Qualitative and Quantitative Assessment of Step Size Adaptation Rules

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

Christian Igel
Department of Computer Science
University of Copenhagen
Copenhagen, Denmark
igel@di.ku.dk

ABSTRACT

We present a comparison of step size adaptation methods for evolution strategies, covering recent developments in the field. Following recent work by Hansen et al. we formulate a concise list of performance criteria: a) fast convergence of the mean, b) near-optimal fixed point of the normalized step size dynamics, and c) invariance to adding constant dimensions of the objective function. Our results show that algorithms violating these principles tend to underestimate the step size or are unreliable when the function does not fit to the algorithm's tuned hyperparameters. In contrast, we find that cumulative step size adaptation (CSA) and two-point adaptation (TPA) provide reliable estimates of the optimal step size. We further find that removing the evolution path of CSA still leads to a reliable algorithm without the computational requirements of CSA.

Categories and Subject Descriptors

[**Continuous Optimization**]

Keywords

evolution strategies, step size adaptation, comparison

1. INTRODUCTION

We consider minimization of a "black-box" function $f : \mathbb{R}^d \to \mathbb{R}$ defined on a d-dimensional real vector space. For optimization we consider Evolution Strategies (ES) using a normal search distribution $\mathcal{N}(m, \sigma^2 C)$ with mean $m \in \mathbb{R}^d$, covariance matrix $C \in \mathbb{R}^{d \times d}$, and global step size $\sigma > 0$. Adaptation of the step size enables linear convergence on scale invariant functions [2], while covariance matrix adaptation (CMA) [8] renders the asymptotic convergence rate independent of the conditioning number of the Hessian.[1]

[1]This statement requires a twice continuously differentiable function with strictly positive definite Hessian in the isolated optimum.

FOGA '17, January 12 - 15, 2017, Copenhagen, Denmark

ISBN 978-1-4503-4651-1/17/01…$15.00

DOI: http://dx.doi.org/10.1145/3040718.3040725

This paper is concerned with the conceptual and empirical comparison of a number of state-of-the-art mechanisms for controlling the algorithm's step size σ.

Step size control mechanisms have been a core component of ESs, dating back to Rechenberg's famous 1/5-rule [16]. Consequently there exists a plethora of step size adaptation (SA) mechanisms [15, 5, 12, 9, 1]

Comparing these algorithms is a problematic task. The traditional approach is to judge the competitors on a set of benchmark problems. However, such a comparison relies heavily on the tuning of the algorithms' hyperparameters. Relying on previously tuned settings of the hyperparameters does not lead to a fair comparison as tuning always involves a trade-off between performance and the stability of the algorithm on unknown functions, which might lead to a much more conservative tuning which can often be improved for specific tasks. Moreover, even assuming well tuned algorithms, in a regime where one algorithm does not dominate all others on every tested benchmark function it is not straightforward to rank the algorithms.

Recently a new, more qualitative approach was proposed. It is based on how the behavior of an algorithm matches a desired reference behavior [7]. Only a small set of simple benchmark functions is used and algorithms are not assessed by their optimization performance alone, but according to *how* they achieve it. Thus it is not only important how fast an algorithm finds a solution of given accuracy. In addition it is judged according to how effective and robust its internal mechanisms work. This type of analysis gives deeper insight into the algorithm's behavior. Consequently, tuning plays a much smaller role than in a purely benchmark function based comparison.

The present comparison study is similar in nature and draws heavily from the approach proposed by [7]. We modify the original catalog of desiderata only slightly for our comparison, most significantly by adding the criterion of invariance under the addition of insignificant variables. We include two relatively recent methods, namely the median success rule [1] and the population step size adaptation [12]. For simplicity and clarity we perform our analysis for an ES without CMA mechanism (fixing C to the identity matrix).

The paper is organized as follows. In the next section we introduce the simple ES with step size adaptation. Then we formulate a list of desirable properties for step size adaptation methods (section 3). In section 4 we review a variety of existing SA methods and list their properties as far as they can be derived analytically. This analysis is enriched with

Table 1: Constants used in the ES.

constant	value
λ	$4 + \lfloor 3\log(d) \rfloor$
w_i	$\frac{\max\{0,\log(\lambda/2+1/2)-\log(i)\}}{\sum_{j=1}^{\lambda}\max\{0,\log(\lambda/2+1/2)-\log(j)\}}$
σ_0	$\frac{1}{\sqrt{d}}$

empirical data in section 5. The results obtained in sections 4 and 5 culminate in a discussion (section 6), from which we extract our conclusions (section 7).

2. EVOLUTION STRATEGIES

Throughout this paper we consider the following ES. At the beginning of each iteration t it samples a new offspring population $\{x_{t,1}, \ldots, x_{t,\lambda}\}$ of size λ from its Gaussian search distribution with density

$$p_t(x) = \frac{1}{\sqrt{2^d\pi^d\sigma_t^d}} \exp\left(-\frac{\|x - m_t\|^2}{2\sigma_t^2}\right) \ ,$$

which is parameterized by the current mean m_t and the step size (standard deviation) σ_t. In the next step it computes the function values $f(x_{t,i})$ and sorts the offspring by fitness, so that $f(x_{t,i}) \leq f(x_{t,i+1})$. The next mean m_{t+1} is obtained as a convex combination of the offspring using weights w_i with $\sum_{i=1}^{\lambda} w_i = 1$

$$m_{t+1} = \sum_{i=1}^{\lambda} w_i x_{t,i} \ .$$

Truncation selection is encoded by giving non-zero weights only to the first $\mu = \lambda/2$ points. Finally a step size adaptation (SA) algorithm computes σ_{t+1}. Even though σ_t appears only in the sampling of offspring points it is a crucial parameter since it governs the length of the step $\mu_{t+1} - \mu_t$. For the parameters λ and w_i we use the default values of CMA-ES [8], which are given in Table 1.

3. REQUIREMENTS FOR SA METHODS

The prime measure of performance of an optimization algorithm operating on a continuous domain is its rate of convergence. However, the convergence rate can be measured in different ways, e.g., based on distance in search and objective space. In the following, we will focus our discussion on scale invariant functions, that is functions for which hold $f(a * x) = g(a) \cdot f(x)$, $x \in \mathbb{R}^d$, $a \in \mathbb{R}$. On these functions, most evolution strategies exhibit convergence to the isolated optimum x^* at a linear rate, which is defined in accordance with [7] as[2]

$$r = \lim_{t\to\infty} \exp\left(\frac{1}{2t}\log\left(\frac{f(x_t) - f(x^*)}{f(x_1) - f(x^*)}\right)\right) \ . \qquad (1)$$

Comparing SA methods empirically based on the achieved rate of convergence is hard since their performance depends crucially on their tuning parameters. In practice, the heuristics are tuned on a set of benchmark functions to show good average or worst case performance. What is missing is an assessment of whether the tuning makes sense or whether it is a byproduct of over-fitting to the chosen benchmark functions.

The primary example is an SA algorithm that naturally adapts the step sizes to too small values. Honest tuning of the algorithm might result in hyperparameter settings that slow down the adaptation so much that the observed step sizes are on the right scale. It is obvious that the tuning result is fragile and may yield sub-optimal step sizes on other problems. Thus, severe flaws of an SA method can be hidden and the chosen parameters might only work reasonably well on the given set of benchmark functions, while the performance on unseen functions might be much worse. In the example above, if a new function is chosen so that the optimizer can approach the optimum only slowly, then the step size has more time to adapt and thus the SA method will select a too small step size. This could be observed, e.g., when tuning on the Sphere function and testing the tuned SA algorithm on an Ellipsoid function.

One way of dealing with this problem is to require tuning on an extensive predefined set of objective functions. However, this makes tuning even harder, and new methods have to compete with well-known and thus well tuned algorithms. Also the benefit of this approach can be disputed since even an extensive set of benchmark functions might not give us a deep understanding of the method.

Therefore we follow a different route proposed in [7]: step size adaptation methods are judged according to whether and to which degree they fulfill a set of requirements. The properties we consider in this paper are:

1. **Mean Progress**. The mean m_t is the ES's best current estimate of the optimum, hence it should improve over time. The SA should optimize this progress by maximizing (1). This means that the optimal step sizes found by the algorithm should be close to the one that optimizes progress in each single step and that this estimate is reliable over different problems.

2. **Fixed Point dynamics** of σ. We require that for a properly tuned algorithm the realized step sizes are close to the fixed-point dynamics of the step size adaptation algorithm. This entails that the fixed point is a meaningful estimate of the optimal step size and thus the learning rates merely govern how fast and how well the algorithm tracks it.

3. Invariance to **Constant Dimensions**. The step size should be independent of the addition of variables in which the objective function values are constant (i.e., the function does not depend on these variables). This is important in a black-box setting in which we cannot find a reparameterization that would remove an irrelevant subspace.

Checking these requirements makes it easier to compare algorithms and, more importantly, to judge whether a given empirical comparison is fair or not. For example, comparing an algorithm fulfilling the requirement of the quality of the stationary distribution to one that does not, might not be fair since the latter could be tuned to a small set of benchmark functions while the former is inherently stable across a larger set of problems.

[2]In contrast to the ES, this definition is not invariant under monotonic transformations of fitness values. However, it is well-defined for quadratic objective functions, which are used in the experimental evaluation.

In addition to these requirements Hansen et al. [7] consider the following goals: on a random or flat fitness $\log(\sigma_t)$ should perform an unbiased random walk, σ_t should grow exponentially fast on a linear function, and the algorithm should be invariant under translation and rotation of the search space. We are not testing these since the desired behavior on a random or flat fitness is debatable, and in our setup all considered algorithms grow σ_t at an exponentially rate on a linear function and fulfill the invariance properties by design.

4. ALGORITHMS

In this section we introduce seven different step size adaptation (SA) algorithms and analyze them in terms of the requirements listed in section 3.

4.1 Cumulative Step Size Adaptation (CSA)

Cumulative Step Size Adaptation (CSA) is the current state-of-the-art used in the CMA-ES algorithm. It is based on the assumption that when the step size is well adapted, steps taken will become uncorrelated [15] on average. The CSA computes an evolution path, a long-term average of previous steps, and compares its length with the length expected for uncorrelated steps. When samples are drawn from a normal distribution with mean m_t and covariance matrix C_t, the update of the evolution path $p_{\sigma,t}$ is given by

$$p_{\sigma,t+1} = (1-c_p)p_{\sigma,t} + \sqrt{c_\sigma(1-c_p)\mu_{\text{eff}}}C_t^{-1/2}\frac{m_{t+1}-m_t}{\sigma_t} \quad .$$

Here, $\mu_{\text{eff}} = \left(\sum_{i=1}^{\lambda} w_i^2\right)^{-1}$ is the effective sample size of the weighted individuals, c_p is a hyperparameter, and $C_t^{-1/2}$ is the matrix square root. The term $C_t^{-1/2}\frac{m_{t+1}-m_t}{\sigma_t}$ is the last step of the mean transformed into a coordinate system which is independent of σ_t and C_t, and $\sqrt{c_\sigma(1-c_\sigma)\mu_{\text{eff}}}$ is a normalization term ensuring $p_{\sigma,t} \sim \mathcal{N}(0, I_d)$ assuming $\sqrt{\mu_{\text{eff}}}C_t^{-1/2}\frac{m_{t+1}-m_t}{\sigma_t} \sim \mathcal{N}(0, I_d)$. Thus, $\|p_\sigma\|$ can be compared with the expected length of a sample from the standard normal distribution. If it is longer (shorter), then the step size in increased (decreased) in an exponential manner. The resulting update reads

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|p_{\sigma,t+1}\|}{\chi_d} - 1\right)\right) \quad ,$$

where d_σ is another damping factor and χ_d is the expectation of the χ distribution with d degrees of freedom.

The CSA method is not invariant w.r.t. the addition of constant dimensions because all individuals and hence the mean take on random values in these components, which adds noise to the path vector. The fixed point dynamics of the CSA is not independent of its hyperparameters, as the learning rate of the evolution path changes how correlation is measured. In the extreme case $c_\sigma = 1$ correlation is ignored completely and only the length of the current step is taken into account.

4.2 xNES SA

The exponential Natural Evolution Strategy (xNES) algorithm [5, 18] is an instance of the information geometric optimization (IGO) method [14]. The xNES algorithm applies multi-variate Gaussian distributions with mean m and covariance matrix C. The square root $\sigma = \sqrt[2d]{\det(C)}$ of the geometric mean of the eigenvalues of C is interpreted as the method's global step size parameter, and xNES allows to control the learning rate for this scale parameter independent of the learning rates for the shape of the distribution.

The IGO framework treats all parameters of the search distribution the same, irrespective of their role. Hence there is no dedicated mechanism for the adaptation of the step size, instead the parameter is adapted just like the mean and the remaining covariance parameters. The update rule is derived from stochastic natural gradient descent on the statistical manifold spanned by the search distribution parameters, where the offspring population serves as a Monte Carlo sample. For details we refer the interested reader to [5, 14].

The simplistic step size adaptation rule of the xNES algorithm can be extracted as a standalone method, subsequently referred to as xNES SA. The update rule reads

$$\sigma_{t+1} = \sigma_t \cdot \exp\left(\frac{c_\sigma}{\sqrt{d}} \cdot \sum_{i=1}^{\lambda} w_i \cdot \left[\left\|C_t^{-1/2}\frac{x_{t,i}-m_t}{\sigma_t}\right\|^2 - d\right]\right) \quad ,$$

where $c_\sigma > 0$ is a hyperparameter. The rule is based solely on the immediate offspring $x_{t,i}$, which enter the stateless exponential update term as standardized samples. The squared norms of these samples are compared to their expectation (the dimension d). If highly weighted offspring correspond to steps longer than the expectation, then the argument of the exponential is positive and the step size is increased. If successful steps are shorter, the step size is decreased.

4.3 mean-xNES SA

In a variant of xNES SA we replace the samples by the actual step taken. This leads to an update similar to the CSA without an evolution path

$$\sigma_{t+1} = \sigma_t \cdot \exp\left(\frac{c_\sigma}{d} \cdot \left[\mu_{\text{eff}}\left\|C_t^{-1/2}\frac{m_{t+1}-m_t}{\sigma_t}\right\|^2 - d\right]\right) \quad .$$

We call this the mean-xNES SA. In contrast to the xNES SA, we scale the χ^2-statistic by d^{-1}, not $d^{-1/2}$, motivated by our experimental findings in Section 6. The difference in scaling factor does not matter in most cases as it can be compensated by the learning rate. However, when changing the learning rate this scaling turns out to be more stable across different dimensions.

4.4 xNES SA with Log-normal Prior

One disadvantage of xNES SA and mean-xNES SA is that they do not compare a-priori defined step sizes, but only a-posteriori observed step sizes, which are not invariant under the addition of constant dimensions. We can expand the xNES approach by introducing a prior probability distribution on σ and amend the ES by sampling individuals as $p_t(x) = p_t(x|\sigma)p_t(\sigma)$ by sampling $\sigma_{t,i} \sim p_t(\sigma)$ and $x_{x,i} \sim p_t(x|\sigma_{t,i})$, for $i = 1, \ldots, \lambda$. We choose $p_t(\sigma)$ as the log-normal distribution with density

$$p_t(\sigma) = \frac{1}{\sigma\beta\sqrt{2\pi}}\exp\left(-\frac{(\log(\sigma)-\log(\sigma_t))^2}{2\beta}\right)$$

and $p_t(x|\sigma)$ as a normal distribution with mean m_t and standard deviation σ. Applying the same derivation as in the xNES SA on σ_t we arrive at the update

$$\sigma_{t+1} = \sigma_t^{1-c_\sigma}\exp\left(c_\sigma\sum_{i=1}^{\lambda} w_i \log(\sigma_{t,i})\right)$$

with learning rate c_σ. The update can be interpreted as a geometric update of the old step size using the weighted geometric mean of the current iteration. The hyperparameters are the learning rate c_σ and the variance parameter β of the log-normal distribution. This update is close to the update rule of the CMSA [3] with $c_\sigma = 1$ and exchanging the geometric average of $\sigma_{t,i}$ by an arithmetic average. The geometric average leads to an unbiased estimate of the mean on a flat or random function and therefore to random walk behaviour.

The xNES SA and the mean-xNES SA are not invariant w.r.t. the addition of constant dimensions due to the variability of $x_{t,i}$ in these dimensions. This defect is fixed by the introduction of a prior, because the step size $\sigma_{t,i}$ is independent of the actual step.

4.5 Median Rule

The Median Rule was proposed in [1] as a way to adapt Rechenberg's 1/5-rule [16] to non-elitist algorithms. The idea is to compare the objective function values $f(x_{t,1}) \leq \cdots \leq f(x_{t,\lambda})$ of the current iteration to a chosen quantile of the function values $f(x_{t-1,1}) \leq \cdots \leq f(x_{t-1,\lambda})$ of the previous iteration. Let κ be the rank of the individual in the previous generation that represents the threshold $f(x_{t-1,\kappa})$. Then we estimate the probability that a newly sampled point has a smaller function value as

$$u_t = \frac{1}{\lambda} \sum_{i=1}^{\lambda} \mathbb{1} \left\{ f(x_{t,i}) \leq f(x_{t-1,\kappa}) \right\} \quad .$$

The rank κ is a hyperparameter. It is chosen so that with an optimal step size the probability that a currently sampled point has a smaller function value than $f(x_{t-1,\kappa})$ is 1/2 on the Sphere function. Normalizing the values of u_t to $[-1, 1]$ so that a success rate of 1/2 transforms to 0 we define the time averaged statistic

$$z_{t+1} = (1 - c_z) z_t + c_z (2u_t - 1)$$

and σ_t is updated in an exponential fashion according to

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{z_{t+1}}{d_\sigma} \right) \quad ,$$

where d_σ is a damping factor controlling the speed of adaptation.

4.6 Population SA

This algorithm has recently been introduced as part of the Limited Memory CMA-ES [12] as an improvement to the Median-Rule, motivated by the need to overcome the runtime complexity problems of CSA. This success-based rule increases the step size when the new samples are more successful than the previous ones. The measure of success is based on a Whitney-U rank sum statistic.

Let $r_{t,i}$ be the rank of the point $x_{t,i}$ and $o_{t,i}$ the rank of the individual $x_{t-1,i}$ in the combined set $\big\{ f(x_{t-1,1}), \ldots, f(x_{t-1,\lambda}), f(x_{t,1}), \ldots, f(x_{t,\lambda}) \big\}$. We define the rank-sum

$$u_t = \frac{1}{\lambda^2} \sum_{i=1}^{\lambda} (o_{t,i} - r_{t,i}) \in [-1, 1] \ .$$

When $u_t > 0$ then the current population is more successful than the previous one. A target success rate $b > 0$ is subtracted and the statistic is time averaged by $z_{t+1} = (1 - c_z) z_t + c_z (u_t - b)$. The update of σ_t is then again performed in an exponential fashion

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{z_{t+1}}{d_\sigma} \right)$$

with damping factor d_σ.

The bias b serves two roles: It corrects for the fact that any successful step of the mean m_t will improve the sampled points compared to the previous iterations and thus setting $b = 0$ would increase the step size until no progress can be seen any more. The second role is to force the algorithm to shrink its step size once the progress seen is smaller than the expected value.

Median Rule and Population SA are perfectly invariant w.r.t. the addition of constant dimensions. This is because they rely solely on function values $f(x_{t,i})$, not on the underlying vectors $x_{t,i}$, where random fluctuation in these components would show. On the other hand, they rely on the assumption that the optimal step size is coupled to a fixed success rate. This assumption is fragile, as the optimal success rate is problem independent. For example, on an ill-conditioned ellipsoid, the success rate will be very small unless individuals are sampled with small variance. Thus a fixed target success rate might entail small steps.

4.7 Two-point Step Size Adaptation (TPA)

The Two-point Step Size Adaptation (TPA) is one of the most simple adaptation rules. It was introduced in [17], adopted in [6] and reformulated in [7]. Its idea can be viewed as a simplified line-search approach. The version we use here is closer to [6], as [7] is incompatible with the ES defined in section 2. We further deviate from [6] by using the step size proposal from [17] since it showed better results in our experiments. Further improvements can be obtained by swapping the order of updating σ_t and x_t in the ES, which is more in line with the line-search motivation and closer to the approach in [17].

After a step in direction s_t is performed with the current step size σ we are at the point $m_{t+1} = m_t + \sigma_t s_t$. We want to know whether it would have been beneficial to have taken a longer or shorter step. In a non-black-box method we could check the sign of the derivative $d = s_t^T \frac{\partial}{\partial m_{t+1}} f(m_{t+1})$. A positive (negative) value indicates that a shorter (longer) step would have made more progress. Hence we should decrease (increase) the step size.

In a black-box setting we cannot compute derivatives, but we can estimate the sign numerically simply by comparing two points created by making the step shorter and longer. We choose to decrease the step size by a factor $\alpha < 1$ and increase it by a factor $\beta > 1$, evaluate $f_{t,\alpha} = f(x_t + \alpha \sigma s_t)$ and $f_{t,\beta} = f(x_t + \beta \sigma s_t)$, and estimate the sign as $\operatorname{sign}(d) = \operatorname{sign}(f_{t,\beta} - f_{t,\alpha})$. We compute the time-average

$$\begin{aligned} z_{t+1} = (1 - c_z) z_t &+ c_z \log(\alpha) \mathbb{1}\{f_{t,\alpha} < f_{t,\beta}\} \\ &+ c_z \log(\beta) \mathbb{1}\{f_{t,\alpha} \geq f_{t,\beta}\} \end{aligned}$$

and finally perform the exponential update

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{z_{t+1}}{d_\sigma} \right) \quad .$$

Analog to [17] we choose $\beta = \frac{1}{\alpha}$ in our experiments. This amounts to increasing or decreasing $\log(\sigma)$ by the same value $|\log(\alpha)| = |\log(\beta)|$.

Table 2: Theoretical and empirical properties of the studied algorithms as described in section 3.

Algorithm	Mean Progress	Fixed Point	Const. Dim.
Cumulative SA	yes	yes	no
Median SA	no	no	yes
Population SA	no	no	yes
xNES SA	no	no	no
prior-xNES SA	no	no	yes
mean-xNES SA	yes	yes	no
Two-point SA	yes	yes	yes

The two-point method is completely invariant under the addition of constant dimensions.

In this study, we considered the original variant of two-point step size adaptation, which uses two additional function evaluations per generation. This makes it the only method in our comparison that requires additional evaluations of the objective function. However, newer variants of two-point step size adaptation do not suffer from this drawback [7]. If the step size adaptation is delayed by one generation, the individuals used for the two-point method can be part of the next offspring population, for details and further enhancements of the two-point method we refer to [7].

5. EXPERIMENTS

While properties like invariance under the addition of constant dimensions (and the need for additional function evaluations) can be determined from the algorithm description, this is not (always) the case for the progress of the mean and steady state stability. Therefore these properties are investigated experimentally in the following.

For all our experiments we utilize the family of quadratic functions

$$f_y(x) = \sum_{i=1}^{d} y_i x_i^2 \quad .$$

This family of functions has the property of scale invariance, i.e., all (non-optimal) level sets share the same shape up to scaling. For the quadratic functions above this is ensured by the relation $f(ax) = a^2 f(x)$. For any given point x, setting $a = \frac{1}{\sqrt{f(x)}}$, we can always "normalize" the algorithm state to the unit level set $f(ax) = 1$.

A special case of this family is the Sphere function $f_{\mathrm{s}}(x) = \|x\|^2$, which is distinguished in this study for exhibiting the same symmetries as the Gaussian search distribution. In addition we consider the Ellipsoid

$$f_{\mathrm{ell},k}(x) = \sum_{i=1}^{d} k^{i/d} x_i^2$$

and the sphere function with $d - k$ constant dimensions ("Constant Sphere")

$$f_{\mathrm{cs}}(x) = \sum_{i=1}^{k} x_i^2 \quad .$$

We set $k = 4$ in all of our experiments. The ill-conditioned Ellipsoid problem is well known to be difficult to solve with isotropic search distributions, which poses a challenge to SA methods. The Constant Sphere function should be no harder to optimize than the Sphere function, provided that the SA method is invariant under the addition of constant dimensions.

Well-tuned default values are available for the various SA methods, usually from the papers where they were originally proposed. However, tuning can be performed aggressively or conservatively, on different benchmark suits and problem dimensions, and with different goals. The resulting optimization performance can differ significantly [13].

Therefore, in order to remove possible biases and to make our comparison fair, we first tuned all algorithms with the same setup. All parameters were tuned on the Sphere function at varying dimensionality for optimal performance. In many cases, the parameters found were close to those published in earlier literature, often only differing by a constant factor which lead to less conservative, larger learning rates. In subsequent experiments the algorithms are analyzed with the pre-tuned parameters. The parameters we found are summarized in Table 3. The experiments can be reproduced with the source code provided as supplementary material.

Experiment 1: Optimization Performance.

In this experiment we measured the performance of all tuned algorithms on the Sphere function. We varied the dimensionality $d \in \{4, 8, 16, 32, 64, 128\}$ and measured the number of function evaluations until reaching the target objective value of 10^{-14}. We report median and lower and upper quantiles over 100 trials.

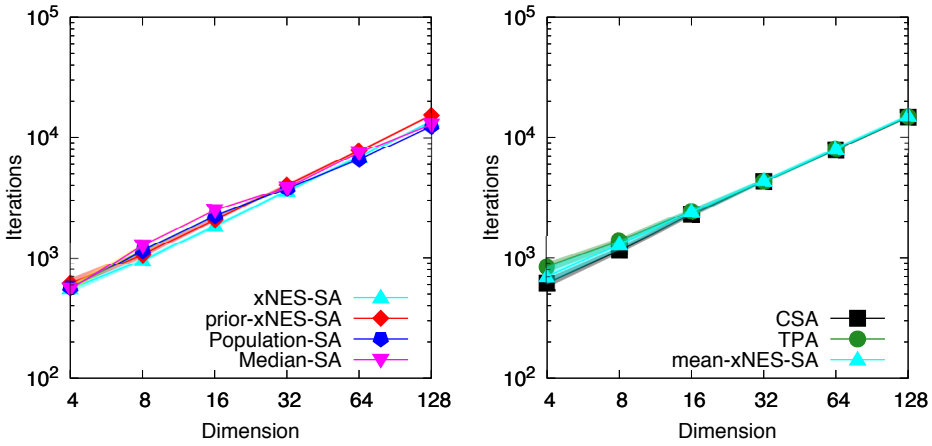
We repeated this experiment on the Constant Sphere function to asses how the algorithms are affected by meaningless variables. We performed this experiment in two variants: firstly, we set the learning rates exactly as on the Sphere function with ambient dimension d, and secondly, we set the values as for dimension $d = 4$, which makes the results comparable to the underlying 4-dimensional Sphere problem. In the latter experiment, an algorithm that is completely invariant to constant dimensions must show identical performance.

Experiment 2: Distribution of Step Sizes.

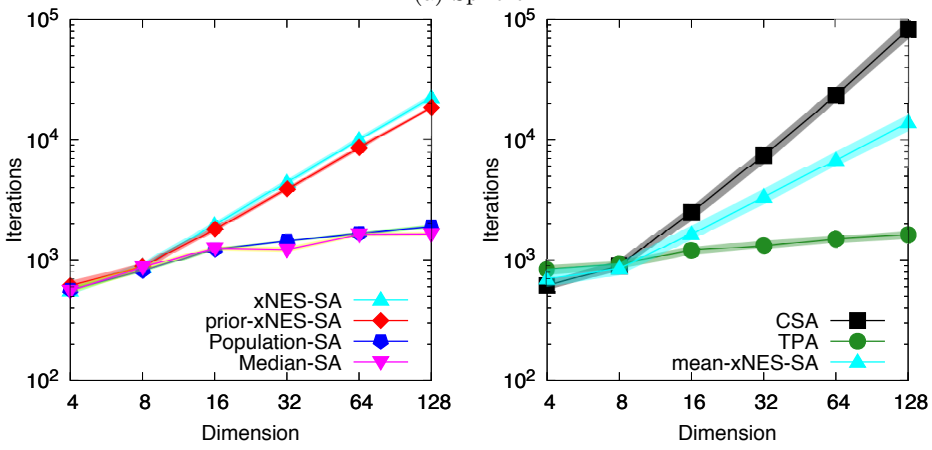
In our second experiment we compare the realized (normalized) step sizes of the different algorithms to the optimal step size. This provides first insights into why an algorithm performs well or not. Scale invariance of the objective function as well as the algorithms under consideration implies that the algorithm state given by m_t, σ_t and other state variables such as evolution paths and stored function values from previous iterations, normalized by the scaling factor $\frac{1}{\sqrt{f(m_t)}}$, forms a Markov chain, the stationary distribution of which gives rise to its long-term behavior [4]. Hence the normalized step size $\sigma_t / \sqrt{f(m_t)}$ can be estimated by the mean or median over a large number of iterations. Numerically more stable results are achieved by instead re-normalizing the algorithm state after each iteration.

In all experiments we first perform $50,000$ iterations to let the algorithm converge to its long term behavior and then compute the median normalized step size over the next $50,000$ iterations. We again analyze problem dimensions $d \in \{4, 8, 16, 32, 64, 128\}$ and report the results on Ellipsoid functions of various difficulties with $k \in \{1, 10, 100\}$.

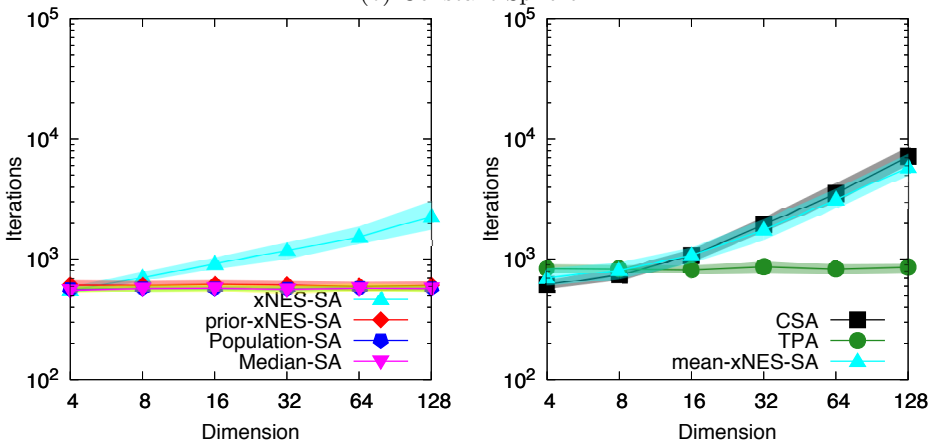
This experiment reveals the realized normalized step sizes



(a) Sphere



(b) Constant Sphere



(c) Constant Sphere, $d = 4$ rates

Figure 1: Number of iterations needed by the different algorithms to reach the target threshold at varying dimensionality. From top to bottom: Sphere, Sphere with $d-4$ constant dimensions, and Sphere with $d-4$ constant dimensions and learning rates of $d = 4$. The shaded areas depict the 25% and 75% percentile. Note that due to the tuning all curves are largely overlapping on Sphere.

while approaching the optimum. On scale invariant functions, the realized step sizes will on average overestimate the value obtained at the fixed point, as the update step of σ_t will update it in direction of the *current* true unnormalized step size, while with every successful step the true unnormalized step size will shrink. To obtain the fixed point, we

perform the same re-normalization as in the previous experiment, but do not re-normalize σ_t. As the renomalization normalizes to the $f(m_t) = 1$ iso-surface, this allows the algorithm to let its value of σ_t converge to the value of the normalized step size at its fixed-point.

We estimated the optimal normalized step size by performing a grid search over $\sigma/\sqrt{f(m)}$ for maximal convergence rate. We used a two stage grid (coarse to fine), first obtaining the best $\hat{\sigma}_0^* \in \{10^{-3+\frac{3}{20}i} \mid i = 1, \ldots, 20\}$ and then finding the best $\sigma_0^* \in \{\hat{\sigma}_0^* 10^{-\frac{1}{5}+\frac{2}{5}\frac{i}{30}} \mid i = 1, \ldots, 30\}$.

6. RESULTS AND DISCUSSION

When interpreting the experimental results we are primarily interested in identifying sub-optimal behavior, and we focus in particular on how and why different SA mechanisms break. We intend to provide insights and potential starting points for improving SA mechanisms in the future.

The results of the first experiment are illustrated in Figure 1. On Sphere (Figure 1, left) all algorithms performed essentially the same. This is of course an artifact of tuning on this very function. More interestingly, on Sphere with constant dimensions (Figure 1, middle) we see large differences between algorithms. While xNES SA, prior-xNES SA, and mean-xNES SA performed the same as on the Sphere function, CSA required more iterations, while the remaining algorithms required less. When setting learning rates and population size to the values of the four-dimensional Sphere problem (Figure 1, right), we saw that CSA performance improved to the level of mean-xNES SA and xNES SA, while for the remaining algorithms the number of iterations became independent of the ambient problem dimension.

These results can be explained as follows. CSA, xNES SA, and mean-xNES SA compare lengths of vectors (either steps or a path) to the expected length under random selection. In this case only four dimensions are relevant for selection, hence the ES performs a random walk in the orthogonal subspace. The squared norm of the vectors in the orthogonal subspace follows the χ^2_{d-4} distribution with variance $2(d-4)$. Thus, to keep the algorithms stable, the learning rate must be scaled to reduce the overall variance in the update of the step size. This is achieved by the factor $1/d$ or $1/\sqrt{d}$ in the update rules of the algorithms, which slows down learning of the meaningful dimensions for $d \gg 4$.

The results of the second experiment are given in Figure 2. Since all algorithms were tuned on Sphere ($k = 1$, left column), the realized step sizes were close to optimal. Still, the fixed-point step sizes of xNES SA and prior-xNES SA were underestimating the optimal step size.

On the Ellipsoid problem, the gap between the optimal and the realized step size increased with increasing conditioning k. For xNES SA and prior-xNES SA the gap increased further as they converged to their fixed-point. Also, the fixed points of Population SA and Median SA were clearly inferior to CSA and TPA, where CSA was superior to TPA for large conditioning numbers k. We observe that for large k algorithms relying on a statistic on the mean (CSA, TPA and mean-xNES SA) outperformed xNES SA relying on the length of individual steps and the population-based variants relying purely on ranking information.

The biggest difference between xNES SA and prior-xNES SA on the one hand and CSA and TPA on the other hand is that the former are based on distances of sampled points from the mean, while the latter consider the lengths of steps of the mean. This is an important difference, as it is improbable to create a sample in the optimal direction, while the mean of the selected samples is a much more stable estimate of this direction. However, when the direction is not optimal, then the optimal step size in this direction will very likely be smaller than in the optimal direction, which holds especially true on Ellipsoid with large k. This results in a bias towards smaller step sizes, which is pronounced for methods relying on properties of selected individuals.

The population-based algorithms Median-Rule and Population SA suffer from their definition of success. With increasing k the probability to draw a point that is better than the current median decreases. Thus the distributions of ranks overlap more and more. Therefore, for optimal performance, with growing k the success rates or percentiles would need to be adapted to smaller values.

The algorithms CSA, TPA, and mean-xNES SA suffered the least losses in performance on the Ellipsoid. Although for $k = 10$ they were close to the optimal learning rate, there was still a relevant loss in performance. This is because the Euclidean length of a step gives less information than on Sphere about how much progress was made. For CSA there is another problem. Samples are selected more aggressively in some directions than in others, thus the distribution of the observed noise is not as expected and the noise level of uncorrelated steps changes. TPA suffers from a similar problem because it adapts the step size based on the expected success probability of a larger or smaller step.

Table 3: Parameters used by the algorithms in the experiments after tuning on Sphere.

constant	value
	CSA
c_p	$\frac{\mu_{\text{eff}}+2}{d+\mu_{\text{eff}}+5}$
d_σ	$\frac{1}{4}\left(1 + c_p + 2\max\left\{0, \sqrt{\frac{\mu_{\text{eff}}-1}{d+1}} - 1\right\}\right)$
	Median-Rule
κ	$\left\lfloor 1 + \frac{\mu_{\text{eff}}}{\lambda}\frac{1}{d} \right\rfloor$
c_z	0.4
d_σ	1
	Population SA
c_z	0.4
b	0.4
d_σ	1
	xNES SA
c_σ	$\frac{\mu_{\text{eff}}}{2\log(d)\sqrt{d}}$
	mean-xNES SA
c_σ	1
	prior-xNES SA
β	$\frac{\log 2}{\sqrt{d}\log(d)}$
c_σ	$\frac{9\mu_{\text{eff}}}{10\sqrt{d}}$
	Two-point SA
α	0.7
c_z	0.5
d_σ	1

This leads to a more conservative choice of the step size in cases where the probability of success is small but the gain of a successful step is large.

7. CONCLUSION

We have analyzed the behavior of a set of step size adaptation (SA) algorithms based on a predefined set of requirements. Some of the requirements can be checked based on obvious properties of the SA methods. Others required an empirical investigation, which could be limited to a small set of easy to analyze benchmark functions found in many of the current benchmark sets.

We focus on the following aspects: Is the algorithm invariant to the addition of constant dimensions or does its performance suffer? Does the normalized step size converge to a reasonable fixed-point on scale invariant functions? And is this fixed point close to optimal across different objective functions?

We argue that an algorithm can only be competitive on a wide range of black-box functions if it fulfills (at least) these requirements, since otherwise its performance will depend crucially on the tuning of its parameters.

It turns out that only one of the analyzed algorithms, the two-point adaptation method TPA, fulfills all these requirements. This makes the new variant of TPA, which does not require additional objective function evaluations, an interesting alternative to CSA in the CMA-ES [7], also because it should work seamlessly with the novel efficient covariance matrix update scheme proposed in [11, 10].

We show that algorithms that are based on information geometric optimization (natural gradients) under-estimate the step size, an effect that gets even worse for ill-conditioned problems. This explains the conservatively tuned learning rates of the xNES algorithm, which has significantly smaller default learning rates than all of its competitors. We argue that this is the case because most offspring are not sampled in the direction towards the optimum, but instead more and more orthogonal with growing problem dimension. Thus the step size adaptation mechanisms take steps into account which the overall algorithm does not take. As in these directions successful steps are typically smaller, this results in a too small step size. We showed that this is indeed the case by implementing a new algorithm that performs the update using the actual step taken. This leads to a competitive algorithm, which is however not derived from information geometry, but instead closely resembles CSA without temporal integration.

While present SA methods work quite satisfactory in many situations, it is still easy to construct problems resulting in far from optimal step sizes. In this paper we have systematically investigated the weak spots of the underlying adaptation mechanisms. The currently available algorithms differ quite widely in their working principles and also in their relative strengths and weaknesses. We therefore believe that there is room for developing techniques that combine the strengths of all present SA algorithms.

Acknowledgements

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

8. REFERENCES

[1] O. Ait Elhara, A. Auger, and N. Hansen. A median success rule for non-elitist evolution strategies: Study of feasibility. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 415–422. ACM, 2013.

[2] 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.

[3] H.-G. Beyer and B. Sendhoff. Covariance matrix adaptation revisited–the CMSA evolution strategy. In *Parallel Problem Solving from Nature (PPSN)*, pages 123–132. Springer, 2008.

[4] A. Chotard, A. Auger, and N. Hansen. Markov chain analysis of evolution strategies on a linear constraint optimization problem. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 159–166. IEEE, 2014.

[5] 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.

[6] N. Hansen. CMA-ES with two-point step-size adaptation. *arXiv N. arXiv:0805.0231*, 2008.

[7] N. Hansen, A. Atamna, and A. Auger. How to assess step-size adaptation mechanisms in randomised search. In *Parallel Problem Solving from Nature (PPSN)*, pages 60–69. Springer, 2014.

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

[9] C. Igel, T. Suttorp, and N. Hansen. A computational efficient covariance matrix update and a (1+1)-CMA for evolution strategies. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 453–460. ACM, 2006.

[10] O. Krause, D. R. Arbonès, and C. Igel. CMA-ES with optimal covariance update and storage complexity. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[11] 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 2015)*, pages 129–136. ACM Press, 2015.

[12] I. Loshchilov. A computationally efficient limited memory CMA-ES for large scale optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 397–404. ACM, 2014.

[13] I. Loshchilov, M. Schoenauer, M. Sebag, and N. Hansen. Maximum likelihood-based online adaptation of hyper-parameters in cma-es. In *Parallel Problem Solving from Nature (PPSN)*, pages 70–79. Springer, 2014.

[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.

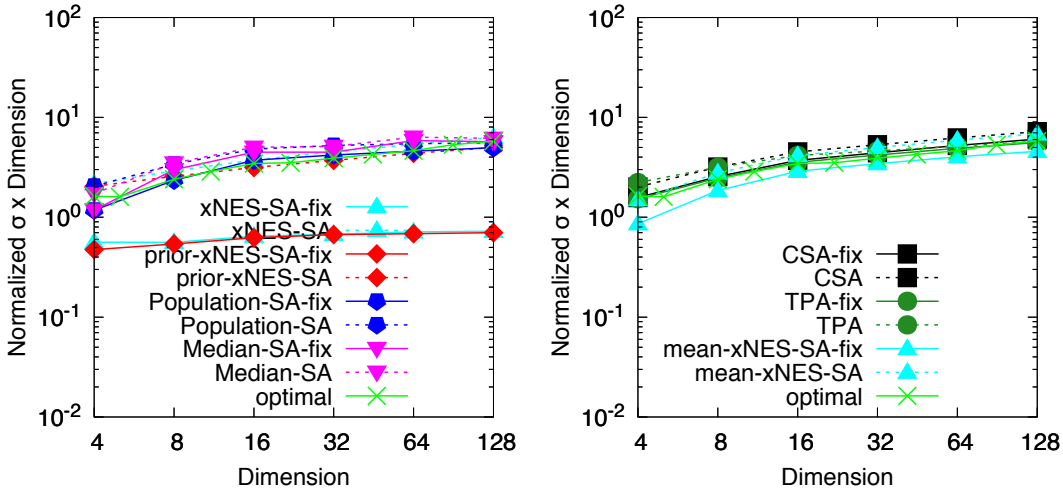
[15] A. Ostermeier, A. Gawelczyk, and N. Hansen. Step-size adaptation based on non-local use of

selection information. In *Parallel Problem Solving from Nature (PPSN)*, pages 189–198. Springer, 1994.

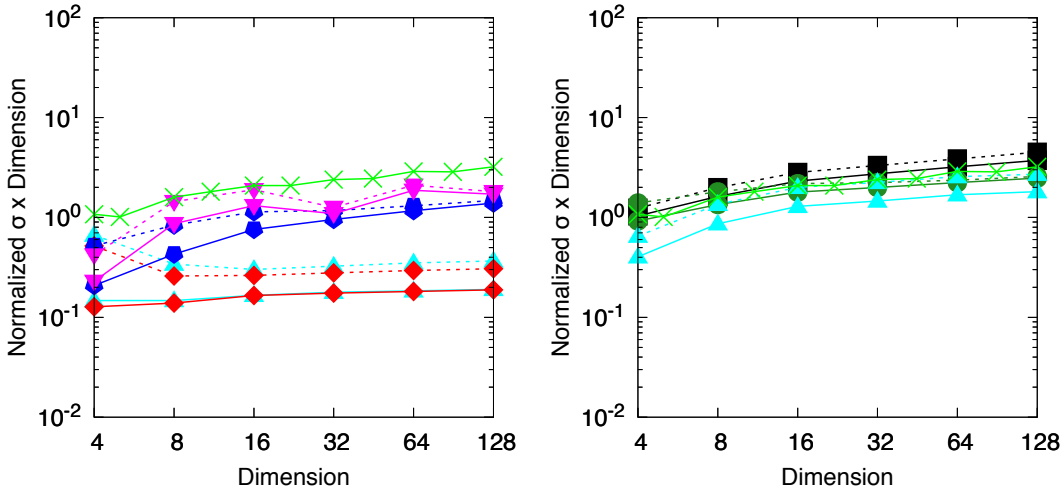
[16] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, 1973.

[17] R. Salomon. Evolutionary algorithms and gradient search: similarities and differences. *IEEE Transactions on Evolutionary Computation*, 2(2):45–55, 1998.

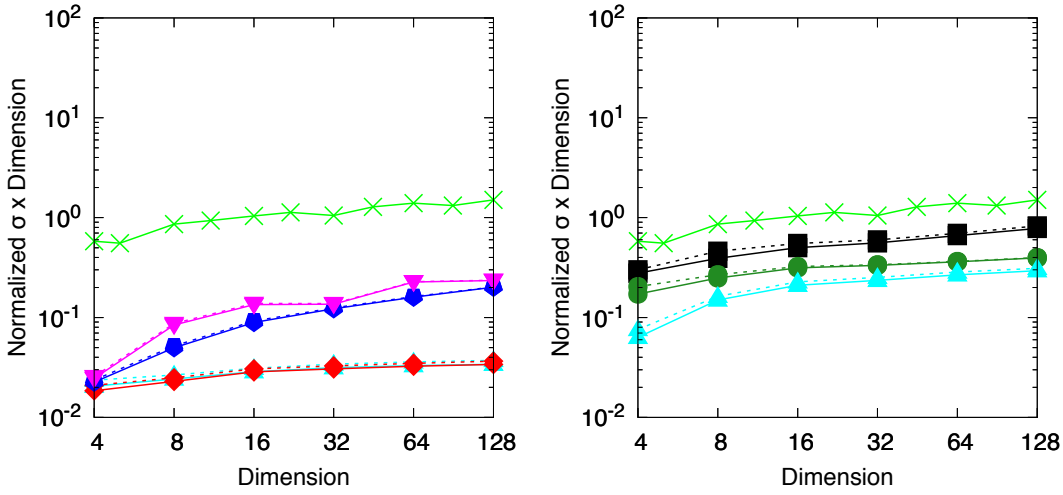
[18] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural Evolution Strategies. *Journal of Machine Learning Research*, 15:949–980, 2014.



(a) Sphere, $k = 1$



(b) $k = 10$



(c) $k = 100$

Figure 2: Normalized step sizes times dimensions obtained by the algorithms on the ellipsoid function, with growing difficulty $k \in \{1, 10, 100\}$ from top to bottom. The plots compare the progress of the fixed-point of the step size adaptation(dashed lines) and the progress of the step size of the optimization dynamics to the progress with optimal step size. Note that for large k fixed-point and normal dynamics are overlapping.