Exponential Natural Evolution Strategies

Tobias Glasmachers
IDSIA, University of Lugano
Manno 6928, Switzerland
tobias@idsia.ch

Tom Schaul
IDSIA, University of Lugano
Manno 6928, Switzerland
tom@idsia.ch

Sun Yi
IDSIA, University of Lugano
Manno 6928, Switzerland
yi@idsia.ch

Daan Wierstra
EPFL
Lausanne 1015, Switzerland
daan.wierstra@epfl.ch

Jürgen Schmidhuber
IDSIA, University of Lugano
Manno 6928, Switzerland
juergen@idsia.ch

ABSTRACT

The family of natural evolution strategies (NES) offers a principled approach to real-valued evolutionary optimization by following the natural gradient of the expected fitness. Like the well-known CMA-ES, the most competitive algorithm in the field, NES comes with important invariance properties. In this paper, we introduce a number of elegant and efficient improvements of the basic NES algorithm. First, we propose to parameterize the positive definite covariance matrix using the exponential map, which allows the covariance matrix to be updated in a vector space. This new technique makes the algorithm completely invariant under linear transformations of the underlying search space, which was previously achieved only in the limit of small step sizes. Second, we compute all updates in the *natural* coordinate system, such that the natural gradient coincides with the vanilla gradient. This way we avoid the computation of the inverse Fisher information matrix, which is the main computational bottleneck of the original NES algorithm. Our new algorithm, exponential NES (xNES), is significantly simpler than its predecessors. We show that the various update rules in CMA-ES are closely related to the natural gradient updates of xNES. However, xNES is more principled than CMA-ES, as all the update rules needed for covariance matrix adaptation are derived from a single principle. We empirically assess the performance of the new algorithm on standard benchmark functions.

Categories and Subject Descriptors

[**Evolution Strategies and Evolutionary Programming**]

Keywords

evolution strategies, natural gradient, black box optimization, unconstrained optimization

1. INTRODUCTION

Evolutionary algorithms aim to optimize a 'fitness' function that is either unknown or too complex to be modeled directly. Their weak assumptions on the type of fitness function make them applicable to black box optimization problems. For example, evolution strategies (ESs) can typically handle noise and do not require the fitness function to be differentiable or even continuous. Being direct search methods, they rely on the fitness value only and do not require gradients or higher derivatives. Such algorithms allow domain experts to search for good or near-optimal solutions to numerous difficult real-world problems in areas ranging from medicine and finance to control and robotics.

Natural Evolution Strategies (NES [13]), form a new class of evolutionary algorithms that maintain and iteratively update a multivariate Gaussian mutation distribution. Parameters are updated by estimating a *natural evolution gradient*, i.e., the natural gradient on the parameters of the mutation distribution, and following it towards better expected fitness. A well-known advantage of natural gradient methods over 'vanilla' gradient ascent is isotropic convergence on fitness landscapes with highly correlated coordinates [1]. Although relying exclusively on function value evaluations, the resulting optimization behavior closely resembles second order optimization techniques. This avoids drawbacks of regular gradients which are prone to slow or even premature convergence [8].

Although more principled and theoretically sound than vanilla gradient ascent methods, the original NES algorithm [13] struggles with similar problems. On some benchmark functions it converges slowly or even prematurely due to instabilities of its updates, most probably resulting from unreliable estimates of the natural gradient, particularly in medium to high dimensional problems.

Recent work on the Exact NES (eNES) algorithm [12, 11] has improved the robustness and reduced the computational complexity of NES. This was achieved by analytically computing the exact Fisher information matrix, which is needed for the natural gradient, instead of estimating it from samples, and by introducing a number of novel techniques such as importance mixing, fitness baselines [12], and a computationally efficient update of the inverse Fisher matrix [11]. However, these improvements come at the cost of relinquishing true invariance w.r.t. linear transformations of the search space.

Here we propose a number of novel techniques to resolve the problems described above, building on both the original NES algorithm and the improved eNES variant. In this paper we introduce an *exponential parametrization* of the search distribution that guarantees invariance, while at the same time providing an elegant and efficient way of computing the natural gradient without the need of the explicit Fisher information matrix (or its costly inverse). The key insight consists in performing a 'natural' *change of coordinate system* at each step. Furthermore, we uncover a close relationship between the resulting updates of the search distribution and those of the well-known CMA-ES [4, 6] algorithm, providing a retroactive theoretical justification for some of its heuristics.

The resulting algorithm, *exponential NES* (xNES), is simpler and significantly more stable, even with greatly reduced population sizes. On standard unimodal benchmarks, we show that xNES is consistently faster than its predecessors.

The remainder of this paper is organized as follows: We introduce the basic NES scheme in section 2 and its extension eNES in section 3. Then we propose two novel techniques that address their shortcomings in section 4 and summarize the resulting xNES algorithm in section 5. Section 6 is dedicated to our experimental evaluation. We finish with conclusions in section 7.

2. NATURAL EVOLUTION STRATEGIES

The family of natural evolution strategies (NES) [13] exhibits the typical characteristics of evolution strategies (ESs). It maintains a population of vector-valued candidate solutions, and samples new offspring and adapts its search distribution generation-wise. The essential concepts of NES are briefly revisited in the following.

Gradient of Expected Fitness. The core idea of NES is its strategy adaptation mechanism: NES follows a sampled natural gradient of expected fitness in order to update (the parameters of) the search distribution. In its standard form, NES uses Gaussian distributions with fully adaptive covariance matrix, but it may in principle be used with a different family of search distributions.

We collect the parameters of the Gaussian, the mean $\mu \in \mathbb{R}^d$ and the covariance matrix $C \in \mathbb{R}^{d\times d}$, in the variable $\theta = (\mu, C)$. However, to sample efficiently from this distribution we need a square root of the covariance matrix (a matrix $A \in \mathbb{R}^{d\times d}$ fulfilling $AA^T = C$). Then $x = \mu + Az$ transforms a standard normal vector $z \sim \mathcal{N}(0, I)$ into a sample $x \sim \mathcal{N}(\mu, C)$. Let

$$p(x \,|\, \theta) = \frac{1}{(\sqrt{2\pi})^d \det(A)} \cdot \exp\left(-\frac{1}{2}\left\|A^{-1} \cdot (x-\mu)\right\|^2\right)$$

denote the density of the normal search distribution $\mathcal{N}(\mu, C)$. Then,

$$J(\theta) = \mathbb{E}[f(x) \,|\, \theta] = \int f(x)\; p(x \,|\, \theta)\; dx \qquad (1)$$

is the expected fitness under the search distribution given by θ. The so-called 'log-likelihood trick' enables us to write

$$\begin{aligned}
\nabla_\theta J(\theta) =& \nabla_\theta \int f(x)\; p(x \,|\, \theta)\; dx \\
=& \int f(x)\; \nabla_\theta p(x \,|\, \theta)\; dx \\
=& \int f(x)\; \nabla_\theta p(x \,|\, \theta)\; \frac{p(x \,|\, \theta)}{p(x \,|\, \theta)}\; dx \\
=& \int \Big[f(x)\; \nabla_\theta \log(p(x \,|\, \theta))\Big]\; p(x \,|\, \theta)\; dx \;\; .
\end{aligned}$$

From this form we obtain the Monte Carlo estimate

$$\nabla_\theta J(\theta) \approx \frac{1}{n} \sum_{i=1}^{n} f(x_i)\; \nabla_\theta \log(p(x \,|\, \theta))$$

of the expected fitness gradient. For the Gaussian search distribution $\mathcal{N}(\mu, C)$, the term $\nabla_\theta \log(p(x \,|\, \theta))$ can be computed efficiently, see e.g. [12].

Natural Gradient. Instead of using the stochastic gradient directly for updates, NES follows the *natural* gradient [1]. In a nutshell, the natural gradient amounts to $G = F^{-1}\nabla_\theta J(\theta)$, where F denotes the Fisher information matrix of the parametric family of search distributions. Note that for d-dimensional search space the parameter vector $\theta = (\mu, C)$ has $d + d(d+1)/2 \in \mathcal{O}(d^2)$ components, such that the Fisher matrix F consists of $\mathcal{O}(d^4)$ entries. In principle, the Fisher matrix can be estimated from samples [13]. Computing and inverting[1] this matrix in a naïve implementation takes $\mathcal{O}(d^6)$ operations, which is intractable even for moderate dimensions.

Natural gradient ascent has well-known advantages over vanilla gradient ascent. Most prominently it results in isotropic convergence on ill-shaped fitness landscapes because the natural gradient is invariant under linear transformations of the search space.

Fitness Shaping. NES introduces rank-based fitness shaping in order to render the algorithm invariant under monotonically growing (i.e., rank preserving) transformations of the fitness function. For this purpose, the fitness of the population is shaped into a set of utility values $u_1 \geq \cdots \geq u_n$. Let x_i denote the i^{th} best individual (the i^{th} individual in the population, sorted by fitness, such that x_1 is the best and x_n the worst individual). Replacing fitness with utility, the gradient estimate becomes, with slight misuse of notation,

$$\nabla_\theta \widehat{J}(\theta) = \sum_{i=1}^{n} u_i\; \nabla_\theta \log(p(x_i \,|\, \theta)) \;\; . \qquad (2)$$

Selection. In NES, parent and offspring population are both of the same size n. In each generation the whole population is replaced with the newly sampled offspring population, such that there is essentially no selection mechanism at

[1]Care has to be taken because the Fisher matrix estimate may not be (numerically) invertible even if the exact Fisher matrix is.

work. Instead of actually selecting individuals, NES employs the smoother concept of assigning utility values, which are then used to adapt the search distribution. Thus, the state of the NES algorithm is essentially encoded in the search distribution, and not in the population.

Invariance Properties. Invariance against a large set of transformations of the fitness function and/or the underlying search space is a desirable property for evolution strategies. Just like CMA-ES [4, 6], NES enjoys a number of invariance properties. Rank-based fitness shaping makes the algorithm invariant under monotonic transformations of the fitness function, and the natural gradient is invariant under linear transformations of the search space. Thus, when transforming both the search space and the initial search distribution with the same linear transformation, the natural gradient is transformed accordingly.

3. BOTTLENECKS AND FIXES: EFFICIENT EXACT NES

In this section we briefly outline the current state-of-the-art *Exact NES* (eNES) algorithm as presented in [12]. Its efficiency was improved in [11]. We point out the most prominent performance bottlenecks of the basic NES scheme and discuss the various strategies introduced in eNES to overcome these difficulties.

Exact Fisher Matrix Computation. In the original NES algorithm the Fisher information matrix is estimated from the samples in the population. This process, and even worse the inversion of the Fisher matrix, add high variance to the gradient steps. Restricting the NES algorithm to Gaussian search distributions with adaptive covariance matrix allows for the *exact* analytical computation of the Fisher matrix. This procedure results in the name exact NES (eNES). The complexity per generation of the eNES algorithm has been reduced from $\mathcal{O}(d^6)$ to $\mathcal{O}(d^3)$ operations by representing the covariance matrix C by its (lower triangular) Cholesky decomposition. Then the Fisher matrix turns out to have a block-diagonal structure which can be exploited for iteratively computing F^{-1}.

Fitness Baselines. In order to reduce the variance of the gradient (2), NES uses a fitness baseline. The baseline is further improved in eNES, taking into account the block structure of the Fisher information matrix. Experiments in [12] confirm that the application of fitness baselines significantly reduces premature convergence.

Importance Mixing. Whenever gradient steps are reasonably small, the search distributions in subsequent generations will overlap to some extent. In this situation eNES re-uses samples from the old distribution, i.e., individuals from the current population, with a technique called *importance mixing*. This technique makes sure that the population is still sampled from the correct distribution, although individuals from the previous population are re-used with a positive probability depending on the quotient of the densities of old and new search distribution. This has the effect that subsequent populations are no more independent. It has been shown empirically [12] that importance mixing can save a large fraction of the otherwise n fitness evaluations per generation. On the other hand it requires larger population sizes n, which in part compensates this effect. An interesting side effect seems to be that importance mixing renders eNES less sensitive to parameter settings.

Selection. Importance mixing is an interesting technique in the light of evolutionary computation. While in the basic version of NES a completely new population is sampled in each generation, importance mixing allows individuals in eNES to survive in principle indefinitely. Here, the selection pressure is not directly coupled to fitness or utility, but instead to the probability of being sampled from the current search distribution.

Invariance. As the natural gradient is invariant under linear transformations of the search space, the resulting algorithm should inherit this property. However, this is not completely true for eNES, as the corresponding (finite) gradient step is not invariant. This is because the Cholesky decomposition of the covariance matrix C, the resulting block structure of the Fisher matrix, and the corresponding block-wise fitness baselines depend on the coordinate system. Thus, invariance is achieved only in the limit of small step sizes.

4. EXPONENTIAL NES

The eNES algorithm addresses a large part of the issues of the original NES algorithm. However, its performance is not quite satisfactory. Compared to CMA-ES, eNES is still relatively slow (although it managed to close a large fraction of the gap), and it systematically shows premature convergence on some standard benchmark problems such as the Rosenbrock function in high dimensions. Furthermore, invariance is lost to some extent through the otherwise elegant trick to parameterize the covariance matrix by its Cholesky decomposition. The novel approach presented in the next section constitutes alternative solutions to the speed and stability issues of NES, and also manages to recover full invariance.

4.1 Exponential Parameterization

Gradient steps on the covariance matrix C result in a number of technical problems. When updating C directly with the gradient step δC, we have to ensure that $C + \delta C$ still is a valid positive definite covariance matrix. This is not guaranteed *a priori*, because the (natural) gradient δC may be any symmetric matrix. If we instead update a factor A of C, it is at least ensured that AA^T is symmetric and positive semi-definite. But when shrinking an eigenvalue of A it may happen that the gradient step swaps the sign of the eigenvalue, possibly resulting in undesired oscillations.

An elegant way to fix these problems is to represent the covariance matrix using the exponential map for symmetric matrices (see e.g. [3] for a related approach). Let

$$\mathcal{S}_d := \left\{ M \in \mathbb{R}^{d \times d} \,\middle|\, M^T = M \right\}$$

and

$$\mathcal{P}_d := \left\{ M \in \mathcal{S}_d \,\middle|\, v^T M v > 0 \text{ for all } v \in \mathbb{R}^d \setminus \{0\} \right\}$$

denote the vector space of symmetric and the (cone) manifold of symmetric positive definite matrices, respectively. Then the exponential map

$$\exp : \mathcal{S}_d \to \mathcal{P}_d \,, \qquad M \mapsto \sum_{n=0}^{\infty} \frac{M^n}{n!} \tag{3}$$

is a diffeomorphism: The map is bijective, and both exp as well as its inverse map $\log : \mathcal{P}_d \to \mathcal{S}_d$ are smooth. The mapping can be computed in cubic time, for example by decomposing the matrix $M = UDU^T$ into orthogonal U and diagonal D, taking the exponential of D (which amounts to taking the element-wise exponentials of the diagonal entries), and composing everything back[2] as $\exp(M) = U \exp(D) U^T$.

Thus, we can represent the covariance matrix $C \in \mathcal{P}_d$ as $\exp(\xi)$ with $\xi \in \mathcal{S}_d$. The resulting gradient update for ξ has two important properties: First, because $\mathcal{S}_d$ is a *vector space*, any update automatically corresponds to a valid covariance matrix.[3] Second, the update of ξ makes the gradient step invariant w.r.t. linear transformations. This follows from an information geometric perspective, viewing $\mathcal{P}_d$ as the Riemannian parameter manifold equipped with the Fisher information metric. The invariance property is a direct consequence of the Cartan-Hadamand theorem [2].

However, the exponential parameterization considerably complicates the computation of the Fisher information matrix F, which now involves partial derivatives of the matrix exponential (3). This can be done in cubic time per partial derivative according to [7], resulting in an unacceptable complexity of $\mathcal{O}(d^7)$ for the computation of the Fisher matrix.

4.2 Natural Coordinates

The second novel contribution of the paper is a technique that avoids the computation of the Fisher matrix altogether. Instead of using the "global" coordinates $C = \exp(\xi)$ for the covariance matrix, we linearly transform the coordinate system in each iteration to a coordinate system in which the current search distribution is the standard normal distribution with zero mean and unit covariance. Let the current search distribution be given by $(\mu, A) \in \mathbb{R}^d \times \mathcal{P}_d$ with $AA^T = C$. We use the tangent space $T_{(\mu,A)}(\mathbb{R}^d \times \mathcal{P}_d)$ of the parameter manifold $\mathbb{R}^d \times \mathcal{P}_d$, which is isomorphic to the vector space $\mathbb{R}^d \times \mathcal{S}_d$, to represent the updated search distribution as

$$(\delta,\, M) \mapsto (\mu_{\text{new}},\, A_{\text{new}}) = \left(\mu + A\delta,\, A \exp\left(\frac{1}{2} M \right) \right) \,. \tag{4}$$

This coordinate system is *natural* in the sense that the Fisher matrix w.r.t. an orthonormal basis of (δ, M) is the identity matrix. The current search distribution $\mathcal{N}(\mu, AA^T)$ is encoded as $(\delta, M) = (0, 0)$.

[2]The same computation works for the logarithm, and thus also for powers $\mathcal{P}_d \to \mathcal{P}_d$, $M \mapsto M^c = \exp(c \cdot \log(M))$ for all fixed $c \in \mathbb{R}$, for example for the (unique) square root ($c = 1/2$).

[3]The tangent bundle $T\mathcal{P}_d$ of the manifold $\mathcal{P}_d$ is isomorphic to $\mathcal{P}_d \times \mathcal{S}_d$ and globally trivial. Thus, arbitrarily large gradient steps are meaningful in this representation.

Thus, for the variables (δ, M) in equation (4) the vanilla gradient and the natural gradient coincide. Consequently the natural gradient can be computed in $\mathcal{O}(d^3)$ operations. The trick to compute the update in the natural coordinate system is an alternative to the exponential parameterization for making the algorithm invariant under linear transformations of the search space, which is achieved in a very direct and constructive way. In the new coordinate system the log-density becomes

$$\begin{aligned} \log\left(p(x \,|\, \delta, M)\right) = &- \frac{d}{2}\log(2\pi) - \operatorname{tr}(A) \\ &- \frac{1}{2} \left\| \exp\left(-\frac{1}{2} M \right) A^{-1} \cdot (x - \mu) \right\|^2 \,. \end{aligned}$$

4.3 Updates

The resulting update takes a surprisingly simple form compared to earlier versions of the NES algorithm. Consider a population of offspring $x_i = \mu + A \cdot z_i$ with $z_i \sim \mathcal{N}(0, I)$ for $i \in \{1, \ldots, n\}$. Let u_i be the utility associated with the individual of rank i, then the gradient becomes

$$\begin{aligned} G_\delta =& \nabla_\delta \widehat{J}(0,0) \\ =& \sum_{i=1}^{n} u_i \cdot \nabla_\delta|_{\delta=0} \log(p(x_i \,|\, M = 0, \delta)) \\ =& \sum_{i=1}^{n} u_i \cdot \nabla_\delta|_{\delta=0} \left[-\frac{1}{2} \left\| A^{-1} \cdot (x_i - (\mu + \delta)) \right\|^2 \right] \\ =& \sum_{i=1}^{n} u_i \cdot A^{-1} \cdot (x_i - \mu) \\ =& \sum_{i=1}^{n} u_i \cdot z_i \end{aligned} \tag{5}$$

and

$$\begin{aligned} G_M =& \nabla_M \widehat{J}(0,0) \\ =& \sum_{i=1}^{n} u_i \cdot \nabla_M|_{M=0} \log(p(x_i \,|\, \delta = 0, M)) \\ =& \sum_{i=1}^{n} u_i \cdot \nabla_M|_{M=0} \Bigg[-\operatorname{tr}(A) \\ & \qquad - \frac{1}{2} \left\| \exp\left(-\frac{1}{2} M \right) A^{-1} (x_i - \mu) \right\|^2 \Bigg] \\ =& \sum_{i=1}^{n} u_i \cdot \Bigg[-I - [A^{-1}(x_i - \mu)] \cdot \left(-\frac{1}{2} I \right) \\ & \qquad \cdot [A^{-1}(x_i - \mu)]^T \Bigg] \\ =& \frac{1}{2} \sum_{i=1}^{n} u_i \cdot (z_i z_i^T - I) \,. \end{aligned} \tag{6}$$

As an update, this step can of course be modulated by a learning rate.

We decompose the parameter vector space $(\mu, M) \in \mathbb{R}^d \times \mathcal{S}_d$

into the product

$$\mathbb{R}^d \times \mathcal{S}_d = \underbrace{\mathbb{R}^d}_{(\mu)} \times \underbrace{\mathcal{S}_d^{\parallel}}_{(\sigma)} \times \underbrace{\mathcal{S}_d^{\perp}}_{(B)} \ , \tag{7}$$

of orthogonal subspaces. Here, the one-dimensional space $\mathcal{S}_d^{\parallel} = \{\lambda \cdot I \,|\, \lambda \in \mathbb{R}\}$ is spanned by the unit matrix I and $\mathcal{S}_d^{\perp} = \{M \in \mathcal{S}_d \,|\, \mathrm{tr}(M) = 0\}$ denotes its orthogonal complement in $\mathcal{S}_d$. This decomposition is canonical in the sense that no other decomposition is invariant under linear transformations of the search space. The different components have roles with clear interpretations: The (μ)-component G_δ describes the update of the center of the search distribution, the (σ)-component with value $G_\sigma \cdot I$ for $G_\sigma = \mathrm{tr}(G_M)/d$ has the role of a step size update, which becomes clear from the identity $\det(\exp(M)) = \exp(\mathrm{tr}(M))$, and the (B)-component $G_B = G_M - G_\sigma$ describes the update of the transformation matrix, normalized to unit determinant.

On these subspaces we introduce independent learning rates η_μ, η_σ, and η_B, respectively. For simplicity we also split the transformation matrix $A = \sigma \cdot B$ into the step size $\sigma \in \mathbb{R}^+$ and the normalized transformation matrix B with $\det(B) = 1$. Then the resulting update is

$$\mu_{\text{new}} = \mu + \eta_\mu \cdot G_\delta = \mu + \eta_\mu \cdot \sum_{i=1}^{n} u_i \cdot z_i \tag{8}$$

$$\begin{aligned} \sigma_{\text{new}} &= \sigma \cdot \exp\left(\frac{\eta_\sigma}{2} \cdot G_\sigma\right) \\ &= \sigma \cdot \exp\left(\frac{\eta_\sigma}{2} \cdot \mathrm{tr}\left(\sum_{i=1}^{n} u_i \cdot (z_i z_i^T - I)\right) \Big/ d\right) \end{aligned} \tag{9}$$

$$\begin{aligned} B_{\text{new}} &= B \cdot \exp\left(\frac{\eta_B}{2} \cdot G_B\right) \\ &= B \cdot \exp\left(\frac{\eta_B}{2} \cdot \left(\Delta - \frac{\mathrm{tr}(\Delta)}{d} \cdot I\right)\right) \ , \end{aligned} \tag{10}$$

where $\Delta = \sum_{i=1}^{n} u_i \cdot (z_i z_i^T - I)$. In case of $\eta_\sigma = \eta_B$, in this case referred to as η_A, the updates (9) and (10) simplify to

$$\begin{aligned} A_{\text{new}} &= A \cdot \exp\left(\frac{\eta_A}{2} \cdot G_M\right) \\ &= A \cdot \exp\left(\frac{\eta_A}{2} \cdot \sum_{i=1}^{n} u_i \cdot (z_i z_i^T - I)\right) \ . \end{aligned} \tag{11}$$

4.4 Connection to CMA-ES

The family of NES algorithms is related to CMA-ES [4] in the sense that both are evolution strategies using Gaussian search distributions with fully adaptive covariance matrices. They also share the same invariance properties. It turns out that the new xNES algorithm has even tighter links to CMA-ES.

The update (8) is very similar to the update of the center of the search distribution in CMA-ES [4]. The utility function exactly takes the role of the weights in CMA-ES, which assumes a fixed learning rate of one. From equation (11) we deduce the update rule

$$\begin{aligned} C_{\text{new}} &= A_{\text{new}} \cdot (A_{\text{new}})^T \\ &= A \cdot \exp\left(\eta_C \cdot \sum_{i=1}^{n} u_i \left(z_i z_i^T - I\right)\right) \cdot A^T \end{aligned}$$

for the covariance matrix, with learning rate $\eta_C = \eta_A$. The exponential term can be approximated by its first order Taylor expansion

$$\exp\left(\eta_C \cdot \sum_{i=1}^{n} u_i \left(z_i z_i^T - I\right)\right) \approx I + \eta_C \cdot \sum_{i=1}^{n} u_i \left(z_i z_i^T - I\right) \ ,$$

so the first order approximate update yields

$$\begin{aligned} C'_{new} &= A \cdot \left(I + \eta_C \cdot \sum_{i=1}^{n} u_i \left(z_i z_i^T - I\right)\right) \cdot A^T \\ &= (1 - U \cdot \eta_C) \cdot A A^T + \eta_C \cdot \sum_{i=1}^{n} u_i \, (A z_i) \, (A z_i)^T \\ &= (1 - U \cdot \eta_C) \cdot C + \eta_C \cdot \sum_{i=1}^{n} u_i \, (x_i - \mu) \, (x_i - \mu)^T \end{aligned}$$

with $U = \sum_{i=1}^{n} u_i$. We obtain this update when using only the natural coordinate system technique without the exponential parameterization. Up to the evolution path, this update equation coincides with the update of the covariance matrix in CMA-ES. We argue that just like for the global step size σ in CMA-ES, the multiplicative update of the covariance matrix C (or the transformation matrix A) as done in xNES is more natural than the above additive covariance matrix update.

The decomposition (7) exactly corresponds to the representation of the search distribution in CMA-ES, where the covariance matrix is augmented with a global step size parameter. One of the most valuable contributions of natural evolution strategies is to set us into the position to derive the updates of the center μ, the step size σ, and the normalized transformation matrix B, all from the same principle of natural gradient ascent.

5. THE ALGORITHM

In this section we combine the basic components of NES and the new concepts introduced in the previous section to form the exponential NES algorithm (xNES).

The algorithm state is specified by the search distribution, such that the initial search distribution should be input to the algorithm, together with problem dimension and fitness function. In the generation loop we sample n offspring, determine their ranks (which is done by sorting in the actual implementation), and compute the gradient (G_δ, G_M) in the natural coordinate system according to formulas (5) and (6). An application of the update formulas (8), (9), and (10) for the strategy parameters μ, σ, and B completes the loop. Pseudo-code for the xNES algorithm is presented in Algorithm 1.

The tunable parameters of xNES are comprised of the population size n, the learning rates (η_μ, η_σ, and η_B) and the utility function u.

It is highly desirable to have good default settings that scale with the problem dimension and lead to robust performance on a broad class of benchmark functions. Table 1 provides such default values as functions of the problem dimension d. We borrowed several of the settings from CMA-ES, which seems natural due to the apparent similarity discussed in

Algorithm 1: The xNES Algorithm

Input: $d \in \mathbb{N}$, $f : \mathbb{R}^d \to \mathbb{R}$, $\mu \in \mathbb{R}^d$, $A \in \mathbb{R}^{d \times d}$
$\sigma \leftarrow \sqrt[d]{|\det(A)|}$
$B \leftarrow A/\sigma$
while *stopping condition not met* **do**
  for $i \in \{1, \ldots, n\}$ **do**
    $z_i \leftarrow \mathcal{N}(0, I)$
    $x_i \leftarrow \mu + \sigma B \cdot z_i$
  end
  sort $\{(z_i, x_i)\}$ with respect to $f(x_i)$
  $G_\delta \leftarrow \sum_{i=1}^{n} u_i \cdot z_i$
  $G_M \leftarrow \sum_{i=1}^{n} u_i \cdot (z_i z_i^T - I)$
  $G_\sigma \leftarrow \mathrm{tr}(G_M)/d$
  $G_B \leftarrow G_M - G_\sigma \cdot I$
  $\mu \leftarrow \mu + \eta_\mu \cdot \sigma B \cdot G_\delta$
  $\sigma \leftarrow \sigma \cdot \exp(\eta_\sigma / 2 \cdot G_\sigma)$
  $B \leftarrow B \cdot \exp(\eta_B / 2 \cdot G_B)$
end

parameter	default value
n	$4 + \lfloor 3 \log(d) \rfloor$
η_μ	1
$\eta_\sigma = \eta_B$	$\frac{3}{5} \cdot \frac{(3+\log(d))}{d\sqrt{d}}$
u_i	$\frac{\max\left(0, \log(\frac{n}{2}+1) - \log(i)\right)}{\sum_{j=1}^{n} \max\left(0, \log(\frac{n}{2}+1) - \log(j)\right)} - \frac{1}{n}$

Table 1: Default parameter values for xNES as a function of problem dimension d.

section 4.4. Both the population size n and the learning rate η_μ are the same as for CMA-ES, even if this learning rate never explicitly appears in CMA-ES. For the utility function we copied the weighting scheme of CMA-ES, but we normalized the values such that they sum to zero, which is the simplest form of implementing a fitness baseline. In [5] a similar approach has been proposed for CMA-ES. The remaining parameters have been determined via an empirical investigation, aiming for robust performance. They are used throughout this paper. A Python implementation of xNES is available within the open-source machine learning library PyBrain [9].

The alternative to using dimension-dependent default parameters is to use importance mixing (as in [12]). In that case, the exact values of parameters are less important, as importance mixing implicitly adapts them to the situation (by reusing more or less old points). We found this to be robust for xNES as well, but it lead to slightly worse performance.

6. EXPERIMENTS

In this section we present our experimental evaluation of the new xNES algorithm. It is benchmarked against CMA-ES on a set of standard benchmark functions in order to assess its performance with respect to state-of-the-art.

6.1 Experimental Setup

We empirically validate our algorithm on nine unimodal functions out of the set of standard benchmark functions from [10] and [4], that are typically used in the literature, for comparison purposes, and for competitions. We randomly choose the center of the initial search distribution at average distance one from the optimum and start with a radial search distribution of unit variance. In order to prevent potentially biased results, we follow [10] and consistently transform (by a combined rotation and translation) the functions' inputs, making the variables non-separable and avoiding trivial optima (e.g. at the origin). This immediately renders many other methods virtually useless, since they cannot cope with correlated mutation directions. xNES, however, is perfectly invariant under translation and rotation with initially radial search distribution, and even invariant under arbitrary linear transformations if we transform the search distribution accordingly. In addition, the rank-based fitness shaping makes it invariant under order-preserving transformations of the fitness function.

We ran xNES and CMA-ES on this set of benchmark functions with dimensions between 2 and 64 and a target fitness of -10^{-10} (10^3 for the unbounded functions SharpR and ParabR)[4]. A run is counted as a success if it reaches the target fitness within 10^7 total fitness evaluations. A run is terminated and counted as a failure whenever numerical instability is detected.

6.2 Results and Discussion

Figure 1 provides the full results on the set of benchmark functions over a broad range of dimensions up to 64. It demonstrates that xNES scales approximately quadratically on the dimensionality of the problem. In particular, xNES is able to solve the Rosenbrock benchmark in high dimensions consistently, which is a significant improvement over its predecessors. It is also worth noting that the scaling factor of xNES is almost the same over all functions. However, xNES does appear to be more stable, especially in the Sharp ridge function.

Figure 2 illustrates the behavior of CMA-ES, eNES and xNES on the Rosenbrock benchmark, in dimension 8. The performance of xNES is drastically improved compared to eNES. In addition, xNES exhibits a similar behavior to CMA-ES, with the exception of the late phase (descent into the approximately quadratic optimum) where the curve is not as steep. Still, all algorithms under consideration exhibit a log-linear convergence in that late phase, due to their intrinsic scale-invariance.

Another interesting difference between xNES and CMA-ES can be observed for the Sharp ridge function in eight or more dimensions. CMA-ES is prone to premature convergence because it adapts its search distribution too quickly to the ridge, which does not allow it to find a point in the other valley.

We identify two key differences between xNES and CMA-ES. CMA-ES descends faster into approximately quadratic local optima, corresponding to the steeper curve in the late phase in Figure 2. On the other hand, xNES better resists premature convergence.

[4]Note that the minus sign is due to fitness maximization.

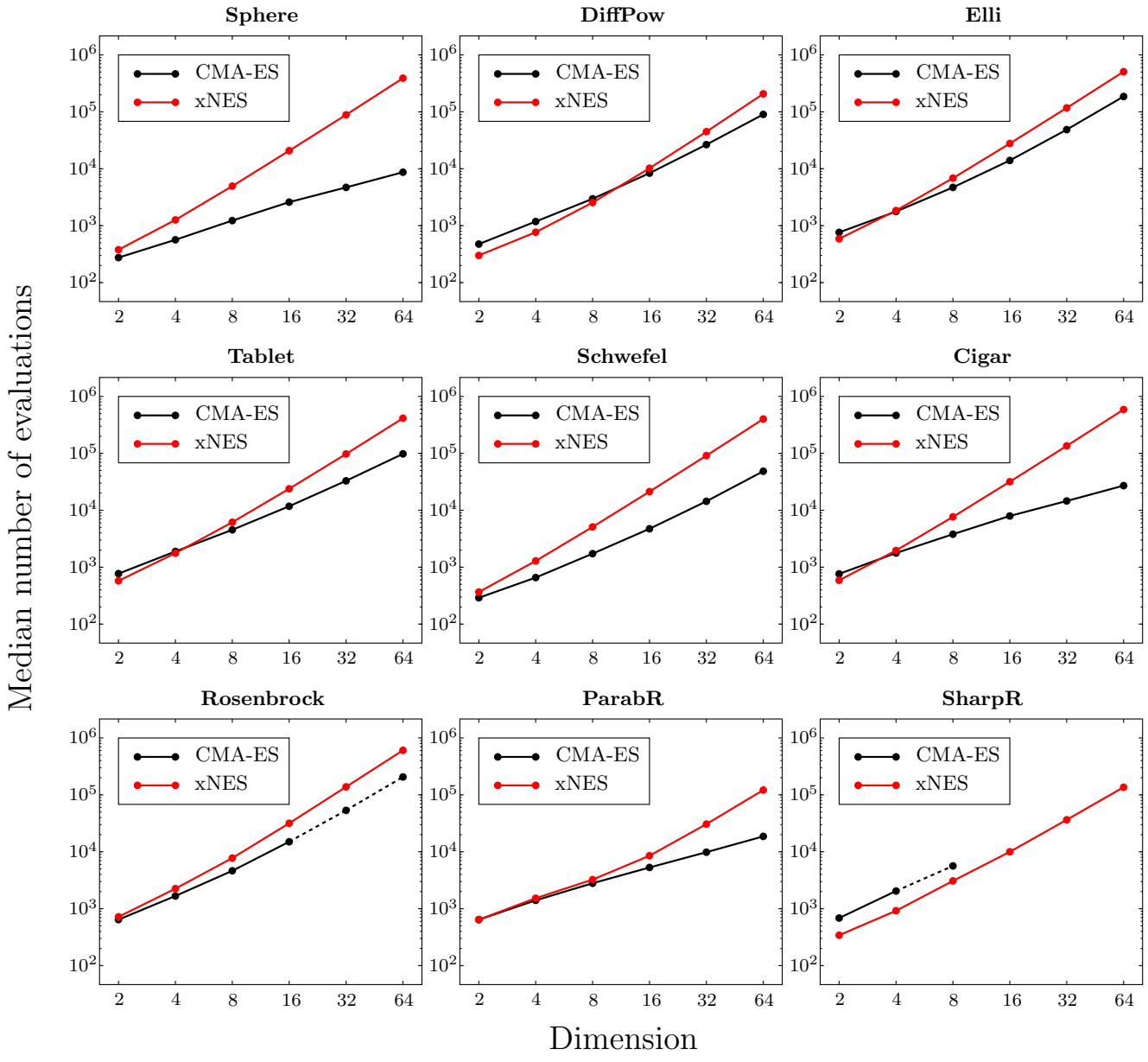

Figure 1: Log-log plot of the median number of fitness evaluations (over the successful trials out of 100) required to reach the target fitness value of -10^{-10} (10^3 for the unbounded functions ParabR and SharpR) for 9 different benchmark functions on dimensions 2 to 64. Dashed connections denote cases where the algorithm prematurely converged in at least 10% of the runs. Setups for which no single run converged are not shown at all. We observe that xNES – unlike CMA-ES – scales identically with dimension (same steepness) for all nine benchmarks. It does not suffer from premature convergence.

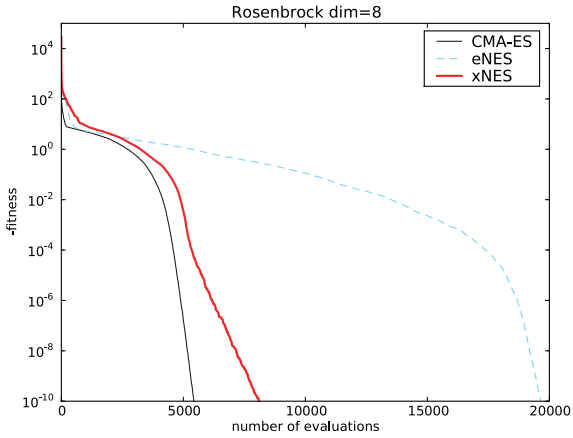

Figure 2: Average number of evaluations required to reach a fitness of -10^{-10} on the Rosenbrock benchmark in dimension 8.

7. CONCLUSION

We introduced the xNES algorithm, a new member of the family of natural evolution strategies. Compared to its predecessors, the new algorithm is more elegant and enjoys better theoretical invariance properties. In our experiments, it shows convincing performance and robustness.

In contrast to the earlier eNES algorithm, xNES is truly invariant w.r.t. linear transformations of the coordinate system. This is an important property for evolution strategies in general whenever a problem specific coordinate system is not known. Furthermore, xNES completely avoids the costly computation and inversion of the Fisher information matrix by executing all updates in a local 'natural' coordinate system, in which the natural gradient coincides with the standard gradient.

The resulting update equations turn out to closely resemble some of the strategy adaptation rules found in CMA-ES. However, we argue that the multiplicative update rule for the covariance matrix in xNES more natural than the additive mixing in CMA-ES. This result indicates a much tighter link between NES and CMA-ES than expected. In particular, we deduced update rules very similar to the diverse rules found in CMA-ES from the single principle of natural gradient ascent. On the one hand, this shows that xNES follows a more principled approach than CMA-ES. On the other hand, we consider this finding an important contribution to the theoretical understanding of the mechanism of covariance matrix adaptation in general.

Empirically, xNES performs on par with CMA-ES on many standard benchmark functions. The new algorithm is slower when descending into quadratic optima, while it outperforms CMA-ES in terms of stability, successfully avoiding premature convergence, e.g. in the 'sharp ridge' benchmark.

The xNES algorithm is a big step forward. Still, it leaves a few problems open for future research. The most pressing one is the choice of learning rates, which is not yet satisfactory. One possible strategy for coming up with more convincing rules is to improve our theoretical understanding of xNES. An orthogonal approach is to apply self-adaptation mechanisms to control (some of) the learning rates. Such an approach may even achieve the best of two worlds in a single algorithm, enabling us to combine optimal learning rates for fast descent into local optima (where the task is to keep the shape of the search distribution stable) with different settings for quick adaptation of the search distribution whenever necessary.

8. ACKNOWLEDGMENTS

This research was funded by SNF grants 200020-116674/1, 200021-111968/1 and 200021-113364/1.

9. REFERENCES

[1] S. Amari and S. C. Douglas. Why natural gradient? In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '98)*, volume 2, pages 1213–1216, 1998.

[2] É. Cartan. Sur la représentation géométrique des syetèmes matérieles non holonomes. In *Proc Int Congr Math, Bologna*, volume 4, pages 253–261, 1928.

[3] T. Glasmachers and C. Igel. Gradient-based Adaptation of General Gaussian Kernels. *Neural Computation*, 17(10):2099–2105, 2005.

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

[5] G. A. Jastrebski and D. V. Arnold. Improving Evolution Strategies through Active Covariance Matrix Adaptation. In *IEEE Congress on Evolutionary Computation*, 2006.

[6] 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.

[7] I. Najfeld and T. F. Havel. Derivaties of the Matrix Exponential and Their Computation. *Adv. Appl. Math*, 16:321–375, 1994.

[8] J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

[9] T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, T. Rückstieß, and J. Schmidhuber. PyBrain. *Journal of Machine Learning Research*, 11:743–746, 2010.

[10] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. Technical report, Nanyang Technological University, Singapore, 2005.

[11] Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Efficient Natural Evolution Strategies. In *Genetic and Evolutionary Computation Conference (GECCO)*, 2009.

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

[13] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural Evolution Strategies. In *Proceedings of the Congress on Evolutionary Computation (CEC08), Hongkong*. IEEE Press, 2008.