Increasing the Serial and the Parallel Performance of the CMA-Evolution Strategy with Large Populations

Sibylle D. Müller[1], Nikolaus Hansen[2], and Petros Koumoutsakos[1]

[1] Institute of Computational Science, ETH Zürich
8092 Zürich, Switzerland
{muellers,petros}@inf.ethz.ch
[2] Fachgebiet für Bionik, Technische Universität Berlin
13355 Berlin, Germany
hansen@bionik.tu-berlin.de

Abstract. The derandomized evolution strategy (ES) with covariance matrix adaptation (CMA), is modified with the goal to speed up the algorithm in terms of needed number of generations. The idea of the modification of the algorithm is to adapt the covariance matrix in a faster way than in the original version by using a larger amount of the information contained in large populations. The original version of the CMA was designed to reliably adapt the covariance matrix in small populations and turned out to be highly efficient in this case. The modification scales up the efficiency to population sizes of up to $10n$, where n ist the problem dimension. If enough processors are available, the use of large populations and thus of evaluating a large number of search points per generation is not a problem since the algorithm can be easily parallelized.

1 Introduction

One of the commonly proposed advantages of evolution strategies (ES's) is that they can be easily parallelized, see e.g. Schwefel (1995) or Bäck, Hammel, and Schwefel (1997). ESs with λ children per generation (population size λ) are usually parallelized by distributing the function evaluation for each of the λ children on a different processor. When the number of children is smaller than the number of available processors, the advantage of using ES's in parallel cannot be fully exploited. Consequently, for a large number of processors the algorithm should be able to use a large population efficiently.

In this article, we consider a derandomized ES with covariance matrix adaptation (CMA-ES) for which experimental results (Hansen and Ostermeier, 1997, 2001) show a clear convergence velocity improvement when compared to other ES's. The primary feature of the CMA-ES is its reliability in adapting an arbitrarily oriented scaling of the search space in small populations. The algorithm is in particular independent of any orthogonal transformation of the coordinate system.

When optimizing considerably complex, e.g. highly nonseparable functions, the adaptation time becomes the limiting factor for the performance of the CMA-ES if the problem dimension n exceeds a certain threshold, usually $n \geq 10$. That is, the number of generations to adapt the covariance matrix of the search distribution to the function topology is the prominent factor for the degraded performance of the algorithm. The reason is that in the CMA-ES $(n^2+n)/2$ elements of the symmetric covariance matrix $\boldsymbol{C}$ need to be adapted while the search process itself needs only to adjust n variables. Interestingly, for population sizes greater than 20 the adaptation time (i.e., the time to adapt the $(n^2+n)/2$ elements of the covariance matrix) becomes practically independent of the population size (Hansen 1998) . That means, the performance in number of function evaluations decreases linearly with increasing population size. Alternatively, the implementation of the original CMA-ES on massively parallel computer architectures, e.g. the Beowulf cluster with hundreds of processors, supplies no substantial advantage compared to the use of twenty processors. On remarkably complex functions, the time complexity is of $\mathcal{O}(n^2)$, independent of the population size and the processor number.

How can we use the CMA-ES efficiently on massively parallel architectures with hundreds of processors? How can we increase the efficiency of the CMA-ES, when a large population is preferable to a small one due to other reasons? To increase λ alone does not help shortening the adaptation time as pointed out above. Additionally, a faster adaptation mechanism must be implemented being comparably reliable. The idea that we present in this paper is to increase the adaptation rate of the covariance matrix compared to the original algorithm, without losing its reliablity by exploiting a larger amount of information per generation. This is possible because an increased population should contain more information ready to be exploited in order to obtain a reduced adaptation time. Compared to the original adaptation mechanism the proposed modification would usually require much fewer function evaluations when λ is large but could be slightly less effective within small populations.

The working principles of the original algorithm, the CMA-ES, referred to as Orig-CMA here, are outlined in Section 2 and the modifications are presented in Section 3. In Section 4, the simulation results are discussed and Section 5 provides a conclusion.

2 Algorithm of the CMA-ES

Following Hansen and Ostermeier (2001) , in the $(\mu_{\mathrm{I}}, \lambda)$-CMA-ES the λ offspring of generation $g+1$ are computed by

$$\boldsymbol{x}_k^{(g+1)} = \langle \boldsymbol{x} \rangle_\mu^{(g)} + \sigma^{(g)} \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{z}_k^{(g+1)} \quad , \quad k = 1, \ldots, \lambda, \tag{1}$$

where

$$\langle \boldsymbol{x} \rangle_\mu^{(g)} = \frac{1}{\mu} \sum_{i \in I_{sel}^{(g)}} \boldsymbol{x}_i^{(g)} \tag{2}$$

represents the center of mass of the selected individuals of generation g, and $I_{sel}^{(g)}$ is the set of indices of the selected individuals of generation g, with $|I_{sel}^{(g)}| = \mu$. The random vectors $\boldsymbol{z}$ from Equation (1) are $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ distributed (n-dimensional normally distributed with expectation zero and the identity covariance matrix) and serve to generate offspring for generation $g+1$. They can be used to calculate $\langle \boldsymbol{z} \rangle_\mu^{(g+1)}$ analogously to $\langle \boldsymbol{x} \rangle_\mu^{(g)}$. The columns of $\boldsymbol{B}^{(g)}$ represent eigenvectors of the covariance matrix $\boldsymbol{C}^{(g)}$. $\boldsymbol{D}^{(g)}$ is a diagonal matrix whose elements are the square roots of the eigenvalues of $\boldsymbol{C}^{(g)}$. Hence, the relation of $\boldsymbol{B}^{(g)}$ and $\boldsymbol{D}^{(g)}$ to $\boldsymbol{C}^{(g)}$ can be expressed by

$$\boldsymbol{C}^{(g)} = \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \left(\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \right)^T \qquad \text{and} \qquad \boldsymbol{C}^{(g)} \boldsymbol{b}_i^{(g)} = \left(d_{ii}^{(g)} \right)^2 \cdot \boldsymbol{b}_i^{(g)} \tag{3}$$

where $\boldsymbol{b}_i^{(g)}$ represents the i-th column of $\boldsymbol{B}^{(g)}$. Each covariance matrix corresponds to a hyperellipsoid which defines the surface of equal probability to place offspring. Here, the eigenvectors of the covariance matrix define the orientation of the hyperellipsoid and the eigenvalues define the lengths of its axes.

The evolution path $\boldsymbol{p}_c^{(g+1)}$ is calculated by

$$\boldsymbol{p}_c^{(g+1)} = (1 - c_c) \cdot \boldsymbol{p}_c^{(g)} + \sqrt{c_c \cdot (2 - c_c)} \cdot \underbrace{\frac{\sqrt{\mu}}{\sigma^{(g)}} \left(\langle \boldsymbol{x} \rangle_\mu^{(g+1)} - \langle \boldsymbol{x} \rangle_\mu^{(g)} \right)}_{= \sqrt{\mu}\, \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \langle \boldsymbol{z} \rangle_\mu^{(g+1)}} \tag{4}$$

and is used to build the covariance matrix of generation $g+1$

$$\boldsymbol{C}^{(g+1)} = (1 - c_{\text{cov}}) \cdot \boldsymbol{C}^{(g)} + c_{\text{cov}} \cdot \boldsymbol{p}_c^{(g+1)} \left(\boldsymbol{p}_c^{(g+1)} \right)^T . \tag{5}$$

The update of $\boldsymbol{C}$ is done with a symmetric matrix of rank one (right summand in Equation (5)). The strategy parameters $c_c \in]0, 1]$ and $c_{\text{cov}} \in [0, 1[$ determine the accumulation time for the evolution path $\boldsymbol{p}_c^{(g+1)}$ and the change rate of the covariance matrix, respectively. Note that for $c_c = 1$ in Equation (4) the evolution path reduces to $\sqrt{\mu} \boldsymbol{B} \boldsymbol{D} \langle \boldsymbol{z} \rangle_\mu$ which is the mean mutation step of the last generation. Also, the update of $\boldsymbol{C}$ is independent of the adaptation of the global step size.

For the adaptation of the global step size, the evolution path $\boldsymbol{p}_\sigma^{(g+1)}$ that is not scaled by $\boldsymbol{D}^{(g)}$ is calculated by

$$\boldsymbol{p}_\sigma^{(g+1)} = (1 - c_\sigma) \cdot \boldsymbol{p}_\sigma^{(g)} + \sqrt{c_\sigma \cdot (2 - c_\sigma)} \cdot \underbrace{\sqrt{\mu}\, \boldsymbol{B}^{(g)} \langle \boldsymbol{z} \rangle_\mu^{(g+1)}}_{= \boldsymbol{B}^{(g)} \left(\boldsymbol{D}^{(g)} \right)^{-1} \left(\boldsymbol{B}^{(g)} \right)^{-1} \frac{\sqrt{\mu}}{\sigma^{(g)}} \left(\langle \boldsymbol{x} \rangle_\mu^{(g+1)} - \langle \boldsymbol{x} \rangle_\mu^{(g)} \right)} \tag{6}$$

and its length is used to compute the step size for generation $g+1$

$$\sigma^{(g+1)} = \sigma^{(g)} \cdot \exp \left(\frac{1}{d_\sigma} \frac{\| \boldsymbol{p}_\sigma^{(g+1)} \| - \hat{\chi}_n}{\hat{\chi}_n} \right), \tag{7}$$

where $\hat{\chi}_n = \mathrm{E}\left[\|\mathcal{N}(\mathbf{0}, \boldsymbol{I})\|\right]$ is the expected length of a $(\mathbf{0}, \boldsymbol{I})$-normally distributed random vector and $\hat{\chi}_n$ is approximated by $\hat{\chi}_n \approx \sqrt{n}\left(1 - \frac{1}{4n} + \frac{1}{21n^2}\right)$. The strategy parameter $c_\sigma \in]0,1]$ determines the accumulation time for the evolution path $\boldsymbol{p}_\sigma^{(g+1)}$, and d_σ is a damping parameter.

The default strategy parameter setting, discussed in Hansen and Ostermeier (2001) in detail, is as follows:

$$c_c = \tfrac{4}{n+4},\; c_{\mathrm{cov}} = \tfrac{2}{(n+\sqrt{2})^2},\; c_\sigma = \tfrac{4}{n+4},\; d_\sigma = c_\sigma^{-1} + 1 \tag{8}$$

Initial values are $\boldsymbol{p}^{(0)} = \mathbf{0}, \boldsymbol{p}_\sigma^{(0)} = \mathbf{0}$ and the initial covariance matrix $\boldsymbol{C}^{(0)}$ is the identity matrix $\boldsymbol{I}$.

3 Modified Algorithm

All modifications solely regard Equation (5) that describes the change of the covariance matrix, i.e., the change of the mutation distribution shape, and c_{cov}. Everything else, in particular the global step size adaptation mechanism remains unchanged in this paper. We add to Equation (5) the following term:

$$\begin{aligned}\boldsymbol{Z}^{(g+1)} &= \frac{1}{\mu} \sum_{i \in I_{sel}^{(g+1)}} \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{z}_i^{(g+1)} \left(\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \boldsymbol{z}_i^{(g+1)}\right)^T \\ &= \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \left(\frac{1}{\mu} \sum_{i \in I_{sel}^{(g+1)}} \boldsymbol{z}_i^{(g+1)} \left(\boldsymbol{z}_i^{(g+1)}\right)^T\right) \left(\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)}\right)^T \end{aligned} \tag{9}$$

that is a symmetrical $n \times n$ matrix with rank $\min(\mu, n)$ (with probability one).

The modification of Equation (5) then reads

$$\boldsymbol{C}^{(g+1)} = (1 - c_{\mathrm{cov}}) \cdot \boldsymbol{C}^{(g)} + c_{\mathrm{cov}} \Big(\alpha_{\mathrm{cov}} \cdot \boldsymbol{p}_c^{(g+1)} \left(\boldsymbol{p}_c^{(g+1)}\right)^T + (1 - \alpha_{\mathrm{cov}}) \cdot \boldsymbol{Z}^{(g+1)}\Big) \tag{10}$$

where $0 \le \alpha_{\mathrm{cov}} \le 1$. Note that for $\alpha_{\mathrm{cov}} = 1$ Equation (10) and Equation (5) are identical and the original CMA-ES is restored. Decreasing α_{cov} changes the parameterized algorithm continuously. Results are presented for $\alpha_{\mathrm{cov}} = 0$ and $\alpha_{\mathrm{cov}} = \frac{1}{\mu}$ and compared with the original CMA algorithm, where $\alpha_{\mathrm{cov}} = 1$, in Section 4.

Since the rank of $\boldsymbol{Z}^{(g+1)}$ is larger than 1, more information is passed to the covariance matrix in each generation. Thus, the adaptation becomes more reliable. Therefore, the adaptation time $1/c_{\mathrm{cov}}$ can be decreased, or in other words, the learning rate c_{cov} can be increased yielding a higher adaptation speed. For $\alpha_{\mathrm{cov}} = 1$, c_{cov} is used as in Equation (8). For $\alpha_{\mathrm{cov}} = 0$, experiments showed that

$$c_{\mathrm{cov}} = \min\left(1, \frac{2\mu - 1}{(n+2)^2 + \mu}\right) . \tag{11}$$

Table 1. Convex quadratic test functions and stopping criteria.

Name	Function
Sphere	$f_{\text{sphere}} = \sum_{i=1}^{n}(x_i)^2$
Ellipsoid	$f_{\text{elli}} = \sum_{i=1}^{n}(1000^{\frac{i-1}{n-1}} x_i)^2$
Cigar	$f_{\text{cigar}} = x_1^2 + \sum_{i=2}^{n}(1000 x_i)^2$

trades of reasonably well the reliability and the adaptation speed of the covariance matrix adaptation, and is thus chosen in the strategies with $0 \leq \alpha_{\text{cov}} < 1$. A closed expression for c_{cov} has yet to be found that can be used in both types of strategies represented by $\alpha_{\text{cov}} = 1$ and $0 \leq \alpha_{\text{cov}} < 1$.

Equations (9) and (10) are analyzed to motivate the coefficients $\frac{1}{\mu}$ in Equation (9) and α_{cov} in conjunction with $(1 - \alpha_{\text{cov}})$ from Equation (10). Under random selection, the symmetric matrix

$$\sum_{i \in I_{sel}} \boldsymbol{z}_i (\boldsymbol{z}_i)^T = \sum_{i \in I_{sel}} \begin{pmatrix} z_{i1}^2 & z_{i1}z_{i2} & \cdots & z_{i1}z_{in} \\ z_{i2}z_{i1} & z_{i2}^2 & \cdots & z_{i2}z_{in} \\ \vdots & \vdots & \ddots & \vdots \\ z_{in}z_{i1} & z_{in}z_{i2} & \cdots & z_{in}^2 \end{pmatrix} \tag{12}$$

from Equation (9) has diagonal elements that are χ_μ^2 distributed and off-diagonal elements with expectation zero. With $\mathrm{E}\left[\sum_{i \in I_{sel}} z_{ij}^2\right] = \mu$, $j = 1, \ldots, n$, we have that

$$\mathrm{E}\left[\boldsymbol{Z}^{(g+1)}\right] = \frac{1}{\mu} \boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)} \, \mu \boldsymbol{I} \, (\boldsymbol{B}^{(g)} \boldsymbol{D}^{(g)})^T = \boldsymbol{C}^{(g)} \tag{13}$$

under the given selection model. Equation (13) is the reason for choosing the coefficient $\frac{1}{\mu}$ in Equation (9). With $\mathrm{E}\,[\boldsymbol{p}_c^{(g+1)} (\boldsymbol{p}_c^{(g+1)})^T] = \boldsymbol{C}^{(g)}$ (Hansen 1998) and $\mathrm{E}\,[\boldsymbol{Z}^{(g+1)}] = \boldsymbol{C}^{(g)}$, we conclude from Equation (10) that $\mathrm{E}\,[\boldsymbol{C}^{(g+1)}] = \boldsymbol{C}^{(g)}$. This is the reason for choosing $(1 - \alpha_{\text{cov}})$ in conjunction with α_{cov} in Equation (10).

To compare the strategies with different α_{cov}, the functions shown in Table 1 are tested. Tests are carried out in the dimensions $n = [2, 3, 5, 10, 20, 40, 80]$ and for parent numbers $\mu = [2, \lceil n/4 \rceil, \lceil n/2 \rceil, n, 2n, 4n, \lceil n^2/4 \rceil, \lceil n^2/2 \rceil, n^2]$ with a population size of $\lambda = 4\mu$. Initial values are set to $\langle \boldsymbol{x} \rangle_\mu^{(0)} = \mathbf{1}$ and $\sigma^{(0)} = 1$. The search is terminated as soon as $f_{\text{stop}} = 10^{-10}$ is reached.

4 Discussion of the Results

Three different strategy variants are presented: New-CMA, where $\alpha_{\text{cov}} = 0$; Orig-CMA, where $\alpha_{\text{cov}} = 1$; and Hybr-CMA, where $\alpha_{\text{cov}} = \frac{1}{\mu}$. Note that for $\mu = 1$, Hybr-CMA is identical to Orig-CMA. The simulation results for the various strategies are analyzed from two different point of views:

Serial performance. We analyze the number of overall function evaluations to reach f_{stop}. This is the appropriate point of view if optimization is performed on a single processor or on a small number of processors that does not exceed the smallest sensible population size λ, usually five to ten. In this case the number of function evaluations is an appropriate measure for the time to reach f_{stop}.

Parallel performance. We analyze the number of generations to reach f_{stop}. When a larger number of processors is available, it becomes interesting to evaluate the number of generations to reach f_{stop}, especially if λ is equal to the number of processors. In this case, the number of generations is an appropriate measure for the time to reach f_{stop} in a single run.

First, we discuss the serial performance on f_{sphere} as shown in Figure 1, upper left. For $\lambda < 10n$ the performance of Orig-CMA and Hybr-CMA are similar as expected. For larger population sizes, Hybr-CMA becomes faster than Orig-CMA by a factor of up to three. This effect cannot be attributed to a faster adaptation of the distribution *shape* because on f_{sphere} the shape is optimal already at the beginning. The reason for the better performance of Hybr-CMA is the faster adaptation of the overall variance of the distribution. Originally, the cumulative path length control (Equations (6) and (7)) facilitates the adaptation of the global step size, i.e. the adaptation of the overall variance. The parameter that tunes the adaptation speed (d_σ in Equation (7)) was (a) chosen conservatively resulting in a somewhat slower but more robust algorithm (Hansen and Ostermeier 2001 , Section 5.1) and (b) chosen w.r.t. small population sizes that realize smaller progress rates than larger populations per generation and therefore demand slower adaptation rates. Consequently, for $\mu > n$ and $\alpha_{\text{cov}} \ll 1$ the distribution adaptation in Equation (10) can successfully contribute to the adapation speed of the overall variance in Hybr-CMA because the change rate $c_{\text{cov}} > \frac{1}{n}$. This is the presumable reason for the observed speed-up on f_{sphere}.

Even though this effect seems to be advantageous at first sight, it may become disadvantageous if the distribution adaptation influences the magnitude of the overall variance significantly. While the path length control is shown to adapt nearly optimal step lengths even for $\mu > 1$ (at least for $\mu < \lambda < n$, Hansen 1998), the distribution adaptation as described in Equation (10) acquires too small step lengths rapidly, if—unlike on f_{sphere}—*the optimal* step length remains constant over time (i.e. in a stationary environment). When the optimal step length decreases significantly fast over time, as on f_{sphere}, this can be an advantage. However, an algorithm that adapts drastically too small variances in a stationary environment is not preferable in general. This suggests an upper limit for reasonable population sizes, arguable at $\lambda \approx 10n$.

Next, the serial performance of the strategy variants on f_{elli} and f_{cigar} is evaluated. The most prominent effect in these results is the dependency on λ. The smallest $\lambda = 8$ (and even a smaller λ for $n = 5$) performs best in all cases. For example, if $n = 20$ the decline of the serial performance between $\lambda = 8$ and $\lambda = 80$ amounts roughly to a factor of 5.5 (on f_{cigar}) and 7.5 (on f_{elli}) for Orig-CMA and of 1.7 (on f_{elli}) and 4.5 (on f_{cigar}) for Hybr-CMA. Considering the serially optimal $\lambda = 8$, the performance difference between Orig-CMA and Hybr-CMA is small. This leads to the conclusion that the introduced modifications in

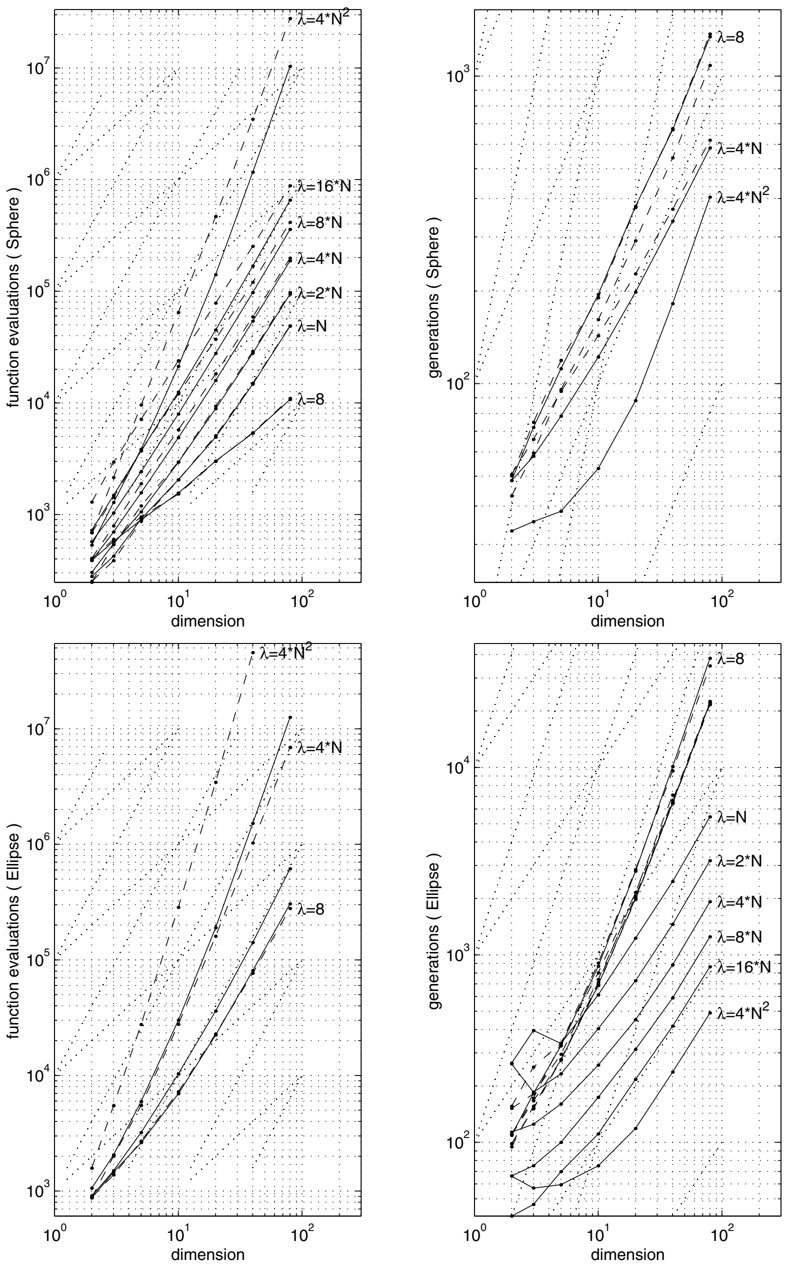

Fig. 1. Number of function evalutations (left) and number of generations (right) over the problem dimension for the sphere function (above) and the ellipse function (below). The Orig-CMA (– – –) and the Hybr-CMA (—) are plotted for $\lambda = 8, n, 2n, 4n, 8n, 16n, 4n^2$ if curves are far apart, or for $\lambda = 8, 4n, 4n^2$.

Hybr-CMA yield at least similar, if not slightly better serial performance even for $\mu = 2$ and $\lambda = 8$ in the tested cases.

Comparing New-CMA with the other strategy variants where $\lambda = 8$ reveals a significant result on f_{cigar} (Figure 2). While Orig-CMA and Hybr-CMA need about $500n$ function evaluations to reach f_{stop}, New-CMA needs about $120n^2$ function evaluations. Using the evolution path $\boldsymbol{p}_c$ in Equation (10) in addition with a cumulation parameter of $c_c \approx \frac{1}{n}$ in Equation (4) yields this impressive speed-up of Orig-CMA and Hybr-CMA. Although detected in earlier investigations (Hansen and Ostermeier 2001), this speed-up is noteworthy in that a completely adaptable covariance matrix with $\frac{n^2+n}{2}$ free parameters can be adapted to certain topologies in $\mathcal{O}(n)$ function evaluations. Since this observation on f_{cigar} is the major difference between Hybr-CMA and New-CMA, the latter algorithm is excluded from the remaining discussion.

While the differences between Orig-CMA and Hybr-CMA are marginal for $\lambda = 8$, the picture changes in favor of Hybr-CMA when the population size is increased. For $\lambda = 8$, the scaling of the needed function evaluations w.r.t. the problem dimension (i.e. the slope of the graphs) is linear on f_{sphere} and f_{cigar}, but it is almost quadratic on f_{elli}. For $\lambda \propto n$, the scaling of Hybr-CMA becomes nearly quadratic, regardless whether the scaling is linear or quadratic for $\lambda = 8$. This is in contrast to Orig-CMA where the scaling *always* deteriorates when λ is increased from a constant value to $\lambda \propto n$. As a result, Hybr-CMA performs never worse and often greatly better than Orig-CMA if $\lambda \propto n$. This clear advantage can of course be expected only if $\mu \propto \lambda$, e.g. $\mu \approx \lambda/4$ as chosen in our investigations. Concluding these observations, Hybr-CMA must be undoubtedly preferred w.r.t. its serial performance if $\mu > 3$ and $n > 5$.

Second, we discuss the parallel performance that comes into play if λ is chosen considerably large. This means that we concentrate the discussion of parallel performance on the cases where $\lambda \propto n$ and $\lambda = 4n^2$.

Certainly, the difference in performance between Orig-CMA and Hybr-CMA discussed above translates to the parallel case. Hybr-CMA outperforms Orig-CMA overall and on any single function, if $\lambda \gg 8$ (assuming $\mu \approx \lambda/4$). Therefore, it is more interesting to interpret the parallel strategy behavior in relation to λ. The parallel performance of Orig-CMA does not change dramatically when λ is increased. The improvement never exceeds a factor of two in dimensions up to 80. In contrast to Orig-CMA, where the parallel performance is less dependent on λ, Hybr-CMA shows a vigorous improvement when λ is increased. For example, increasing λ from n to $8n$, i.e. by a factor of eight, improves the parallel performance by a factor greater than four on f_{elli}.

However, the most impressive result concerns the scaling of Hybr-CMA with respect to the number of generations where $\lambda \propto n$. When $\lambda \leq 10n$, Orig-CMA scales (nearly) quadratically w.r.t. the number of generations on all functions except on f_{sphere}. The same observation holds for Hybr-CMA for $\lambda = 8$. However, when λ is increased to be proportional to n in Hybr-CMA, the number of generations scales *linearly* with the problem dimension on all convex quadratic test functions. On f_{sphere}, the scaling for $\lambda \propto n$ is even slightly sublinear. The

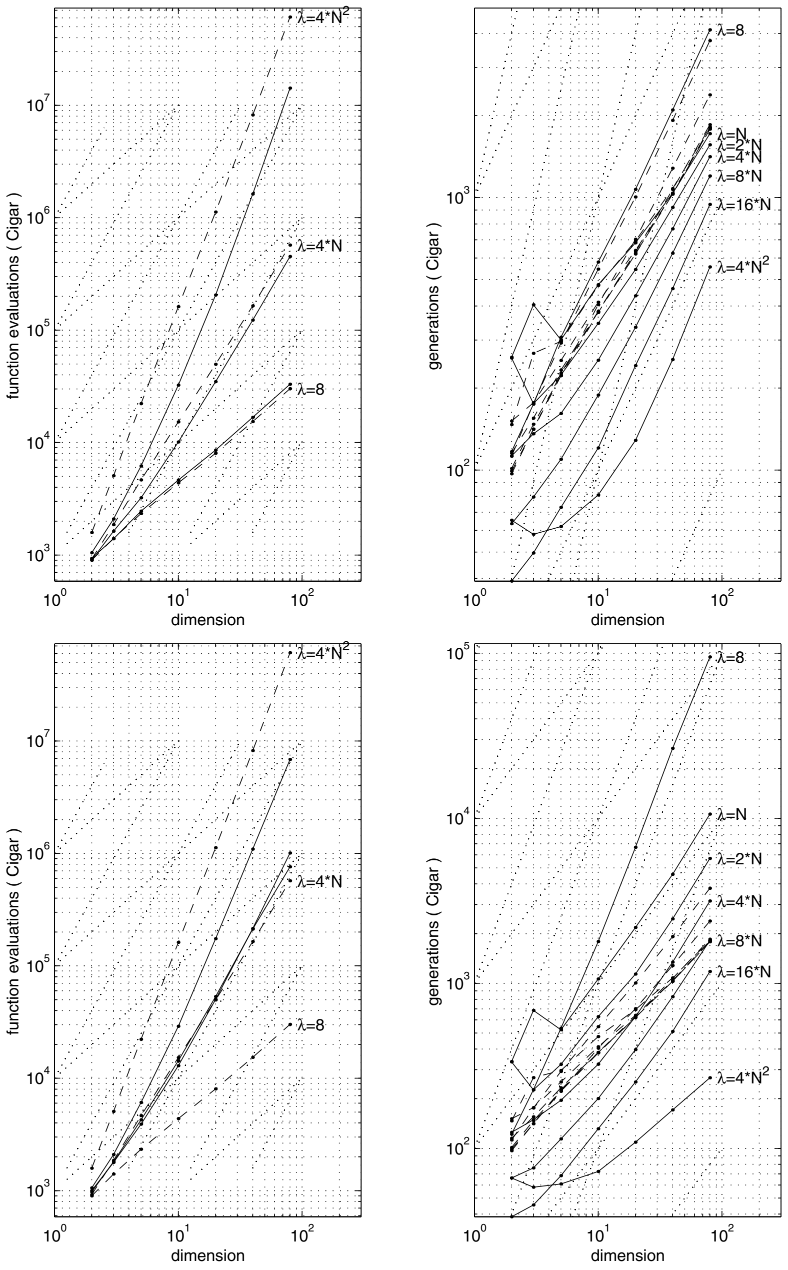

Fig. 2. Number of function evaluations (left) and number of generations (right) over the problem dimension for the cigar comparing Orig-CMA with Hybr-CMA (above) and the cigar comparing Orig-CMA with New-CMA (below). The Orig-CMA (– – –) and the Hybr-CMA or New-CMA (—) are plotted for $\lambda = 8, n, 2n, 4n, 8n, 16n, 4n^2$ if curves are far apart, or for $\lambda = 8, 4n, 4n^2$.

roughly linear scaling for $\lambda \propto n$ is the main improvement of the new Hybr-CMA compared with Orig-CMA.

5 Conclusions

We presented a modified algorithm derived from the derandomized evolution strategy with covariance matrix adapation. Our goal was to devise a technique with which we can optimize in fewer number of generations than with the original strategy, allowing to exploit the often emphasized feature of evolution strategies being easily parallelizable.

This goal is achieved for population sizes up to $\lambda = 10n$. Choosing $\alpha_{\text{cov}} = \frac{1}{\mu}$ and $\mu \approx \lambda/4$, the modified algorithm seems to efficiently exploit the information prevalent in the population and reveals mainly linear time complexity for population sizes proportional to n and up to $10n$, if fully parallelized. This means we were able to reduce the time complexity roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.

For $\mu = 1$, the modified algorithm is identical with the original one (assuming identical recombination weights in the latter). With increasing μ, the performance improves remarkably compared with the original algorithm. In our tests, the modified algorithm with $\alpha_{\text{cov}} = \frac{1}{\mu}$ reveals no disadvantage compared with the original one. Only for population sizes larger than $10n$ the adjustment of the overall variance becomes problematic. As a conclusion, the efficiency of the adaptation of the distribution shape in large populations seems to be satisfying for the moment. Future work will address further developments for the adaptation of the global step size.

References

Bäck et al., 1997. Bäck, T., Hammel, U., and Schwefel, H.-P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):3–17.

Hansen, 1998. Hansen, N. (1998). *Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie. Eine Untersuchung zur entstochastisierten, koordinatensystemunabhängigen Adaptation der Mutationsverteilung.* Mensch und Buch Verlag, Berlin. ISBN 3-933346-29-0.

Hansen and Ostermeier, 1997. Hansen, N. and Ostermeier, A. (1997). Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu_{\text{I}}, \lambda)$-CMA-ES. In Zimmermann, H.-J., editor, *EUFIT'97, 5th Europ. Congr. on Intelligent Techniques and Soft Computing, Proceedings*, pages 650–654, Aachen, Germany. Verlag Mainz.

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

Schwefel, 1995. Schwefel, H.-P. (1995). *Evolution and Optimum Seeking.* Sixth-Generation Computer Technology Series. John Wiley & Sons Inc., New York.