

Empirical investigation of simplified step-size control in metaheuristics with a view to theory

Jens Jägersküpper* & Mike Preuss

Technische Universität Dortmund, Fakultät für Informatik,
44221 Dortmund, Germany

Abstract. Randomized direct-search methods for the optimization of a function $f: \mathbb{R}^n \rightarrow \mathbb{R}$ given by a black box for f -evaluations are investigated. We consider the cumulative step-size adaptation (CSA) for the variance of multivariate zero-mean normal distributions. Those are commonly used to sample new candidate solutions within metaheuristics, in particular within the CMA Evolution Strategy (CMA-ES), a state-of-the-art direct-search method. Though the CMA-ES is very successful in practical optimization, its theoretical foundations are very limited because of the complex stochastic process it induces. To forward the theory on this successful method, we propose two simplifications of the CSA used within CMA-ES for step-size control. We show by experimental and statistical evaluation that they perform sufficiently similarly to the original CSA (in the considered scenario), so that a further theoretical analysis is in fact reasonable. Furthermore, we outline in detail a probabilistic/theoretical runtime analysis for one of the two CSA-derivatives.

1 Introduction

The driving force of this paper is the desire for a theoretical runtime analysis of a sophisticated stochastic optimization algorithm, namely the *covariance matrix adaptation evolution strategy (CMA-ES)*. As this algorithm is very hard to analyze theoretically because of the complex stochastic process it induces, we follow an unorthodox approach: We decompose it into its main algorithmic components: the *covariance matrix adaptation (CMA)*, and the *cumulative step-size adaptation (CSA)*. While the CMA is, in some sense, supposed to handle ill-conditionedness in the optimization problems, it is the duty of the CSA to cope with a challenge that every algorithm for real-valued black-box optimization faces: step-size control, i.e. the adaptation of step-sizes when approaching an optimum. The idea behind this decomposition is to substitute one of the components by an alternative mechanism that is more amenable to a theoretical analysis. While doing so, we rely on experimentation to assess how far we depart from the original algorithm. Here simpler mechanisms for step-size control are substituted for CSA. The desired outcome of this process is a modified

* supported by the German Research Foundation (DFG) through the collaborative research center “Computational Intelligence” (SFB 531)

algorithm that is both: tractable by theoretical analysis techniques, and, tested empirically, still working reasonably similar to the original one. At the same time we aim at better understanding the core-mechanisms of the original algorithm. (A simplified algorithm may also be interesting for practitioners, as simple methods often spread much faster than their complicated counterparts, even if the latter have slight performance advantages.) The full theoretical analysis itself, however, is not part of this work, but is still pending. Rather we discuss why a theoretical analysis seems feasible and outline in detail how it could be accomplished. Note that proving (global) convergence is *not* the subject of the theoretical investigations. The subject is a probabilistic analysis of the random variable corresponding to the number of steps necessary to obtain a predefined reduction of the approximation error. For such an analysis to make sense, the class of objective functions covered by the analysis must necessarily be rather restricted and simple. This paper provides evidence by experimental and statistical evaluation that a theoretical analysis for the simplified algorithm does make sense to yield more insight into CSA, the step-size control within the CMA-ES.

In a sense, our approach can be seen as algorithm re-engineering, a viewpoint which is to our knowledge uncommon in the field of metaheuristics. Therefore, we also strive for making a methodological contribution that will hopefully inspires other researchers to follow a similar path. To this end, it is not necessary to be familiar with the latest scientific work on metaheuristics to understand the concepts applied herein. It is one of the intrinsic properties of stochastic algorithms in general that even simple methods may display surprising behaviors that are not at all easy to understand and analyze. We explicitly focus on practically relevant dimensions. As it may be debatable what relevant is, we give a 3-fold categorization used in the following: Problems with up to 7 dimensions are seen as small, whereas medium sized ones from 8 to 63 dimensions are to our knowledge of the highest practical importance. Problems with 64 dimensions and beyond are termed large.

In the following section the original CSA is described in detail. The two new simplified CSA-derivatives are presented in Section 3, and their technical details and differences as well as some related work are discussed in Section 4. Furthermore, a detailed outline of a theoretical runtime analysis is given. The experimental comparison (including statistical evaluation) of the three CSA-variants is presented in Section 5. Finally, we conclude in Section 6.

2 Cumulative step-size adaptation (CSA)

The CMA-ES of Hansen and Ostermeier [1] is regarded as one of the most efficient modern stochastic direct-search methods for numerical black-box optimization, cf. the list of over 100 references to applications of CMA-ES compiled by Hansen [2]. Originally, CSA was designed for so-called $(1,\lambda)$ *Evolution Strategies*, in which 1 candidate solution is iteratively evolved. In each iteration, λ new search points are sampled, each independently in the same way, namely by adding a zero-mean multivariate normal distribution to the current candidate

solution. When CMA is *not* used, like here, each of the λ samples is generated by independently adding a zero-mean normal distribution with variance σ^2 to each of its n components. The best of the λ samples becomes the next candidate solution—irrespective of whether this best of λ amounts to an improvement or not (so-called *comma selection*; CSA does not work well for so-called *elitist selection* where the best sample becomes the next candidate solution only if it is at least as good as the current one). The idea behind CSA is as follows: Consecutive steps of an iterative direct-search method should be orthogonal. Therefore, one may recall that steps of steepest descent (a gradient method) with perfect line search (i.e., the truly best point on the line in gradient direction is chosen) are indeed orthogonal when a positive definite quadratic form is minimized. Within CSA the observation of mainly positively [negatively] correlated steps is taken as an indicator that the step-size is too small [resp. too large]. As a consequence, the standard deviation σ of the multivariate normal distribution is increased [resp. decreased]. Since the steps' directions are random, considering just the last two steps does not enable a smooth σ -adaptation because of the large variation of the random angle between two steps. Thus, in each iteration the so-called *evolution path* is considered, namely its length is compared with the length that would be expected when the steps were orthogonal. Essentially, the evolution path is a recent part of the trajectory of candidate solutions. Considering the complete trajectory is not the most appropriate choice, though. Rather, a certain amount of the recent history of the search is considered. In the original CSA as proposed by Hansen and Ostermeier, σ is adapted continuously, i.e. after each iteration. The deterministic update of the evolution path $p \in \mathbb{R}^n$ after the i th iteration works as follows:

$$p^{[i+1]} := (1 - c_\sigma) \cdot p^{[i]} + \sqrt{c_\sigma \cdot (2 - c_\sigma)} \cdot m^{[i]} / \sigma^{[i]} \quad (1)$$

where $m^{[i]} \in \mathbb{R}^n$ denotes the displacement (vector) of the i th step, and the fixed parameter $c_\sigma \in (0, 1)$ determines the weighting between the recent history of the optimization and its past within the evolution path. We use $c_\sigma := 1/\sqrt{n}$ as suggested by Hansen and Ostermeier [1]. Note that $m^{[i]}$ is one of λ vectors each of which was independently chosen according to a zero-mean multivariate normal distribution with standard deviation $\sigma^{[i]}$. The length of such a vector follows a scaled (by $\sigma^{[i]}$) χ -distribution. Initially, $p^{[0]}$ is chosen as the all-zero vector. The σ -update is done deterministically as follows:

$$\sigma^{[i+1]} := \sigma^{[i]} \cdot \exp\left(\frac{c_\sigma}{d_\sigma} \cdot \left(\frac{|p^{[i+1]}|}{\bar{\chi}} - 1\right)\right) \quad (2)$$

where the fixed parameter d_σ is called *damping factor*, and $\bar{\chi}$ denotes the expectation of the χ -distribution. Note that σ is kept unchanged if the length of the evolution path equals $\bar{\chi}$. We used $d_\sigma := 0.5$ because this leads to a better performance than $d_\sigma \in \{0.25, 1\}$ for the considered function scenario. Naturally, there is interdependence between d_σ and c_σ , and moreover, an optimal choice depends (among others) on the function to be optimized, of course.

3 Two simplified CSA-Derivatives

In this section we introduce two simplifications of the original CSA, created by subsequently departing further from the defining Equations (1) and (2). The first simplification (common to both CSA-derivatives) will be to partition the course of the optimization into phases of a fixed length in which σ is not changed. Such a partitioning of the process has turned out useful in former theoretical analyses, cf. [3,4] for instance. Thus, both variants use phases of a fixed length, after each of which σ is adapted—solely depending on what happened during that phase, respectively. Therefore, recall that $c_\sigma = 1/\sqrt{n}$ in the path update, cf. Eqn. (1). Since $(1 - 1/\sqrt{n})^i = 0.5$ for $i \asymp \sqrt{n} \cdot \ln 2$ (as n grows), the half-live of a step within the evolution path is roughly $0.5\sqrt{n}$ iterations for small dimensions and roughly $0.7\sqrt{n}$ for large n . For this reason we choose the *phase length* k as $\lceil \sqrt{n} \rceil$ a priori for the simplified versions to be described in the following. The second simplification to be introduced is as follows: Rather than comparing the length of the displacement of a phase with the expected length that would be observed if the steps in the phase were completely orthogonal, the actual correlations of the steps of a phase in terms of orthogonality are considered directly and aggregated into a criterion that we call *correlation balance*.

pCSA. The “p” stands for *phased*. The run of the ES is partitioned into phases lasting $k := \lceil \sqrt{n} \rceil$ steps, respectively. After each phase, the vector corresponding to the total movement (in the search space) in this phase is considered. The length of this displacement is compared to $\ell := \sqrt{k} \cdot \sigma \cdot \bar{\chi}$, where $\bar{\chi}$ is the expectation of the χ -distribution with n degrees of freedom. Note that ℓ equals the length of the diagonal of a k -dimensional cube with edges of length $\sigma \cdot \bar{\chi}$, and that $\sigma \cdot \bar{\chi}$ equals the expected step-length in the phase. Thus, if all k steps of a phase had the expected length, and if they were completely orthogonal, then the length of the displacement vector in such a phase would just equal ℓ . Depending on whether the displacement’s actual length is larger [or smaller] than ℓ , σ is considered as too small (because of positive correlation) [resp. as too large (because of negative correlation)]. Then σ is scaled up [resp. down] by a predefined scaling factor larger than one [resp. by the reciprocal of this factor].

CBA. “CBA” stands for *correlation-balance adaptation*. The optimization is again partitioned into phases each of which lasts $k := \lceil \sqrt{n} \rceil$ steps. After each phase, the k vectors that correspond to the k movements in the phase are considered. For each pair of these k vectors the correlation is calculated, so that we obtain $\binom{k}{2} = k(k-1)/2$ correlation values. If the majority of these values are positive [negative], then the σ used in the respective phase is considered as too small [resp. as too large]. Hence, σ is scaled up [resp. down] after the phase by some predefined factor larger than one [resp. by the reciprocal of this factor].

4 Related work, discussion, and a view to theory

Evolutionary algorithms for numerical optimization usually try to *learn* good step-sizes, which may be implemented by self-adaptation or success-based rules,

the most prominent of which may be the 1/5-rule to increase [decrease] step sizes if more [less] than 1/5 of the samples result in an improvement. This simple deterministic adaptation mechanism, which is due to Rechenberg [5] and Schwefel [6], has already been the subject of a probabilistic analysis of the (random) number of steps necessary to reduce the approximation error in the search space. The first results from this viewpoint of analyzing ES like “usual” randomized algorithms were obtained in [7] for the simplest quadratic function, namely $x \mapsto \sum_{i=1}^n x_i^2$, which is commonly called SPHERE. This analysis has been extended in [8] to quadratic forms with bounded condition number as well as to a certain class of ill-conditioned quadratic forms (parameterized in the dimensionality of the search space) for which the condition number grows as the dimensionality of the search space increases. The main result of the latter work is that the runtime (to halve the approximation error) increases proportionally with the condition number. This drawback has already been noticed before in practice, of course. As a remedy, the CMA was proposed which, in some sense, learns and continuously adjusts a preconditioner by adapting the covariance matrix of the multivariate normal distribution used to sample new candidate solutions. As noted above, within the CMA-ES the CSA is applied for step-size control, which is neither a self-adaptive mechanism nor based on a success rule. In the present work, we exclusively deal with this CSA mechanism. Therefore, we consider a spherically symmetric problem, so that the CMA mechanism is dispensable. We demand that the simplified CSA-derivatives perform sufficiently well on this elementary type of problem at least. If they did not, they would be unable to ensure local convergence at a reasonable speed, so that efforts on a theoretical analysis would seem questionable.

CSA, pCSA, and CBA are basically $(1, \lambda)$ ES, and results obtained following a dynamical-system approach indicate that the expected spatial gain towards the optimum in an iteration is bounded above by $O(\ln(\lambda) \cdot d/n)$, where d denotes the distance from the optimum, cf. [9, Sec. 3.2.3, Eqn. (3.115)]. As this result was obtained using simplifying equations to describe the dynamical system induced by the ES, however, it cannot be used as a basis for proving theorems on the runtime of $(1, \lambda)$ ES. Nevertheless, this is a strong indicator that a $(1, \lambda)$ ES converges, if at all, at best linearly at an expected rate $1 - O(\ln(\lambda)/n)$. From this, we can conjecture that the expected number of iterations necessary to halve the approximation error is bounded below by $\Omega(n/\ln \lambda)$. This lower bound has in fact been rigorously proved in [10, Thm. 1] for a framework of iterative methods that covers pCSA as well as CBA. (Unfortunately, CSA is not covered since the factor for the σ -update in Eqn. (2) depends on the course of the optimization, namely on the length of the evolution path.) Actually, it is proved that *less* than $0.64n/\ln(1+3\lambda)$ iterations suffice to halve the approximation error only with an exponentially small probability $e^{-\Omega(n)}$ (implying the $\Omega(n/\ln \lambda)$ -bound on the expected number of steps). Moreover, [10, Thm. 4] provides a rigorous proof that a $(1, \lambda)$ ES using a simple σ -adaptation mechanism (based on the relative frequency of improving steps) to minimize SPHERE needs with very high probability at most $O(n/\sqrt{\ln \lambda})$ iterations to halve the approximation error. This

upper bound is larger than the lower bound by a factor of order $\sqrt{\ln \lambda}$, which is actually due to the σ -adaptation: It adapts σ such that it is of order $\Theta(d/n)$, whereas $\sigma = \Theta(\sqrt{\ln \lambda} \cdot d/n)$ seems necessary. For the original CSA minimizing SPHERE, a result obtained in [11, Eqn. (20)] (using the dynamical-system approach again) indicates that the expected spatial gain in an iteration tends to $(\sqrt{2} - 1) \cdot (c_{1,\lambda})^2 \cdot d/n = \Theta(\ln(\lambda) \cdot d/n)$ as the dimensionality n goes to infinity, where $c_{1,\lambda} = \Theta(\sqrt{\ln \lambda})$ is the so-called $(1,\lambda)$ -progress coefficient (obtained for optimal σ , cf. [9, Eqn. (3.114)]); σ is reported to tend to $\sqrt{2} \cdot c_{1,\lambda} \cdot d/n$ as the dimensionality n goes to infinity, which is indeed $\Theta(\sqrt{\ln \lambda} \cdot d/n)$ [11, Eqn. (19)].

So, the long-term objective is a probabilistic analysis which rigorously proves that the $(1, \lambda)$ ES using CSA needs only $O(n/\ln \lambda)$ iterations to halve the approximation error (for SPHERE) with very high probability, which is optimal w.r.t. the asymptotic in n and λ . As this seems intractable at present because of the involved stochastic dependencies due to the evolution path, the intermediate objective may be to prove the $O(n/\ln \lambda)$ -bound for CBA. This makes (the most) sense if CBA behaves sufficiently similarly to the original CSA, of course. And this is what we investigate in the present paper. (If CBA performed much worse than CSA, however, we would have to reconsider whether it makes sense to try to prove the $O(n/\ln \lambda)$ -bound for CBA as it might just not adapt σ such that it is $\Theta(\sqrt{\ln \lambda} \cdot d/n)$.) The partition of the optimization process into phases in each of which σ is not changed, and after each of which σ is deterministically updated solely depending on what happened in that phase, enables the following line of reasoning in a formal proof:

If σ is too small at the beginning of a phase, i.e., $\sigma < c_1 \cdot \sqrt{\ln \lambda} \cdot d/n$ for an appropriately chosen constant $c_1 > 0$, then it is up-scaled after the phase with very high probability (w.v.h.p.). If, however, σ is too large at the beginning of a phase, i.e., $\sigma > c_2 \cdot \sqrt{\ln \lambda} \cdot d/n$ for another appropriately chosen constant $c_2 > c_1$, then it is down-scaled after the phase w.v.h.p. With these two lemmas, we can obtain that $\sigma = \Theta(\sqrt{\ln \lambda} \cdot d/n)$ for any polynomial number of steps w.v.h.p. once σ is of that order. Subsequently, we show that, if $\sigma = \Theta(\sqrt{\ln \lambda} \cdot d/n)$ in a step, then the actual spatial gain towards the optimum in this step is $\Omega(\ln \lambda \cdot d/n)$ with probability $\Omega(1)$; this can be proved analogously to [10, Sec. 5]. Thus, given that at the beginning of a phase (with \sqrt{n} steps) $\sigma = \Theta(\sqrt{\ln \lambda} \cdot d/n)$, the expected number of steps in the phase each of which actually reduces the approximation error by at least an $\Omega(\ln \lambda/n)$ -fraction is $\Omega(\sqrt{n})$. Using Chernoff's bound, in a phase there are $\Omega(\sqrt{n})$ such steps w.v.h.p., so that a phase reduces the approximation error at least by an $\Omega(\ln \lambda/\sqrt{n})$ -fraction w.v.h.p. Finally, this implies that $O(\sqrt{n}/\ln \lambda)$ phases, i.e. $O(n/\ln \lambda)$ steps, suffice to reduce the approximation error by a constant fraction w.v.h.p. This directly implies the $O(n/\ln \lambda)$ -bound on the number of iterations to halve the approximation error.

The proof of that CBA ensures $\sigma = \Theta(\sqrt{\ln \lambda} \cdot d/n)$ (w.v.h.p. for any polynomial number of steps) remains. Therefore, note that the probability that two steps are exactly orthogonal is zero (because of the random directions). Thus, in a phase of k steps the number of positively correlated pairs of steps equals (with probability one) $\binom{k}{2}$ minus the number of negatively correlated pairs. For

a theoretical analysis of CBA, for each phase $\binom{k}{2}$ 0-1-variables can be defined. Each of these indicator variables tells us whether the respective pair of steps is positively correlated (“1”) or not (“0”). (Recall that in CBA the decision whether to up- or down-scale σ is based on whether the sum of these indicator variables is larger than $\binom{k}{2}/2$ or smaller.) There are strong bounds on the deviation of the actual sum of 0-1-variables from the expected sum, in particular when the variables are independent—which is not the case for CBA. This can be overcome by stochastic dominance arguments, so that we deal with independent Bernoulli trials, rather than with dependent Poisson trials. We merely need to know how the success probability of a Bernoulli trial depends on σ .

All in all, CBA is a candidate for a theoretical runtime analysis of an ES using this simplified CSA-variant. It was an open question, whether CBA works at all and, if so, how well it performs compared to the original CSA (and pCSA). Thus, we decided for an experimental comparison with statistical evaluation.

5 Experimental investigation of the CSA-variants

Since the underlying assumption in theory on black-box optimization is that the evaluation of the function f to be optimized is by far the most expensive operation, the number of f -evaluations (λ times the iterations) is the sole performance measure in the following comparison of the three CSA-variants. To find out the potentials of the σ -adaptation mechanisms described above, we focus on the simplest unimodal function scenario, namely the minimization of the distance from a fixed point. This is equivalent (here) to the minimization of a perfectly (pre)conditioned positive definite quadratic form. One of these functions, namely $x \mapsto \sum_{i=1}^n x_i^2$, is the commonly investigated SPHERE (level sets of such functions form hyper-spheres). We decided for a (1,5) Evolution Strategy, i.e. $\lambda := 5$. The reason for this choice is that, according to Beyer [9, p. 73], five samples are most “effective” for comma selection, i.e. allow maximum progress per function evaluation. Thus, differences in the adaptations’ abilities (to choose σ as close to optimal as possible) should be most noticeable for this choice.

Experiment: Do the CSA-derivatives perform similar to the original CSA?

Pre-experimental planning. In addition to the adaptation rule and the phase length, the scaling factor by which σ is increased or decreased after a phase had to be fixed for pCSA and CBA. For both CSA-derivatives, the σ -scaling factor was determined by parameter scans, cf. Figure 1, whereas the phase length k was chosen a priori as $\lceil \sqrt{n} \rceil$ (for the reason given above). Interestingly, the parameter scans show that the σ -scaling factor should be chosen identically as $1 + 1/n^{1/4}$ for pCSA as well as for CBA.

Task. The hypothesis is that the three CSA-variants perform equally well in terms of number of iterations. As the data can not be supposed to be normally distributed, we compare two variants, namely their runtimes (i.e. number of iterations), by the Wilcoxon rank-sum test (as implemented by “wilcox.test” in “R”), where a p-value of 0.05 or less indicates a *significant* difference.

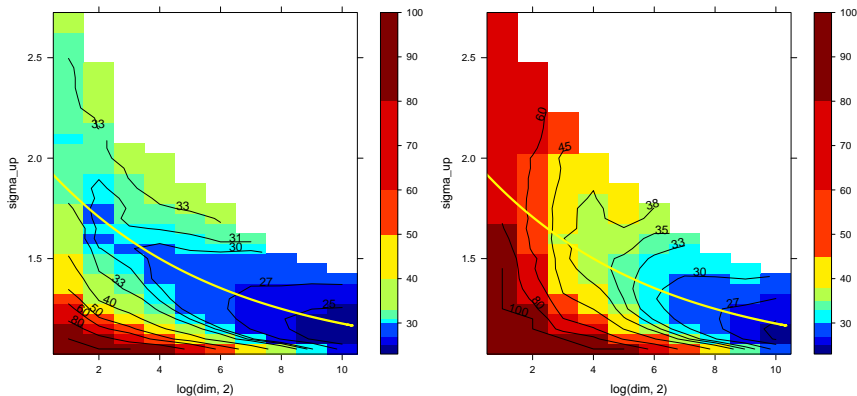


Fig. 1: Parameter scan of σ -scaling factors for pCSA (left) and CBA (right) over 2 to 1024 dimensions. The shading corresponds to the median number of steps divided by dimensionality; 101 runs in the same setting as the final experiment. The solid line represents $1 + 1/n^{1/4}$, whereas the thin lines show the contour.

Setup. The initial distance from the optimum is 2^{20} and the stopping criterion is a distance of less than 1 from the optimum, i.e., we measure the number of iterations to halve the approximation error in the search space 20 times. The initial σ is set to $2^{20} \cdot 1.225/n$ in each case. We investigate ten search-space dimensions, namely $n = 2^i$ for $i \in \{1, 2, \dots, 10\}$. Each of the three algorithms is run 1001 times.

Results. Figure 2 shows median runtimes and hinges for the three algorithms. Additionally, the σ -adaptation within typical runs (i.e. median runtime) is shown in Figure 3. Note that σ is considered well-chosen (after normalization w.r.t. dimension and distance from the optimum) when it lies between the two horizontal lines, which correspond to a normalized σ of 1.0 (blue) resp. 2.0 (red).

Observations. Independently of the search-space dimension, i.e. for *all* ten dimensions investigated, namely dimension $n = 2^i$ for $i \in \{1, \dots, 10\}$, we find:

1. The original CSA performs significantly better than pCSA and CBA.
2. Moreover, CBA performs significantly worse than pCSA.

As can be seen in Figure 2, transferring the continuous CSA-mechanism to a phased one, namely to pCSA, leads to an increase in the runtimes by clearly less than 50% in all ten dimensions. Concerning the runtimes at least, pCSA is closer to the original CSA than CBA.

Discussion. Despite the reported findings, we attest that CBA does not fail, it ensures a reliable σ -adaptation—it is merely worse. In particular, we are interested in how much worse CBA is compared to the original CSA. Actually,

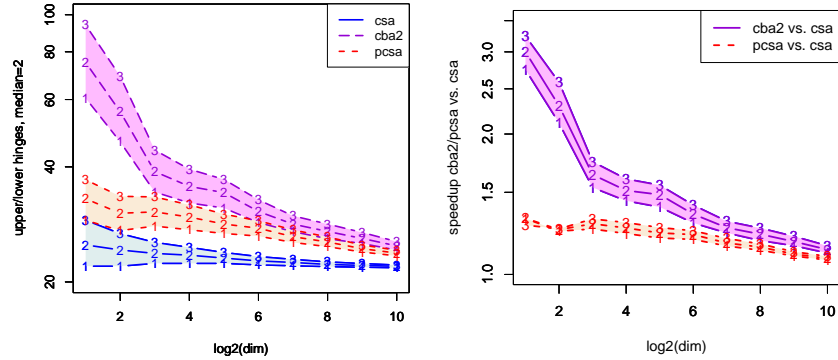


Fig. 2: Left: Number of steps divided by dimensionality for CBA, pCSA, CSA, where (1) lower hinge, (2) median, (3) upper hinge of 1001 runs, respectively. Right: Ratio between (1) lower hinge, (2) median, (3) upper hinge of the 1001 runtimes of CBA vs. CSA and pCSA vs. CSA

in 2-dimensional search space, the ratio between the medians of the runtimes of CBA to CSA equals 3. And in fact, when we multiply each of the 1001 runtimes of CSA by 3, then the Wilcoxon test does *not* tell us a significant difference between CSA and CBA (p-value 0.85). Thus, we may conclude that CBA is slower than CSA by roughly a factor of 3. However, “roughly” may be considered too vague. Rather, a confidence interval should be given: For the factor 2.91 as well as for the factor 3.1, the p-value drops below 2.5%, respectively. Thus, the 95%-confidence interval for the factor by which CBA is slower than CSA for the 2-dimensional problem is (at most) [2.91, 3.1], which has a spread of only $3.1/2.91 < 1.066$. The respective confidence intervals for all ten dimensions investigated are given in Table 1. As one might expect, they are pretty much concentrated around the ratio of the median runtimes, respectively.

Concerning the performance differences observed, we think that CBA—as the only variant for which a theoretical analysis seems feasible at present—still works sufficiently similar to CSA to legitimate such analysis, at least for the practically relevant and large dimensions: For eight and more dimensions the number of function evaluations (five times the iterations) that CBA spends is

CBA vs. CSA	2D	4D	8D	16D	32D	64D	128D	256D	512D	1024D
conf. interval	2.91–	2.25–	1.61–	1.49–	1.46–	1.33–	1.25–	1.219–	1.171–	1.127–
runtime ratio	3.10	2.37	1.67	1.54	1.50	1.36	1.27	1.226	1.182	1.134
spread	<6.6%	<5.4%	<3.8%	<3.4%	<2.8%	<2.2%	1.6%	<0.6%	<1%	<0.7%

Table 1: Confidence intervals for the ratios between the runtimes of CBA and CSA

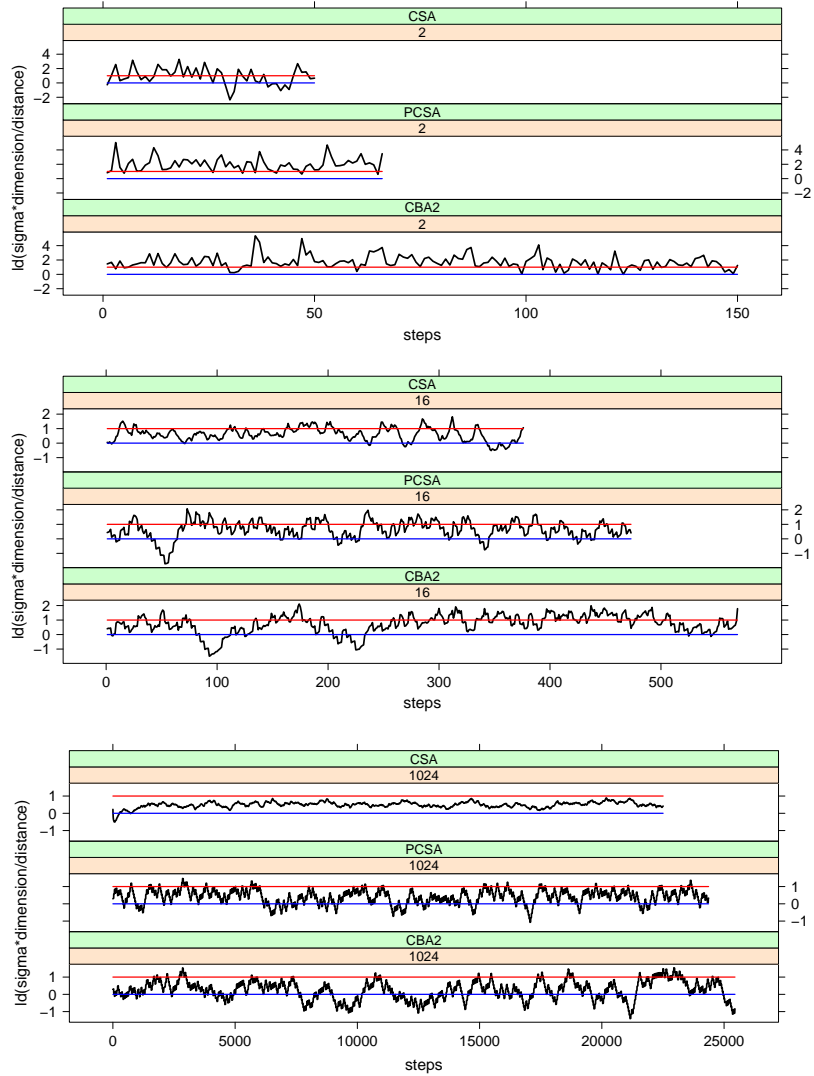


Fig. 3: Normalized σ (w.r.t. dimensionality and distance from optimum) in a typical run (i.e. median runtime) of each variant on 2-/16-/1024-dimensional SPHERE

mean±std of $\log_2(\hat{\sigma})$	2D	16D	1024D
CSA	0.914 ± 1.183 , 1.884	0.615 ± 0.454 , 1.370	0.492 ± 0.171 , 1.406
pCSA	2.012 ± 0.947 , 4.033	0.642 ± 0.617 , 1.560	0.408 ± 0.417 , 1.327
CBA	1.699 ± 0.923 , 3.247	0.734 ± 0.649 , 1.663	0.236 ± 0.549 , 1.178

Table 2: Means and standard deviations of $\log_2(\hat{\sigma})$ in the typical runs shown in Figure 3. Additionally, the **log-mean** $\hat{\sigma}$, namely $2^{\text{mean}(\log_2 \hat{\sigma})}$, is given.

larger than for CSA by (at most) 67%, decreasing for larger dimensions down to (at most) 13.4% in 1024 dimensions.

Clearly, the differences in the runtimes are due to differences in the ability to adapt σ . Therefore, let $\hat{\sigma} := \sigma \cdot n/d$ denote the normalized σ (w.r.t. dimensionality and distance from the optimum point). Note that, for the simple function scenario considered, for each dimension there is a unique value of $\hat{\sigma}$ resulting in maximum expected reduction of the approximation error per iteration (which is not known exactly, unfortunately). Figure 3 shows the course of $\hat{\sigma}$ in \log_2 -scale for each of the three CSA-variants for typical runs in 2-/16-/1024-dimensional space, respectively, where “typical” means median runtime. In Table 2 the mean and the standard deviation of $\log_2(\sigma^*)$ for these runs are given, as well as the log-mean of $\hat{\sigma}$, which we refer to as the average $\hat{\sigma}$.

For the 16- and 1024-dimensional problems, the original CSA adapts σ much more smoothly than its two derivatives, which is apparently due to its continuous σ -adaptation (most obvious in 1024 dimensions). For the 2-dimensional problem, however, CSA shows a larger standard deviation from the average $\hat{\sigma}$. Taking CSA as a reference, pCSA as well as CBA adapt σ such that it is too large on average in 16 dimensions, whereas in 1024 dimensions they adapt σ such that it is too small on average—besides the larger fluctuations. For 16 dimensions, which we consider practically relevant, as well as for 1024 dimensions, the average $\hat{\sigma}$ of pCSA lies right between the ones of CSA and CBA, respectively, which fits well with the differences in the runtimes. For the 2-dimensional problem, however, correlations between the average $\hat{\sigma}$ and the runtimes can hardly be found, which becomes especially clear for pCSA. In two dimensions the runs are actually quite short and, in addition, the step-lengths can deviate strongly from the expectation $\sigma \cdot \bar{\chi}$, so that the data might just be too noisy. Alternatively, there might be a completely different reason for the good performance of pCSA in 2-dimensional space, which would be very interesting to reveal.

6 Conclusions & Outlook

The main aim of this work has been to develop an algorithm for step-size control in black-box optimization which closely resembles the original CSA mechanism, but which is sufficiently simple to enable a theoretical/probabilistic analysis. From the experimental results obtained and from the construction details of the two proposed CSA-variants we conclude that CBA does fulfill both criteria.

Additionally, some interesting facts have been unveiled. One of these affects the change from continuous σ -update to phases in which σ is kept constant. Contrary to what we expected, this modification does not explain the large runtime differences in small dimensions. Those may rather be due to the large fluctuations observed in the step-sizes adjusted by the two CSA-derivatives; CSA step-size curves are obviously much smoother in higher dimensions. Furthermore, it has been found that an appropriate σ -scaling factor for both CSA-derivatives seems to follow a double square root (namely $1 + 1/n^{1/4}$) rather than a single square root function (like $1 + 1/\sqrt{n}$) as one might expect.

Currently, we work on the details of a theoretical/probabilistic analysis of CBA following the approach outlined in Section 4. Furthermore, we are going to experimentally review the performance of the two CSA-derivatives on more complex—actually, less trivial—functions in comparison to the original CSA (also and in a particular in combination with CMA), but also to classical direct-search methods like the ones reviewed in [12].

Acknowledgment. We thank Thomas Stützle for helping us improve the paper.

References

1. Hansen, N., Ostermeier, A.: Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In: Proc. IEEE Int'l Conference on Evolutionary Computation (ICEC). (1996) 312–317
2. Hansen, N.: List of references to various applications of CMA-ES (2008) <http://www.bionik.tu-berlin.de/user/niko/cmaapplications.pdf>.
3. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) Evolutionary Algorithm. *Theoretical Computer Science* **276** (2002) 51–82
4. Jägersküpper, J.: Algorithmic analysis of a basic evolutionary algorithm for continuous optimization. *Theoretical Computer Science* **379** (2007) 329–347
5. Rechenberg, I.: Cybernetic solution path of an experimental problem. Royal Aircraft Establishment (1965)
6. Schwefel, H.P.: Numerical Optimization of Computer Models. Wiley, New York (1981)
7. Jägersküpper, J.: Analysis of a simple evolutionary algorithm for minimization in Euclidean spaces. In: Proc. 30th Int'l Colloquium on Automata, Languages and Programming (ICALP). Volume 2719 of LNCS., Springer (2003) 1068–79
8. Jägersküpper, J.: How the (1+1) ES using isotropic mutations minimizes positive definite quadratic forms. *Theoretical Computer Science* **361** (2005) 38–56
9. Beyer, H.G.: The Theory of Evolution Strategies. Springer (2001)
10. Jägersküpper, J.: Probabilistic runtime analysis of (1+ λ) ES using isotropic mutations. In: Proc. 2006 Genetic and Evolutionary Computation Conference (GECCO), ACM Press (2006) 461–468
11. Beyer, H.G., Arnold, D.V.: Performance analysis of evolutionary optimization with cumulative step-length adaptation. *IEEE Transactions on Automatic Control* (2004) 617–622
12. Kolda, T.G., Lewis, R.M., Torczon, V.: Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review* **45** (2004) 385–482