariance matrix in the CMA-ES. This technique was first proposed for the xNES algorithm. It results in a multiplicative update formula for the covariance matrix. We show that the exponential parameterization and the multiplicative update are compatible with all mechanisms of CMA-ES. The resulting algorithm, xCMA-ES, performs at least on par with plain CMA-ES. Its advantages show in particular with updates that actively decrease the sampling variance in specific directions, i.e., for active constraint handling.

Categories and Subject Descriptors

[**Continuous Optimization**]

General Terms

Algorithms

Keywords

evolution strategies, covariance matrix adaptation, CMA-ES, multiplicative update, exponential coordinates

1. INTRODUCTION

Evolution Strategies (ES) are randomized direct search algorithms suitable for solving black box problems in the continuous domain, i.e., minimization problems $f : \mathbb{R}^d \to \mathbb{R}$ defined on a d-dimensional real vector space. Most of these algorithms generate a number of normally distributed offspring in each generation. The efficiency of this scheme, at least for unimodal problems, crucially depends on online adaptation of parameters of the Gaussian search distribution $\mathcal{N}(m, \sigma^2 C)$, namely the global step size σ and the covariance matrix C. Adaptation of the step size enables linear convergence on scale invariant functions [4], while covariance matrix adaptation (CMA) [10] renders the asymptotic convergence rate independent of the conditioning number of the Hessian in the optimum of a twice continuously differentiable function.

The most prominent algorithm implementing the above principles is CMA-ES [10, 8, 12]. Nowadays there exists a plethora of variants and extensions of the basic algorithm.

A generic principle for the online update of parameters (including the search distribution) is to maximize the expected progress. This goal can be approximated by adapting the search distribution so that the probability of the perturbations that generated successful offspring in the past are increased. This is likely to foster the generation of better points in the near future.[1]

The application of the above principle to CMA means to change the covariance matrix towards the maximum likelihood estimator of successful steps. To this end let $\mathcal{N}(m, C)$ denote the search distribution, and let $x_1, \dots, x_\mu$ denote successful offspring. The maximum likelihood estimator of the covariance matrix generating step $\delta_i = x_i - m$ is the rank-one matrix $\delta_i \delta_i^T$. All CMA updates of CMA-ES are of the generic form $C \leftarrow (1-c) \cdot C + c \cdot \delta\delta^T$, which is a realization of this technique keeping an exponentially fading record of previous successful steps δ. CMA-ES variants differ in which step vectors enter the covariance matrix update. Early variants were based on cumulation of directions in an evolution path vector p_c and a single rank-one update of C per generation [10]. Later versions added a rank-μ update based on immediate information from the current population [8].

A different perspective on CMA techniques is provided within the framework of information geometric optimization (IGO) [14], in particular by the natural evolution strategy (NES) approach [17, 16]. It turns out that the rank-μ update equation can be derived from stochastic natural gradient descent of a stochastically relaxed problem on the statistical manifold of search distributions. This more general perspective opens up new possibilities for CMA mechanisms, e.g., reparameterizing the covariance matrix in exponential form as done in the xNES algorithm [7]. This results in an update equation with the following properties: a) the update is *multiplicative*, in contrast to the standard additive update, b) it is possible to leave the variance in directions orthogonal to

GECCO'15, July 11-15, 2015, Madrid, Spain.

[1] This statement holds only under (mild) assumptions on the regularity of the fitness landscape, which remain implicit, but may be violated, e.g., in the presence of constraints.

all observed steps unchanged, and c) even when performing "active" (negative) updates the covariance matrix is guaranteed to remain positive definite.

In this paper we incorporate the exponential parameterization of the covariance matrix into CMA-ES. We derive all mechanisms found in the standard CMA-ES algorithm in this framework, demonstrating the compatibility of cumulative step size adaptation and evolution paths (two features missing in xNES) with exponential coordinates. The new algorithm is called xCMA-ES. Its performance on standard (unimodal) benchmarks coincides with that of CMA-ES, however, in addition it benefits from neat properties of the exponential parameters, which show up prominently when performing active CMA updates with negative weights, e.g., for constraint handling.

In the next section we recap CMA-ES and xNES. Based thereon we present our new xCMA-ES algorithm. In section 3 its performance is evaluated empirically on standard benchmarks. We demonstrate the superiority for special tasks involving active CMA updates.

Algorithm 1: CMA-ES

```
Input: $m$, $\sigma$
$C \leftarrow \mathrm{I}$
while stopping condition not met do
    // sample and evaluate offspring
    for $i \in \{1, \dots, \lambda\}$ do
        $x_i \leftarrow \mathcal{N}(m, \sigma^2 C)$
    end
    sort $\{x_i\}$ with respect to $f(x_i)$
    // internal update (paths and parameters)
    $m' \leftarrow \sum_{i=1}^{\mu} w_i x_i$
    $p_s \leftarrow (1 - c_s) \cdot p_s + \sqrt{c_s(2 - c_s)\mu_{\text{eff}}} \cdot C^{-1/2}(m' - m)$
    $p_c \leftarrow (1 - c_c) \cdot p_c + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}} \cdot \frac{m' - m}{\sigma}$
    $C \leftarrow (1 - c_1 - c_\mu) \cdot C + c_1 \cdot p_c p_c^T$
          $+ c_\mu \cdot \sum_{i=1}^{\mu} w_i \left(\frac{x_i - m}{\sigma}\right) \left(\frac{x_i - m}{\sigma}\right)^T$
    $\sigma \leftarrow \sigma \cdot \exp\left(\frac{c_s}{D_\sigma} \cdot \left(\frac{\|p_s\|}{\chi_d} - 1\right)\right)$
    $m \leftarrow m'$
end
```

2. ALGORITHMS

In this section we provide the necessary background for our new algorithm before introducing the xCMA-ES. We will cover the well-known CMA-ES algorithm as well as xNES, both with a focus on the components required for our own contribution. In the following algorithms, a few implementation details are not shown, e.g., the eigen decomposition of C required for sampling as well as for the computation of the inverse matrix square root $C^{-1/2}$.

2.1 CMA-ES

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [10, 12] is the most prominent evolution strategy in existence. It comes in many variants, e.g., with extensions for handling of fitness noise [9] and multi-modality [5]. Here we describe what can be considered a baseline version, featuring non-elitist (μ, λ) selection, cumulative step size control (CSA), and two different types of covariance matrix updates, namely a rank-1 update based on an evolution path, and the so-called rank-μ update based on the survivors of environmental truncation selection.

The state of CMA-ES is given by the parameters $m \in \mathbb{R}^d$, $\sigma > 0$, and $C \in \mathbb{R}^{d \times d}$ of its multi-variate normal search distribution $\mathcal{N}(m, \sigma^2 C)$, as well as by the two evolution paths $p_s, p_c \in \mathbb{R}^d$. Pseudo-code of a basic CMA-ES is provided in algorithm 1. The algorithm has a number of tuning constants, e.g., the sizes of parent and offspring population μ and λ, the various leaning rates, and the rank-based weights $w_1, \dots, w_\mu$. For robust default settings for the different parameters we refer to [12].

In each generation, CMA-ES executes the following steps:

1. Sample offspring $x_1, \dots, x_\lambda \sim \mathcal{N}(m, \sigma^2 C)$. This step is realized by sampling standard normally distributed vectors $z_1, \dots, z_\lambda \in \mathbb{R}^d$, which are then transformed via $x_i \leftarrow m + \sigma A z_i$, where A is a factor of the covariance matrix fulfilling $AA^T = C$. It can be computed via a Cholesky decomposition of C, however, the usual method is eigen decomposition since this operation is needed anyway later on.
2. Evaluate the offspring's fitness values $f(x_1), \dots, f(x_\lambda)$. This function call is often considered a black box, and it is assumed that its computational cost is substantial, usually exceeding the internal computational complexity of the algorithm.
3. Sort the offspring by fitness. Post-condition: $f(x_1) \leq f(x_2) \leq \cdots \leq f(x_\lambda)$.
4. Perform environmental selection: keep $\{x_1, \dots, x_\mu\}$, discard $\{x_{\mu+1}, \dots, x_\lambda\}$, usually the worse half.
5. Update the evolution path for cumulative step size adaptation: the path p_s is an exponentially fading record of steps of the mean vector, back-transformed by multiplication with the inverse matrix square root $C^{-1/2}$ into a coordinate system where the sampling distribution is a standard normal distribution. This path is supposed to follow a standard normal distribution.
6. Update the evolution path for covariance matrix adaptation: the path p_c is an exponentially fading record of steps of the mean vector divided (corrected) by the step size. This path models the movement direction of the distribution center over time.
7. Update the covariance matrix: a new matrix is obtained by additive blending of the old matrix with a rank-one matrix formed by the path p_c and a rank-μ matrix formed by successful steps of the current population.
8. Update the step size: the step size is changed if the norm of p_s indicates a systematic deviation from the standard normal distribution. Note (for later reference) that this update has a multiplicative form, involving the exponential function.
9. Update the mean: discard the old mean and center the distribution on a weighted mean of the new parent population.

2.2 xNES

The exponential natural evolution strategy (xNES) algorithm [7] is a prominent member of the family of natural evolution strategies (NES) [17, 16]. While exhibiting all properties of an evolution strategy, it is derived as a stochastic natural gradient descent method on the statistical manifold of search distributions, an approach that is best understood

Algorithm 2: xNES

```
Input: $m$, $\sigma$
$B \leftarrow \mathrm{I}$
while stopping condition not met do
    // sample and evaluate individuals
    for $i \in \{1, \dots, \lambda\}$ do
        $z_i \leftarrow \mathcal{N}(0, \mathrm{I})$
        $x_i \leftarrow m + \sigma \cdot B z_i$
    end
    sort $\{(z_i, x_i)\}$ with respect to $f(x_i)$
    // internal update (SNGD step)
    $G_\delta \leftarrow \sum_{i=1}^{\lambda} u_i \cdot z_i$
    $G_M \leftarrow \sum_{i=1}^{\lambda} u_i \cdot (z_i z_i^T - \mathrm{I})$
    $G_\sigma \leftarrow \mathrm{tr}(G_M)/d$
    $G_B \leftarrow G_M - G_\sigma \cdot \mathrm{I}$
    $m \leftarrow m + \eta_m \cdot \sigma B \cdot G_\delta$
    $\sigma \leftarrow \sigma \cdot \exp(\eta_\sigma/2 \cdot G_\sigma)$
    $B \leftarrow B \cdot \exp(\eta_B/2 \cdot G_B)$
end
```

as an instance of information geometric optimization [14]. NES was found to have close relations to CMA-ES [1, 7].

The NES family of algorithms is derived as follow. Let $\{P_\theta \,|\, \theta \in \Theta\}$ denote a family of search distributions with parameters θ, where Θ is a differentiable manifold. The most prominent example are multivariate Gaussian distributions $P_\theta = \mathcal{N}(m, C)$ with parameters $\theta = (m, C)$ and density $p_\theta(x)$. The algorithm aims to solve the optimization problem

$$\min_\theta F(\theta) = \mathbb{E}_{x \sim P_\theta}\big[f(x)\big]$$

which is lifted from points x to distributions P_θ. The gradient of $\nabla_\theta F(\theta)$ is $\int f(x) \nabla \log(p_\theta(x)) \mathrm{d}x$ (given that the integral converges in an open set around θ). It is intractable in the black box model, however, it can be approximated by the Monte Carlo estimator

$$G(\theta) = \frac{1}{\lambda} \sum_{i=1}^{\lambda} f(x_i) \nabla \log(p_\theta(x_i))$$

with samples $x_i \sim P_\theta$. These samples correspond to the offspring population of an ES. The update $\theta \leftarrow \theta - \gamma \cdot G(\theta)$ (with learning rate $\gamma > 0$) amounts to minimization of F with stochastic gradient descent (SGD).

This update is unsatisfactory in the context of NES since it depends on the chosen parameterization $\theta \mapsto P_\theta$ (see [14], and refer to [16, 6] for further shortcomings). In the case of optimizing P_θ, the question which direction to follow has a canonical answer. This is because $\{P_\theta \,|\, \theta \in \Theta\}$ is a statistical manifold (a manifold the points of which are distributions) with an intrinsic Riemannian information geometry induced by KL-divergence [17, 16, 14]. The gradient w.r.t. the intrinsic geometry, pulled back to the parameter space Θ, is known as the *natural gradient*, usually denoted by $\tilde{\nabla}_\theta F(\theta)$. It is obtained from the plain gradient as

$$\tilde{\nabla}_\theta F(\theta) = \mathcal{I}(\theta)^{-1} \cdot \nabla_\theta F(\theta) \ ,$$

since the Fisher information matrix $\mathcal{I}(\theta)$ is the metric tensor describing the intrinsic geometry. It is estimated in a straightforward manner by $\mathcal{I}(\theta)^{-1} G(\theta)$. The update

$$\theta \to \theta - \gamma \cdot \mathcal{I}(\theta)^{-1} G(\theta)$$

is known as stochastic natural gradient descent (SNGD).

Several different implementations of this general scheme have been developed with a focus on computational aspects [15]. It is common to replace "raw" fitness values $f(x_i)$ with rank-based utility values u_i that turn out to correspond exactly to the weights w_i of CMA-ES.

The xNES algorithm supersedes earlier developments with two novel approaches. The first of these is to perform the NES update in a *local* parameterisation $\theta \mapsto P_\theta$ for which the Fisher matrix is the identity matrix, which saves its computation or estimation and in particular its (numerical) inversion. The second technique is a parameterization of the positive definite covariance matrix involving the matrix exponential, which allows for an unconstrained representation of the covariance matrix.

Covariance matrices are symmetric and positive definite $d \times d$ matrices. Symmetric matrices form the $\frac{d(d+1)}{2}$ dimensional vector space $\mathcal{S}_d$. The requirement of positive definiteness adds a non-linear constraint. Let $\mathcal{P}_d$ denote the open sub-manifold of positive definite symmetric matrices. Then the parameter space takes the form $(m, C) = \theta \in \Theta = \mathbb{R}^d \times \mathcal{P}_d$.

Note that this space is not closed under subtraction, and also not under addition of terms from $\mathcal{S}_d$ (e.g., (natural) gradients). When performing an additive update of the covariance matrix of the form $C \leftarrow C - \gamma \cdot \Delta$, e.g., an SGD step, then a large enough step $\gamma \cdot \Delta$ can result in a violation of the positivity constraint.[2]

Possible workarounds are line search for a feasible step length or more elaborate constraint handling techniques. A conceptually easier and more elegant solution is to parameterize the manifold $\mathcal{P}_d$ with a vector space and perform SGD on this new parameter space. This is exactly the role of the matrix exponential $\exp : \mathcal{S}_d \to \mathcal{P}_d$, which is a diffeomorphism (a bijective, smooth mapping with smooth inverse).

The exponential map for matrices is in general defined in terms of the power series expansion $\exp(M) = \sum_{n=0}^{\infty} \frac{1}{n!} M^n$, mapping general $d \times d$ matrices to the general linear group of invertible $d \times d$ matrices. For symmetric matrices it can be understood in terms of a spectral transformation. Let $M = UDU^T$ with U orthogonal and D diagonal denote the eigen decomposition of M, then it is easy to see from the power series formula that it holds $\exp(M) = U \exp(D) U^T$. The exponential of a diagonal matrix is simply the diagonal matrix consisting of the scalar exponentials of the diagonal entries. Hence the matrix exponential corresponds to exponentiation of the eigen values, mapping general to positive eigen values and thus $\mathcal{S}_d$ to $\mathcal{P}_d$.

The xNES algorithm uses a special coordinate system for θ, centered into the current search distribution. Let (m, C) denote the parameters of the current search distribution, and let A denote a factor of the covariance matrix C, fulfilling $AA^T = C$. We introduce local coordinates $(\delta, M) = \theta \in \Theta = \mathbb{R}^d \times \mathcal{S}_d$ (hence now Θ forms a vector space, which is in particular closed under addition) so that $(\delta, M) = (0, 0)$ represents the current distribution, and new

[2] This problem does not appear with standard CMA-ES updates where positive semi-definite matrices are added to a positive definite matrix, the result of which is always positive definite.

distribution parameters (m', A') are represented as

$$(\delta,\, M) \mapsto \big(m',\, A'\big) = \left(m + A\delta,\, A \exp\left(\frac{1}{2}M\right)\right) \ .$$

The coordinates are chosen so that the Fisher matrix is the identity: $\mathcal{I}(0,0) = \mathrm{I}$. Hence the coordinates are orthonormal w.r.t. the intrinsic geometry. Plain and natural gradient coincide.

Despite the seemingly complicated derivation of xNES involving stochastic natural gradient on a statistical manifold and a non-linear coordinate system based on the matrix exponential, its update equations are surprisingly simple. The complete pseudo-code is given in algorithm 2.

In this implementation the covariance matrix factor A is represented as $A = \sigma B$, where the transformation matrix B fulfills $\det(B) = 1$. In the chosen exponential coordinates the determinant corresponds to the trace (see computation of G_σ and G_B in the algorithm). The parameters m, σ, and B can be updated with independent SNGD steps, potentially with different learning rates.

The parameters of the xNES algorithm are the sample (population) size λ, the learning rates η_m, η_σ, and η_B, as well as the rank-based weights $u_1, \dots, u_\lambda$. Population size and weights essentially follow the settings of CMA-ES, with the deviation to subtract the mean from the weights. This leads to the weights $u_i = w_i - 1/\lambda$, resulting in negative weights for individuals that simply don't enter the CMA-ES due to truncation selection. The mean learning rate has the canonical value $\eta_m = 1$, which results in the exact same mean update as in CMA-ES [1, 7], while the other learning rates were empirically tuned, see [7].

Note that due to the use of the matrix exponential in xNES the updates of σ and B have exactly the same form. In contrast to CMA-ES, the covariance matrix update of xNES is multiplicative in nature. We argue that conceptually this is a desirable property since σ (scale) and B (shape) both describe complementary properties of the covariance matrix $C = \sigma^2 BB^T$, and they enter the sampling process in a similar way, namely by left-multiplication with the standard normally distributed random vectors z_i. In fact, the exponential parameterization seems to be canonical since it allows for a clear separation of the scale component σ and the shape component B of the search distribution as *linear* sub-spaces of Θ, see also [7].

2.3 Efficient Multiplicative Update

While the multiplicative update rule of the xNES guarantees positive definiteness of the covariance matrix, the matrix exponential in itself is usually a computationally expensive operation. In the following we show that the update can be implemented efficiently with time complexity $\mathcal{O}(d^2\lambda)$, which coincides with the complexity of the additive update of the CMA-ES. Thus the computational difference between the updates is merely a constant.

Lemma 1. *Consider a matrix $G = \sum_{i=1}^{\lambda} u_i(z_i z_i^T - \mathrm{I})$ built from $\lambda < d$ vectors $z_i \in \mathbb{R}^d$, weights $u_i \in \mathbb{R}$, and a positive definite symmetric matrix $C \in \mathbb{R}^{d\times d}$ for which a decomposition $C = AA^T$ is available. Then the term*

$$A \exp(G) A^T$$

can be computed with time complexity $\mathcal{O}(\lambda d^2 + \lambda^2 d + \lambda^3)$.

Proof. The proof has two parts. In the case that $\sum_{i=1}^{\lambda} u_i = 0$, this reduces to $G = \sum_{i=1}^{\lambda} u_i z_i z_i^T$. We first take a look at the case that $\bar{u} = \sum_{i=1}^{\lambda} u_i \neq 0$. Then we have

$$\begin{aligned} \exp(G) &= \exp\left(-\bar{u}\mathrm{I} + \sum_{i=1}^{\lambda} u_i z_i z_i^T\right) \\ &= \exp(-\bar{u}\mathrm{I}) \exp\left(\sum_{i=1}^{\lambda} u_i z_i z_i^T\right) \\ &= \exp(-\bar{u}) \exp\left(\sum_{i=1}^{\lambda} u_i z_i z_i^T\right) \end{aligned}$$

The second step holds since $-\bar{u}\mathrm{I}$ commutes with $\sum_{i=1}^{\lambda} u_i z_i z_i^T$. We can thus assume w.l.o.g. that $G = \sum_{i=1}^{\lambda} u_i z_i z_i^T$, which includes the case $\bar{u} = 0$. Because of $\mathrm{rank}(G) \leq \lambda$ we can find an eigen decomposition of $G = QDQ^T$ with $Q \in \mathbb{R}^{d\times\lambda}$ and $D \in \mathbb{R}^{\lambda\times\lambda}$ in $\mathcal{O}(\lambda^2 d + \lambda^3)$ time[3]. With this decomposition the matrix exponential can be rewritten as

$$\begin{aligned} \exp(G) &= \left(\mathrm{I} - QQ^T\right) + Q\exp(D)Q^T \\ &= \mathrm{I} + Q\left(\exp(D) - \mathrm{I}\right)Q^T \ . \end{aligned}$$

The first equality holds because $\exp(G)$ maps all 0-eigenvalues of G to 1, which leads to the first term. Insertion of the result into the term of interest yields

$$\begin{aligned} A\exp(G)A^T &= A\left[\mathrm{I} + Q\left(\exp(D) - \mathrm{I}\right)Q^T\right]A^T \\ &= AA^T + AQ\left(\exp(D) - \mathrm{I}\right)(AQ)^T \\ &= C + AQ\left(\exp(D) - \mathrm{I}\right)(AQ)^T \end{aligned}$$

We can compute $K = AQ \in \mathbb{R}^{d\times\lambda}$ in $\mathcal{O}(\lambda d^2)$ and $C + K\left(\exp(D) - \mathrm{I}\right)K^T$ in $\mathcal{O}(\lambda d^2)$ time. □

The lemma implies that if a decomposition $C = AA^T$ is available then the update $C \leftarrow A\exp(G)A^T$, or $C \leftarrow C + K\left(\exp(D) - \mathrm{I}\right)K^T$ in the notation of the proof, can now be seen as a rank-λ update to C. It can be computed significantly faster than a full eigen decomposition of C. Typically, as $\lambda = \mathcal{O}(\log(d))$, the runtime costs are dominated by the $\mathcal{O}(\lambda d^2)$ operations of the matrix multiplications, which leads to the same runtime complexity as the additive matrix update. If A was a Cholesky-factor then it could be updated efficiently in $\mathcal{O}(d^2\lambda)$ operations as well, without requiring to store or compute C first [13]. If A has been computed through an eigenvalue decomposition then there is currently no fast algorithm known to perform the update of the eigenvalue decomposition, and recomputing it from C has time complexity $\Theta(d^3)$, dominating the overhead of the exponential update.

2.4 xCMA-ES

In this section we show that exponential coordinates for the covariance matrix are compatible with all mechanisms of CMA-ES. Consequently, we incorporate this technique into the CMA-ES algorithm, resulting in a new method called exponential CMA-ES, or *xCMA-ES* for short.

[3] This can be achieved by first applying a QR-decomposition on the matrix $Z = (z_1, \dots, z_\lambda)$ which yields a $\lambda \times d$-matrix B with $BB^T = I$ and $BGB^T = K \in \mathbb{R}^{\lambda\times\lambda}$ in $O(\lambda d^2)$. An eigenvalue decomposition of $K = VDV^T$ can then be performed in $O(\lambda^3)$ and $Q = B^T V$.

The xCMA-ES algorithm features all techniques found in CMA-ES, but in addition incorporates the multiplicative covariance matrix update of the xNES algorithm. This means that xCMA-ES is equipped with two evolution paths, one for cumulative step size control, and one for the rank-one covariance matrix update. Notably, xCMA-ES comes with explicit step size control, while in xNES the step size is updated with the same mechanism as the shape component of the covariance matrix, with the only difference that the learning rates can be decoupled. For xCMA-ES we follow the proceeding of CMA-ES and do not decouple these parameters explicitly (i.e., the scale of the covariance matrix is allowed to drift).

The beauty of this construction is that all mechanisms of standard CMA-ES are naturally compatible with the exponential parameterization of the covariance matrix. In particular, the updates of the evolution paths and the step size do not require any changes.

The stochastic natural gradient component $\sum_{i=1}^{\lambda} u_i \cdot (z_i z_i^T - \mathrm{I})$ of xNES deserves particular attention. This matrix is—up to scaling—the quantity entering the matrix exponential. It consists of a weighted sum of outer products of steps in the "natural" coordinate system of the standard normally distributed samples z_i minus their expected value, which (for standard normal samples) is the identity matrix. Note that due to $\sum_i u_i = 0$ the weighted identity matrices cancel each other out. Also note that in first order Taylor approximation around the origin the matrix exponential reduces to $\exp(M) \approx \mathrm{I} + M$, resulting in the exact rank-μ update of CMA-ES [1, 7, 14].

Hence it is natural to incorporate the rank-one update of CMA-ES into the multiplicative update by adding the term $c_1 \cdot (pp^T - \mathrm{I})$ to the term entering the matrix exponential, where $p = C^{-1/2} p_c$ is the evolution path transformed back to the coordinate system of standard normally distributed samples.

A disadvantage of the mean-free weights u_i in the xNES algorithm is that the computed steps G_δ are only mean-free with respect to the *estimated* mean $\hat{z} = \frac{1}{\lambda} \sum_{i=1}^{\lambda} z_i$ as

$$G_\delta = \sum_{i=1}^{\lambda} u_i \cdot z_i = \sum_{i=1}^{\lambda} \left(w_i - \frac{1}{\lambda} \right) \cdot z_i = \sum_{i=1}^{\lambda} w_i \cdot z_i - \hat{z} \ ,$$

thus adding additional noise to the update. By replacing $\hat{z}$ by the true (zero) mean, we achieve a better estimate—the original CMA-ES update[4].

These changes applied to CMA-ES define the basic xCMA-ES algorithm. Its pseudo-code is found in algorithm 3.

Algorithm 3: xCMA-ES

```
Input: $m$, $\sigma$
$C \leftarrow \mathrm{I}$
while stopping condition not met do
    // sample and evaluate offspring
    for $i \in \{1, \dots, \lambda\}$ do
        $z_i \leftarrow \mathcal{N}(0, \mathrm{I})$
        $x_i \leftarrow m + \sigma \cdot \sqrt{C} z_i$
    end
    sort $\{(z_i, x_i)\}$ with respect to $f(x_i)$
    // internal update (paths and parameters)
    $m' \leftarrow \sum_{i=1}^{\lambda} (u_i + \frac{1}{\lambda}) x_i$
    $p_s \leftarrow (1 - c_s) \cdot p_s + \sqrt{c_s(2 - c_s)\mu_{\text{eff}}} \cdot C^{-1/2}(m' - m)$
    $p_c \leftarrow (1 - c_c) \cdot p_c + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}} \cdot \frac{m' - m}{\sigma}$
    $Z \leftarrow c_1 \cdot \left[ C^{-1/2} p_c p_c^T C^{-1/2} - \mathrm{I} \right] + c_\mu \cdot \sum_{i=1}^{\lambda} u_i z_i z_i^T$
    $C \leftarrow C^{1/2} \exp(Z) C^{1/2}$
    $\sigma \leftarrow \sigma \cdot \exp\left( \frac{c_s}{D_\sigma} \cdot \left( \frac{\|p_s\|}{\chi_d} - 1 \right) \right)$
    $m \leftarrow m'$
end
```

As noted above, when using identical weights and learning rates, in first order approximation the multiplicative update in xCMA-ES coincides with the additive update in standard CMA-ES. Both updates rely on a rank-$(\lambda + 1)$ matrix[5] formed by the outer products of the sampling steps and the evolution path p_c. An interesting conceptual difference between additive and multiplicative update is that the additive update shrinks the variance in all directions orthogonal to the $\lambda + 1$ update vectors by a factor of $1 - c_1 - c_c$, while the multiplicative update leaves these variance components untouched. In our opinion the latter better reflects the fact that no information was observed about these directions. Also, by means of subtraction of the identity (the expected value) from the positive semi-definite weighted sum of outer products, it is natural for the multiplicative update to (multiplicatively) *grow or shrink* the variance in a specific direction. In contrast, the additive update can only (additively) *grow* the variance in a specific direction, while shrinking works only through global shrinking and subsequently growing all other directions, which can be slow.

This problem was mitigated to some extent with so-called active covariance matrix updates [2, 11]. In an active update step the algorithm subtracts an outer product from the covariance matrix, an operation that must carefully ensure the positive definiteness of the result, e.g., by means of line search or finely tuned learning rates. In contrast, the multiplicative update operates on an unconstrained problem and hence can never result in an invalid configuration.

2.5 Constraint Handling

A powerful constraint handling mechanism for the $(1, \lambda)$-CMA-ES was introduces in [3]. In principle, the same mechanism can be applied to the non-elitist, population-based standard formulation of CMA-ES. The simple yet highly efficient approach amounts to performing active CMA updates for steps resulting in infeasible offspring, effectively reducing the variance in the direction of the constraint normal, while suspending step size adaptation.

The corresponding mechanism for xCMA-ES is essentially the same, namely to perform a standard update with negative weight, but of course without a need for constraining the step size. There is a subtle difference to the elitist setting considered in [3]: with non-elitist selection it is in principle possible for infeasible offspring to enter the new parent population, namely if more than $\lambda - \mu$ offspring happen to be infeasible—this can indeed be observed in experiments. Then the algorithm can get caught in a random walk through the infeasible region. To avoid this effect we propose to make the weights adaptive to the number of infeasible offspring as follows. We first compute the standard

[4] with both update choices, μ_{eff} is the same and computed from the w_i.

[5] In standard CMA-ES the rank is even reduced to $\mu + 1$ due to truncation selection effectively setting the weights of discarded offspring to zero.

weights w_i of CMA-ES. Infeasible offspring are treated as worst, generally obtaining low weights. For the active CMA update we subtract a constant from the weights of infeasible offspring, which is set to $\frac{0.4}{\lambda}\sum_i w_i$. Then we normalize the absolute values of the weights to one (by dividing all weights by $\sum_i |w_i|$), compute μ_{eff} based on these weights, and then make them mean free by subtracting $\frac{1}{\lambda}\sum_i w_i$. Finally, the parameters $c_s, d_\sigma, c_1, c_c, c_\mu$ are recomputed based on the new weights.

Finally, we ensure that the mean m stays in the feasible region by finding the minimum $k \in \mathbb{N}_0$ such that

$$m + \gamma^k (m' - m) \ , \qquad (1)$$

where we set $\gamma = 2/3$.

3. EXPERIMENTS

In this section we perform an empirical evaluation of xCMA-ES and compare it to CMA-ES. This comparison highlights the effect of the exponential covariance matrix parameters and the resulting multiplicative covariance matrix update. In particular, we aim to answer the following questions:

1. Does the exponential update impact black box performance?
2. How do active covariance updates perform in exponential form?

In order to answer the first question we perform experiments on the same standard benchmark sets as used for the xNES algorithm [7].

Due to lack of a fair baseline for comparison we cannot provide a good answer for the second question. However, we demonstrate that active updates work qualitatively as desired in a constraint optimization setting. As the CMA-ES is tuned on a large number of benchmark functions, ranking stability over speed, it is not hard to find parameters which beat it on a smaller set of functions. Thus tuning the parameters of the xCMA-ES on a small set of function would lead to an unfair benchmark. To avoid this situation we avoid tuning altogether and instead resort to the default parameters of CMA-ES which are given in table 1. This choice is justified by the similarity of the update rules, especially in the settings of small dimensionality with the conservative learning rates of CMA-ES.

constant	value
λ, μ	$4 + \lfloor 3\log(d) \rfloor$, $\lambda/2$
c_s	$\frac{\mu_{\text{eff}}+2}{d+\mu_{\text{eff}}+5}$
c_c	$\frac{4+\mu_{\text{eff}}/d}{d+4+2\mu_{\text{eff}}/d}$
c_1	$\frac{2}{(d+1.3)^2+\mu_{\text{eff}}}$
c_μ	$\min\left\{1 - c_1, 2\frac{\mu_{\text{eff}}-2+1/\mu_{\text{eff}}}{(d+2)^2+2\mu_{\text{eff}}/2}\right\}$
D_σ	$1 + c_s + 2\max\{0, \sqrt{\frac{\mu_{\text{eff}}-1}{d+1}} - 1\}$
w_i	$\frac{\max\{0,\log(\lambda/2-1)-\log(i-1)\}}{\sum_{j=1}^{\lambda}\max\{0,\log(\lambda/2-1)-\log(i-1)\}}$

Table 1: Constants used for the CMA-ES and xCMA-ES algorithms

3.1 Black Box Performance

To assess the black box performance of xCMA-ES, we compare it to CMA-ES on the set of benchmark functions used in [7]. We run both algorithms for 100 trials on each function for each dimensionality $d \in \{4, 8, 16, 32, 64\}$ until a target value f_{stop} is reached. We chose as initial step size for all functions $\sigma = \frac{1}{\sqrt{d}}$. We report the median of the required function evaluations over the successful trials for both algorithms. A trial is considered successful if the algorithm converges to the right optimum(which is only an issue on Rosenbrock). The function descriptions and the values of f_{stop} are given in table 2. The results of the experiments are given in Figure 1.

name	$f(x)$	f_{stop}
SharpRidge	$-x_1 + 100\sqrt{\sum_{i=2}^d x_i^2}$	-1000
ParabRidge	$-x_1 + 100\sum_{i=2}^d x_i^2$	-1000
Rosenbrock	$\sum_{i=1}^{d-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	10^{-14}
Sphere	$\sum_{i=1}^d x_i^2$	10^{-14}
Cigar	$\alpha x_1^2 + \sum_{i=2}^d x_i^2$	10^{-14}
Discus	$x_1^2 + \alpha\sum_{i=2}^d x_i^2$	10^{-14}
Ellipsoid	$\sum_{i=1}^d \alpha^{\frac{i}{d-1}} x_i^2$	10^{-14}
Schwefel	$\sum_{i=1}^d \left(\sum_{j=1}^i x_i\right)^2$	10^{-14}
DiffPowers	$\sum_{i=1}^d \lvert x_i \rvert^{2+10\frac{i-1}{d}}$	10^{-14}

Table 2: The formulas for the benchmark functions. The default value $\alpha = 10^{-6}$ was used in all experiments.

The results show that both algorithms perform equally well on all functions, showing nearly no differences between the algorithms, with a minimal (maybe insignificant and surely irrelevant) edge for xCMA-ES.

3.2 Constraint Handling

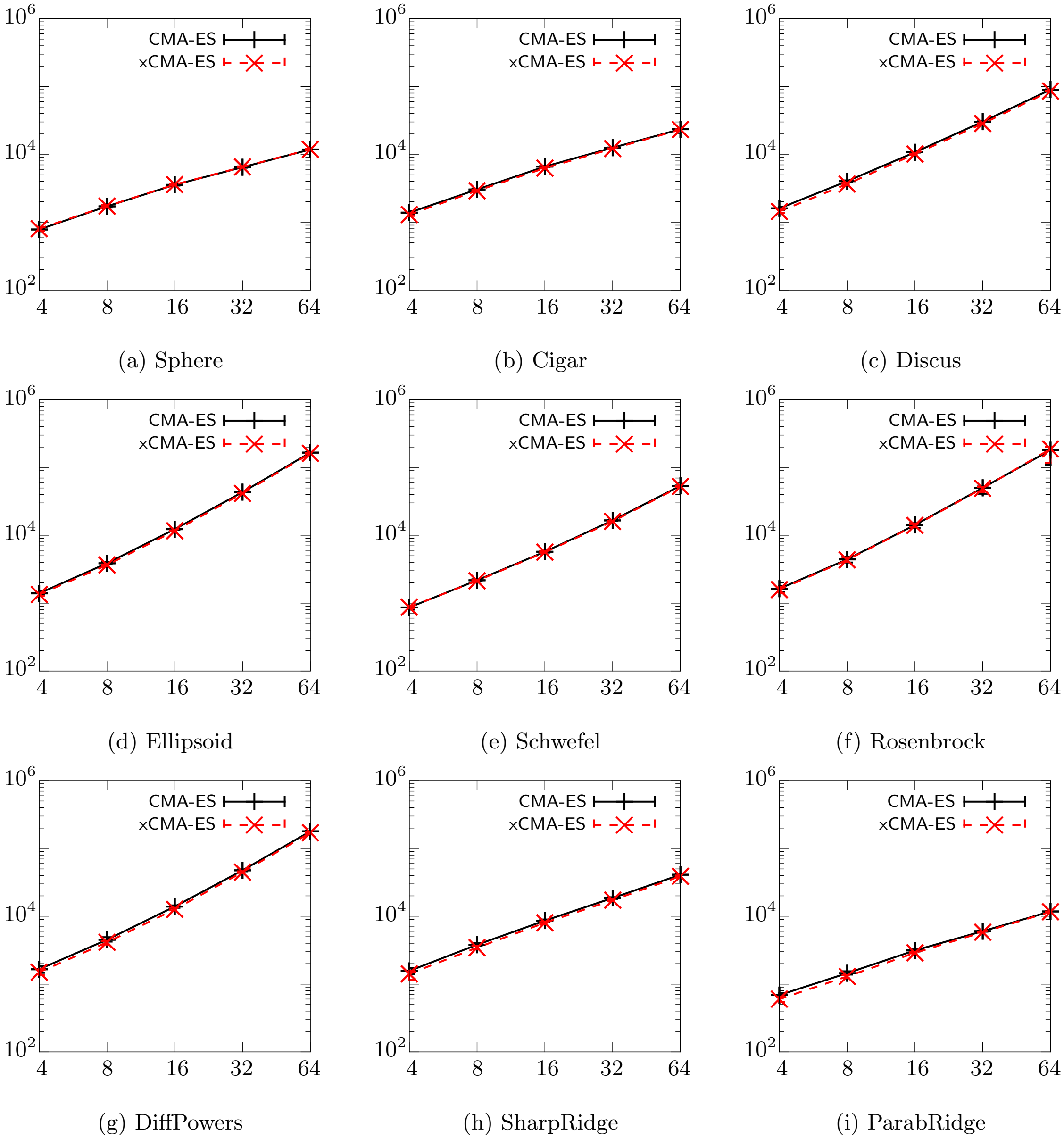
Due to a lack of a non-elitist CMA-ES variant with a constraint handling mechanism based on (active) covariance matrix updates we can not provide a strong baseline algorithm for comparison with xCMA-ES. To obtain a reasonable baseline we constrain the mean of CMA-ES to never leave the feasible region with the same mechanism as proposed for xCMA-ES (see equation (1)). The standard selection operator is already capable of handling constraints to some extent when treating infeasible offspring as worse than feasible ones. The only differences between the baseline CMA-ES and xCMA-ES are the additive update vs. the multiplicative update with active constraint handling.

As a benchmark problem we use the constrained sphere function proposed in [3]:

$$\min_x \ \|x\|^2 - m \qquad \text{s.t } \ x_i \geq 1, i = 1, \dots, m$$

The optimum x^* has components $x_i^* = 1$ for $i \leq m$ and $x_i^* = 0$ for $i > m$. The difficulty of this problem is twofold: In the vicinity of the optimum only a fraction of 2^{-m} of the space it feasible—hence the problem gets harder with growing number of constraints. At the same time, while approaching the optimum the gradient of the objective function in the unconstrained components towards zero vanishes. To counteract this trend the algorithm must reduce the variance of its sampling distribution in the subspace spanned by all constraint normals—a task for which active CMA updates are supposed to be helpful. We perform 100 runs of both algorithm in $d = 16$ and $d = 32$ dimensions with $m \in \{2, 4, \dots, d/2\}$ constraints.

The results are given in figure 2. A plot of an example run with $d = 32$ and $m = 4$ is given in figure 3 showing the



(a) Sphere

(b) Cigar

(c) Discus

(d) Ellipsoid

(e) Schwefel

(f) Rosenbrock

(g) DiffPowers

(h) SharpRidge

(i) ParabRidge

Figure 1: Median number of black box function evaluations over problem dimension of CMA-ES and xCMA-ES on nine standard benchmark problems. Due to low spread, the 25% and 75% quantile indicators are nearly invisible.

function values of the best point in the sampled population as well as the current step-size.

It turns out that xCMA-ES is always faster then the baseline CMA-ES. Moreover, CMA-ES became unstable with increasing m and in the case $d = 32$ and $m = 16$ fails to converge in all 100 runs. The plot in figure 3 shows this instability already for $m = 4$ constraints.

4. CONCLUSION

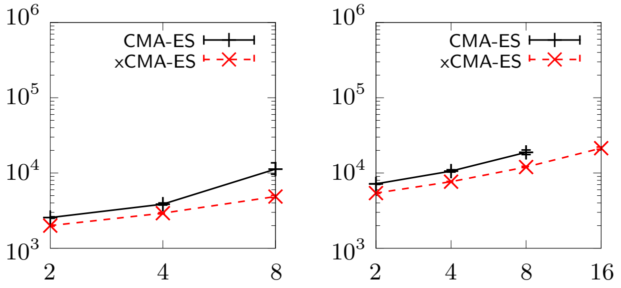
We have incorporated a new type of covariance matrix update into the CMA-ES algorithm. This multiplicative update stems from the xNES algorithm. It guarantees positive definiteness of the covariance matrix even with negative update weights. The resulting algorithm features all mechanisms of CMA-ES and is hence called xCMA-ES.

We showed that its performance on standard benchmarks is nearly indistinguishable from that of the CMA-ES. We demonstrated that, despite the application of the matrix exponential, it is possible to implement the multiplicative update with a time complexity of only $O(d^2\lambda)$.

We further investigated an extension of the algorithm for constrained optimization problems and showed that by a simple use of negative weights xCMA-ES outperforms the CMA-ES. This demonstrates the benefits of the multiplicative update.

Acknowledgements

Oswin Krause acknowledges support from the Danish National Advanced Technology Foundation through the project “Personalized breast cancer screening”.



(a) $d = 16$ (b) $d = 32$

Figure 2: Number of iterations (generations) necessary to reach the target objective value of 10^{-12} over the number m of constraints for the constrained sphere problems in $d = 16$ and $d = 32$ dimensions. The value for CMA-ES at $d = 32$ and $m = 16$ is missing because the algorithm failed in all 100 runs.

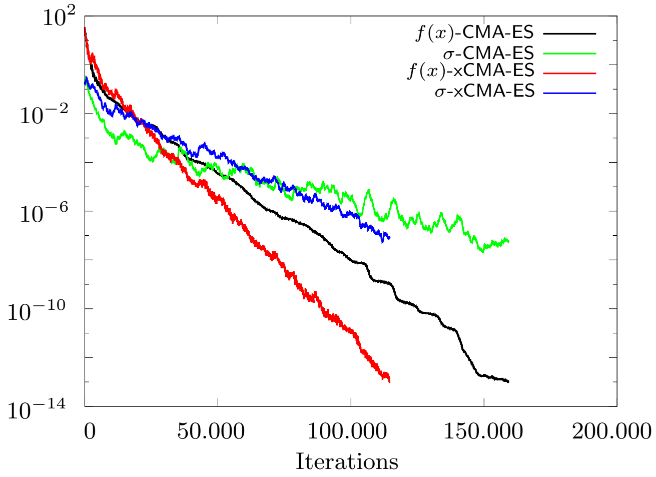


Figure 3: Plot of a single run of the CMA-ES and xCMA-ES algorithms on the constrained Sphere function with $d = 32$ and $m = 4$. Plotted are function value and step length over generations.

5. REFERENCES

[1] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi. Bidirectional Relation between CMA Evolution Strategies and Natural Evolution Strategies. In *Parallel Problem Solving from Nature (PPSN)*, 2010.

[2] D. V. Arnold and N. Hansen. Active covariance matrix adaptation for the (1+1)-CMA-ES. In *Proceedings of the 12th annual conference on Genetic and Evolutionary Computation (GECCO)*, pages 385–392. ACM, 2010.

[3] D. V. Arnold and N. Hansen. A (1+1)-CMA-ES for constrained optimisation. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2012)*, pages 297–304. ACM, 2012.

[4] A. Auger. Convergence results for the $(1, \lambda)$-SA-ES using the theory of φ-irreducible Markov chains. *Theoretical Computer Science*, 334(1–3):35–69, 2005.

[5] A. Auger and N. Hansen. A restart CMA evolution strategy with increasing population size. In B. McKay et al., editors, *The 2005 IEEE International Congress on Evolutionary Computation (CEC'05)*, volume 2, pages 1769–1776, 2005.

[6] H.-G. Beyer. Convergence Analysis of Evolutionary Algorithms that are Based on the Paradigm of Information Geometry. *Evolutionary Computation*, 22(4):679–709, 2014.

[7] T. Glasmachers, T. Schaul, Y. Sun, D. Wierstra, and J. Schmidhuber. Exponential natural evolution strategies. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2010.

[8] 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.

[9] N. Hansen, S. P. N. Niederberger, L. Guzzella, and P. Koumoutsakos. A Method for Handling Uncertainty in Evolutionary Optimization with an Application to Feedback Control of Combustion. *IEEE Transactions on Evolutionary Computation*, 13(1):180–197, 2009.

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

[11] G. A. Jastrebski and D. V. Arnold. Improving evolution strategies through active covariance matrix adaptation. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 2814–2821. IEEE, 2006.

[12] 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(1):77–112, 2004.

[13] O. Krause and C. Igel. A More Efficient Rank-one Covariance Matrix Update for Evolution Strategies. In J. He, T. Jansen, G. Ochoa, and C. Zarges, editors, *Foundations of Genetic Algorithms (FOGA)*. ACM Press, 2015. accepted for publication.

[14] Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. Information-geometric optimization algorithms: a unifying picture via invariance principles. *arXiv preprint arXiv:1106.3708v3*, 2014.

[15] Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Stochastic Search using the Natural Gradient. In *International Conference on Machine Learning (ICML)*, 2009.

[16] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. *Journal of Machine Learning Research*, 15:949–980, 2014.

[17] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural Evolution Strategies. In *Proceedings of the Congress on Evolutionary Computation (CEC)*. IEEE Press, 2008.