Benchmarking a Weighted Negative Covariance Matrix Update on the BBOB-2010 Noiseless Testbed

Nikolaus Hansen, Raymond Ros

Benchmarking a Weighted Negative Covariance Matrix Update on the BBOB-2010 Noiseless Testbed

Nikolaus Hansen
INRIA Saclay, TAO team
LRI, Bat 490 Univ. Paris-Sud
91405 Orsay Cedex, France
nikolaus.hansen@inria.fr

Raymond Ros
TAO Team-Project – INRIA Saclay
LRI, Bat 490, Univ. Paris-Sud
F-91405 Orsay Cedex, France
raymond.ros@inria.fr

ABSTRACT

We implement a weighted negative update of the covariance matrix in the CMA-ES—weighted active CMA-ES or, in short, aCMA-ES. We benchmark the IPOP-aCMA-ES and compare the performance with the IPOP-CMA-ES on the BBOB-2010 noiseless testbed in dimensions between 2 and 40. On nine out of 12 essentially unimodal functions, the aCMA is faster than CMA, in particular in larger dimension. On at least three functions it also leads to a (slightly) better scaling with the dimension. In none of the 24 benchmark functions aCMA appears to be significantly worse in any dimension. On two and five functions, IPOP-CMA-ES and IPOP-aCMA-ES respectively exceed the record observed during BBOB-2009.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*global optimization, unconstrained optimization*; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms, performance, comparison

Keywords

CMA-ES, IPOP-CMA-ES, active CMA-ES, Benchmarking, Black-box optimization

1. INTRODUCTION

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [12, 11, 10] is a stochastic search procedure that samples new candidate solutions from a multivariate normal distribution thereof mean and covariance matrix are adapted after each iteration. The $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES samples λ new candidate solutions and selects the μ best among them. They contribute in a weighted manner to the update of the distribution parameters. The algorithm is non-elitist by nature, but a practical implementation will preserve the best-ever evaluated solution. Elitist variants of the CMA-ES [13] are often slightly faster but more susceptible of getting stuck in a suboptimal local optimum.

The IPOP-CMA-ES [1] implements a restart procedure. Before each restart, λ is doubled. In most cases, doubling λ increases the length of single runs, but it often improves the quality of the best found solution. The BIPOP-CMA-ES [6], proposed recently, maintains two budgets. Under the first budget, an IPOP-CMA-ES is executed. Under the second budget, a multi-start $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES with various small population sizes is entertained.

Previous Benchmarking of CMA-ES. The $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES has been quite successful in several elaborate benchmarking exercises. Notably, the IPOP-CMA-ES [1] in the special session on real parameter optimization CEC-05[1] and the BIPOP-CMA-ES [6] in the black-box optimization benchmarking BBOB-2009[2] have shown excellent performance [5, 8] on respectively 25 and 24 uni- and multi-modal benchmark functions in search space dimension up to 50.

A comparison of two Estimation of Distribution Algorithms and three Evolution Strategies is presented in [15] on 14 functions. The CMA-ES performed best on 11 functions with a median speed-up by a factor of at least 30 compared to any other algorithm. On the remaining functions the maximum loss factor was 1.4.

A comparison on a small collection of multi-modal function with three other algorithms is presented in [10]. The CMA-ES with optimal population size performs clearly best on the non-separable functions and is clearly outperformed on additively decomposable functions by Differential Evolution (DE).

A comparison of derivative-free optimization methods and BFGS on smooth unimodal functions is presented in [2]. On these functions, PSO and DE are in general significantly slower than CMA-ES. Only PSO performs similar on separable problems. NEWUOA and BFGS remarkably outperform CMA-ES, if the problem is convex and has a moderate conditioning. On non-convex problems with at least moderate condition number (i.e. 10^4) and on non-separable problems with higher condition number (i.e. 10^6), the CMA-ES breaks even and becomes advantageous with increasing condition number, tested up to 10^{10}.

In [3][3] the CMA-ES is recognized as state-of-the-art in

GECCO'10, July 7–11, 2010, Portland, Oregon, USA.

[1] http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC-05/CEC05.htm
[2] http://coco.gforge.inria.fr/doku.php?id=bbob-2009
[3] http://www.scholarpedia.org/article/Evolution_

Given $t \in \mathbb{N} \cup \{0\}$, $\mathbf{m}^t \in \mathbb{R}^D$, $\sigma^t \in \mathbb{R}_+$, $\mathbf{C}^t \in \mathbb{R}^{D\times D}$ positive definite, $\mathbf{p}_\sigma^t, \mathbf{p}_\mathrm{c}^t \in \mathbb{R}^D$ and $\mathbf{p}_\sigma^{t=0} = \mathbf{p}_\mathrm{c}^{t=0} = \mathbf{0}$, $\mathbf{C}^{t=0} = \mathbf{I}$

$$\mathbf{x}_i \sim \mathbf{m}^t + \sigma^t \times \mathcal{N}\left(\vec{0}, \mathbf{C}^t\right) \quad \text{is normally distributed for } i = 1, \ldots, \lambda \tag{1}$$

$$\mathbf{m}^{t+1} = \mathbf{m}^t + c_\mathrm{m} \sum_{i=1}^{\mu} w_i \left(\mathbf{x}_{i:\lambda} - \mathbf{m}^t\right) \quad \text{where } f(\mathbf{x}_{1:\lambda}) \le \cdots \le f(\mathbf{x}_{\mu:\lambda}) \le f(\mathbf{x}_{\mu+1:\lambda}) \ldots \tag{2}$$

$$\mathbf{p}_\sigma^{t+1} = (1-c_\sigma)\,\mathbf{p}_\sigma^t + \sqrt{c_\sigma(2-c_\sigma)\mu_\mathrm{w}}\;\mathbf{C}^{t-\frac{1}{2}}\,\frac{\mathbf{m}^{t+1}-\mathbf{m}^t}{c_\mathrm{m}\,\sigma^t} \tag{3}$$

$\mathbf{C}^{t-\frac{1}{2}}$ is symmetric with positive eigenvalues such that $\mathbf{C}^{t-\frac{1}{2}}\mathbf{C}^{t-\frac{1}{2}}$ is the inverse of $\mathbf{C}^t$. If $\mathbf{B}\Lambda\mathbf{B}^\mathrm{T} = \mathbf{C}$ is an eigendecomposition into an orthogonal matrix $\mathbf{B}$ and a diagonal matrix Λ, we have $\mathbf{C}^{-\frac{1}{2}} = \mathbf{B}\Lambda^{-1/2}\mathbf{B}^\mathrm{T}$.

$$h_\sigma = \begin{cases} 1 & \text{if } \|\mathbf{p}_\sigma^{t+1}\| < \sqrt{1-(1-c_\sigma)^{2(t+1)}}\left(1.4 + \frac{2}{D+1}\right)\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\| \\ 0 & \text{otherwise} \end{cases} \quad \text{stalls the update of } \mathbf{p}_\mathrm{c} \text{ in (5) when } \sigma \text{ increases rapidely} \tag{4}$$

$$\mathbf{p}_\mathrm{c}^{t+1} = (1-c_\mathrm{c})\,\mathbf{p}_\mathrm{c}^t + h_\sigma\sqrt{c_\mathrm{c}(2-c_\mathrm{c})\mu_\mathrm{w}}\,\frac{\mathbf{m}^{t+1}-\mathbf{m}^t}{c_\mathrm{m}\,\sigma^t} \quad \text{evolution path for the rank-one update of } \mathbf{C}^t \tag{5}$$

$$\mathbf{C}_\mu^+ = \sum_{i=1}^{\mu} w_i \frac{\mathbf{x}_{i:\lambda}-\mathbf{m}^t}{\sigma^t} \times \frac{\left(\mathbf{x}_{i:\lambda}-\mathbf{m}^t\right)^\mathrm{T}}{\sigma^t} \quad \text{matrix with rank } \min(\mu, D) \text{ for the so-called rank-}\mu \text{ update of } \mathbf{C}^t \tag{6}$$

$$\mathbf{C}_\mu^- = \sum_{i=0}^{\mu-1} w_{i+1}\mathbf{y}_{\lambda-i:\lambda}\mathbf{y}_{\lambda-i:\lambda}^\mathrm{T} \quad \text{with } \mathbf{y}_{\lambda-i:\lambda} = \frac{\|\mathbf{C}^{t-\frac{1}{2}}(\mathbf{x}_{\lambda-\mu+1+i:\lambda}-\mathbf{m}^t)\|}{\|\mathbf{C}^{t-\frac{1}{2}}(\mathbf{x}_{\lambda-i:\lambda}-\mathbf{m}^t)\|} \times \frac{\mathbf{x}_{\lambda-i:\lambda}-\mathbf{m}^t}{\sigma^t} \tag{7}$$

$$\mathbf{C}^{t+1} = (1-c_1-c_\mu + c^-\alpha_\mathrm{old}^-)\,\mathbf{C}^t + c_1\underbrace{\mathbf{p}_\mathrm{c}^{t+1}\mathbf{p}_\mathrm{c}^{t+1\,\mathrm{T}}}_{\text{rank one update}} + (c_\mu + c^-(1-\alpha_\mathrm{old}^-))\underbrace{\mathbf{C}_\mu^+}_{\text{rank-}\mu\text{ update}} - c^-\underbrace{\mathbf{C}_\mu^-}_{\text{"active"}} \tag{8}$$

$$\sigma^{t+1} = \sigma^t \times \exp\left(\Delta_\sigma^{\max} \wedge \frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^{t+1}\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right) \quad \text{for } \mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\| = \sqrt{2}\,\Gamma(\tfrac{n+1}{2})/\Gamma(\tfrac{n}{2}) \text{ we use the approximation } \sqrt{D}\left(1 - \tfrac{1}{4D} + \tfrac{1}{21D^2}\right) \tag{9}$$

Figure 1: Update equations for the state variables in the $(\mu/\mu_\mathbf{w}, \lambda)$-aCMA-ES with iteration index $t = 0, 1, 2, \ldots$ and $a \wedge bc + d = \min(a, bc + d)$. The chosen ordering of equations allows to remove the time index in all variables but $\mathbf{m}^t$. The symbol $\mathbf{x}_{i:\lambda}$ is the i-th best of the solutions $\mathbf{x}_1, \ldots, \mathbf{x}_\lambda$

real-valued evolutionary optimization. Partly due to the fact that the algorithm is quasi-parameter-free, the CMA-ES is widely applied and has become a quasi-standard.

A Further Improvement. The active CMA-ES proposed in [14] introduces a negative update of the covariance matrix in the $(\mu/\mu_\mathrm{I}, \lambda)$-CMA-ES. The authors investigate essentially unimodal functions. They observe a significant speed-up in particular on the discus function. The speed-up reaches almost a factor of three in dimension 20 (compared to the $(\mu/\mu_\mathrm{I}, \lambda)$-CMA-ES) and it increases with increasing dimension (i.e. the update leads to an improved scaling with increasing search space dimension). The negative update can in particular speed-up the adaptation of small variances in a small number of directions. The speed-up is less pronounced with a larger population size λ.

This Paper. We combine the technique of negative updates with the $(\mu/\mu_\mathrm{w}, \lambda)$-CMA-ES and implement a *weighted* negative update procedure for the covariance matrix. The update also improves compared to [14], in that it remains feasible with a large population size. Consequently, the objective of this paper is threefold. 1) Specify a weighted negative update scheme with a parameter setting suited also for a very large population size. 2) Reproduce the effects from [14], observing significant improvements on functions, where a small number of directions need a small variance (here, we want to improve over the $(\mu/\mu_\mathrm{w}, \lambda)$-CMA-ES). 3) Survey the performance of a negative update on a larger number of diverse functions to possibly detect failures of the method. For this survey the BBOB-2010[4] test environment is used.

2. THE ALGORITHM

The $(\mu/\mu_\mathrm{w}, \lambda)$-aCMA-ES is summarized in Figure 1. The initial values $\mathbf{m}^{t=0} \in \mathbb{R}^D$ and $\sigma^{t=0} > 0$ are user defined and given in the next section. Modifications compared to [10, 6] are the addition of (7) and otherwise highlighted in pink (mainly the supplement to (8)).

The default parameter values are shown in Table 1. For $c^- = 0$, the $(\mu/\mu_\mathrm{w}, \lambda)$-CMA-ES is recovered.

Restarts. We apply nine restarts each with maximal $100 + 50(D+3)^2/\sqrt{\lambda}$ iterations. The default termination methods are used, besides that we have set `TolHistFun=1e-12`, `TolX=2e-12`, `StopOnStagnation='on'` and `MaxFunEvals=inf`, following [6]. The population size λ is doubled for each restart, the first value is given in Table 1.

3. PARAMETER TUNING AND SETUP

The new parameter(s) for aCMA have been identified with experiments on the sphere function f_1 with various initial covariance matrices. In particular c^- is chosen such that (a) decreasing c^- from the default value does not improve the

`strategies#Covariance_Matrix_Adaptation_Evolution_Strategies_CMA-ESs`

[4] `http://coco.gforge.inria.fr/doku.php?id=bbob-2010`

Table 1: Default parameter values of $(\mu/\mu_{\mathrm{w}}, \lambda)$-aCMA-ES, where by definition $\sum_{i=1}^{\mu} |w_i| = 1$ and $\mu_{\mathrm{w}}^{-1} = \sum_{i=1}^{\mu} w_i^2$ and $a - b \wedge c + d := \min(a - b, c + d)$. Only population size λ is possibly left to the users choice (see also [1, 6])

λ	=	$4 + \lfloor 3 \ln D \rfloor$	population size, number of newly sampled candidate solutions in each iteration
μ	=	$\lfloor \frac{\lambda}{2} \rfloor$	parent number, number of candidate solution used to update the distribution parameters
w_i	=	$\frac{\ln\left(\frac{\lambda+1}{2}\right) - \ln i}{\sum_{j=1}^{\mu}\left(\ln\left(\frac{\lambda+1}{2}\right) - \ln j\right)}$	recombination weights for $i = 1, \ldots, \mu \leq \lambda/2$
c_{m}	=	1	learning rate for the mean, sometimes interpreted as rescaled mutation with $\kappa = \frac{1}{c_{\mathrm{m}}} \geq 1$
c_σ	=	$\frac{\mu_{\mathrm{w}}+2}{D+\mu_{\mathrm{w}}+3}$	cumulation constant for step-size
d_σ	=	$1 + c_\sigma + 2 \max\left(0, \sqrt{\frac{\mu_{\mathrm{w}}-1}{D+1}} - 1\right)$	step-size damping, is usually close to one. This formula might be replaced by $2\mu_{\mathrm{w}}/\lambda + 0.3 + c_\sigma$ in the near future
c_{c}	=	$\frac{4+0\times\mu_{\mathrm{w}}/D}{D+4+0\times 2\mu_{\mathrm{w}}/D}$	cumulation constant for $\mathbf{p}_{\mathrm{c}}$, the $0\times$ might be removed in near future
c_1	=	$\frac{\alpha_{\mathrm{cov}} \min(1, \lambda/6)}{(D+1.3)^2+\mu_{\mathrm{w}}}$	covariance matrix learning rate for the rank one update using $\mathbf{p}_{\mathrm{c}}$
c_μ	=	$1 - c_1 \wedge \alpha_{\mathrm{cov}} \frac{\mu_{\mathrm{w}}-2+1/\mu_{\mathrm{w}}}{(D+2)^2+\alpha_{\mathrm{cov}}\mu_{\mathrm{w}}/2}$	covariance matrix learning rate for rank-μ update
c^-	=	$c^-_{\min} \wedge (1 - c_\mu)\frac{\alpha_{\mathrm{cov}}}{8}\frac{\mu_{\mathrm{w}}}{(D+2)^{1.5}+2\mu_{\mathrm{w}}}$	remark that c^- becomes small, if c_μ gets close to one
α_{cov}	=	2	could be chosen < 2, e.g. $\alpha_{\mathrm{cov}} = 0.5$ for noisy problems
$c^-_{\min}$	=	$\alpha^-_{\min}\frac{(1-0c_1-c_\mu)(1-\lambda_{\mathrm{mintarget}})}{\lambda_{\max}(\mathbf{C}^{t-1/2}\mathbf{C}^-_\mu\mathbf{C}^{t-1/2})}$	with $\alpha^-_{\min} = \infty$, $\lambda_{\mathrm{mintarget}} = 0.66$ and $\lambda_{\max}(.)$ is the largest eigenvalue; the coefficient $c^-_{\min}$ depends in general on the iteration index, but here, due to the setting of $\alpha^-_{\min}$, no upper bound on c^- is enforced
α^-_{old}	=	0.5	is in $[0, 1]$ and is chosen in the domain middle without deep motives, because it seems quite irrelevant
$\Delta_\sigma^{\max}$	=	∞	might become 1 in near future, which has only a negligible effect under neutral selection, because the term to the right of the $\wedge$ in (9) is approximately $\frac{c_\sigma}{d_\sigma}\mathcal{N}(0, 1/2D)$ under neutral selection

performance, (b) increasing c^- by a factor of two never leads to a failure, and (c) the condition number of $\mathbf{C}^t$ remains below 10 in the stationary limit. The remaining parameters are used in their default settings as found in the source code on the WWW or in [6]. No special attempt has been made to further tune the parameters to the BBOB testbed. The crafting-effort [7] of both algorithms is equal to CrE = 0.

Experiments were conducted according to BBOB-2010 [7] on the 24 benchmark functions given in [4, 9]. On each function, 15 trials are executed (on the first 15 instances) with $\mathbf{m}^0$ uniformly random in $[-4, 4]^D$ and $\sigma^0 = 2$.

Source code to reproduce the experiment is provided at[5].

CPU Timing Experiment. The complete algorithm was run on f_8 for at least 30 seconds on a Intel Core 2 6700 processor (2.66 GHz) with Linux 2.6.28-18 and Matlab R2008a. Results for the IPOP-aCMA-ES are 2.0; 1.7; 1.4; 1.2; 1.1; 1.4 and 3.6 $\times 10^{-4}$ seconds per function evaluation for dimension 2; 3; 5; 10; 20; 40 and 80.

4. RESULTS

We show results for two algorithms: the IPOP-aCMA-ES as presented above and denoted as aCMA, and the IPOP-CMA-ES, the same algorithm where c^- is set to zero, denoted as CMA. In exploratory experiments comparing both $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES and $(\mu/\mu_{\mathrm{w}}, \lambda)$-aCMA-ES with the result of the $(\mu/\mu_{\mathrm{I}}, \lambda)$-variants from [14], the $(\mu/\mu_{\mathrm{w}}, \lambda)$-variants perform at least as good and usually better: as a rule, $(\mu/\mu_{\mathrm{w}}, \lambda)$-aCMA-ES outperforms $(\mu/\mu_{\mathrm{I}}, \lambda)$-aCMA-ES that outperforms $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES that outperforms $(\mu/\mu_{\mathrm{I}}, \lambda)$-CMA-ES.

Runtime results comparing aCMA with CMA and with the respective best algorithm from BBOB-2009 are presented

[5] `http://coco.gforge.inria.fr/doku.php?id=bbob-2010-results`

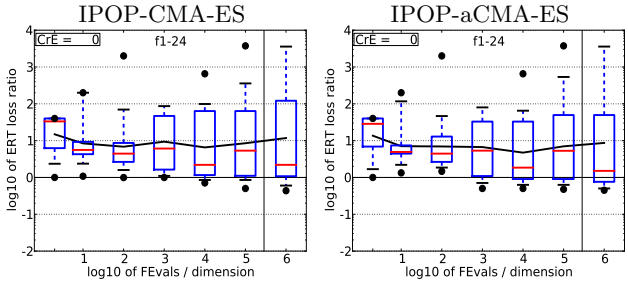

Figure 2: Box-Whisker plots of the ERT ratio to the respective best algorithm from BBOB-2009 for all functions in dimension 20

in Figures 2–5 and in Table 2. The **expected running time (ERT)**, used in the figures and table, depends on a given target function value, $f_{\mathrm{t}} = f_{\mathrm{opt}} + \Delta f$, and is computed over all relevant trials as the number of function evaluations executed during each trial while the best function value did not reach f_{t}, summed over all trials and divided by the number of trials that actually reached f_{t} [7, 16]. **Statistical significance** is tested with the rank-sum test for a given target Δf_{t} (10^{-8} in Figure 5) up to the smallest number of function evaluations, $b^{\mathrm{u}}_{\min}$, executed in any unsuccessful trial under consideration. The datum from each trial is either the best achieved Δf-value, or if f_{t} was reached within the budget $b^{\mathrm{u}}_{\min}$, the number of needed function evaluations to reach Δf_{t} (inverted and multiplied by -1),

In the following, when a performance difference is highlighted on an individual function, the difference is statistically significant.

Figure 2. The figure shows Box-Whisker plots of the ERT ratio compared to the respective best algorith from BBOB-

2009 for all functions. Both algorithms show a very similar characteristic. Besides for very small budgets, the median (horizontal red line) and average (connected line) ERT ratio are somewhat below ten. In the final stage, aCMA improves in about 25% of the functions compared to the best algorithm from 2009 (ratio smaller than one).

Figure 3. The figure shows empirical cumulative distributions (a) of the runtime in number of function evaluations and (b) of the runtime ratio between the two algorithms aCMA/CMA. Both algorithms perform very similar while aCMA appears to be slightly faster. Clearly, the strongest effect in Fig. 3 is observed for the ill-conditioned functions in dimension 20. The runtime is consistently shorter for aCMA, almost uniformly by a factor of close to two. The improvement is statistically significant on all functions (see Table 2).

On the weak structure functions, the observed differences are probably caused by stochastic deviations and not significant. The remaining subgroups show almost identical behavior.

Figure 4. The scatter plots in Fig. 4 visualize the ratio of expected runtime aCMA/CMA for each measurement on each function and each dimension. A slightly improved scaling with aCMA can be observed on f_2, f_{10} and f_{11}. On the Discus function f_{11} the effect is most pronounced and the speed-up comes close to a factor of three in dimension 40.

The advantage of CMA on f_{23} in dimension 10 is far beyond the maximum number of function evaluations, based on two successes (see Fig. 5) and not statistically significant.

Additionally, on functions f_7, f_{12}–f_{14}, and f_{18} an advantage by aCMA, in particular for larger dimension, could be conjectured. On f_{16} an advantage of CMA could be conjectured. According to Table 2, in dimension 20, statistical significance is established for f_7 and f_{12}–f_{14}.

Figure 5. The questions of a significant performance difference can be pursued in Fig. 5 which plots the ERT ratio versus the target function value. The figure reveals more statistically significant (but sometimes small) differences in performance. A non-negligible and statistically significant performance advantage of aCMA in some dimensions larger than 3 can be found on f_2, and f_7–f_{14}. No such advantage is detected for CMA.

Table 2. Finally, Table 2 presents the ERT numbers for dimension 5 and 20 in comparison with the respective best algorithm of BBOB-2009. Again, aCMA is significantly better than CMA on f_2, and f_7–f_{14} in dimension 20 as well as in dimension 5 apart from f_7 and f_{12}.

Compared to the best algorithm from BBOB-2009, chosen for each function respectively, the aCMA can improve the record in dimension 20 on f_{10}, f_{11}, f_{14}, f_{15} and f_{19}. On further four functions aCMA is visibly better, but the results appear not to be statistically significant. The CMA improves the record on f_{15} and f_{19}.

5. SUMMARY AND CONCLUSION

We have introduced the weighted negative update of the covariance matrix into the $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES with weighted recombination, denoted as $(\mu/\mu_{\mathrm{w}}, \lambda)$-aCMA-ES. The negative update is based in the idea of [14]. It provides a similar effect for the $(\mu/\mu_{\mathrm{w}}, \lambda)$-CMA-ES as [14] for the $(\mu/\mu_{\mathrm{I}}, \lambda)$-CMA-ES. The benchmarking of the new IPOP-aCMA-ES with BBOB-2010 reveals no deficiencies or failures compared to the IPOP-CMA-ES on the testbed of 24 functions in any dimension between 2 and 40.

The most prominent effect from aCMA is observed on ill-conditioned functions. In dimension 20, the average runtime advantage on the ill-conditioned functions is about 1.7 (CMA is 1.7 times slower than aCMA). On three ill-conditioned functions the scaling behavior compared to the CMA improves notably. In no case CMA outperformed aCMA with statistical significance.

Overall, the aCMA improves the performance over CMA on nine out of 12 essentially unimodal functions significantly at least in dimension 20, and the advantages are more pronounced in larger dimension. Finally, IPOP-aCMA-ES improves the record on five functions, where it is faster than the respective best algorithm from BBOB-2009 in dimension 20 with a sufficient statistical significance. In two of these cases IPOP-CMA-ES is still slightly faster and the record cannot be attributed to the negative covariance matrix update.

Acknowledgments

NH likes to thank M. Schoenauer for his support of BBOB.

6. REFERENCES

[1] A. Auger and N. Hansen. A restart CMA evolution strategy with increasing population size. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005)*, pages 1769–1776. IEEE Press, 2005.

[2] A. Auger, N. Hansen, J. Perez Zerpa, R. Ros, and M. Schoenauer. Experimental comparisons of derivative free optimization algorithms. In J. Vahrenhold, editor, *8th International Symposium on Experimental Algorithms*, volume 5526 of *LNCS*, pages 3–15. Springer, 2009.

[3] H.-G. Beyer. Evolution strategies. *Scholarpedia*, 2(8):1965, 2007.

[4] S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Research Center PPE, 2009. Updated February 2010.

[5] S. García, D. Molina, M. Lozano, and F. Herrera. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. *Journal of Heuristics*, 15(6):617–644, 2009.

[6] N. Hansen. Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In Rothlauf [17], pages 2389–2396.

[7] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Technical Report RR-7215, INRIA, 2010.

[8] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In J. Branke, editor, *GECCO (Companion)*. ACM, 2010.

[9] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Technical Report RR-6829, INRIA, 2009. Updated February 2010.

[10] N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao et al., editors, *Parallel Problem Solving from Nature PPSN VIII*, volume 3242 of *LNCS*, pages 282–291. Springer, 2004.

[11] N. Hansen, S. Muller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003.

[12] N. Hansen and A. Ostermeier. Completely derandomized

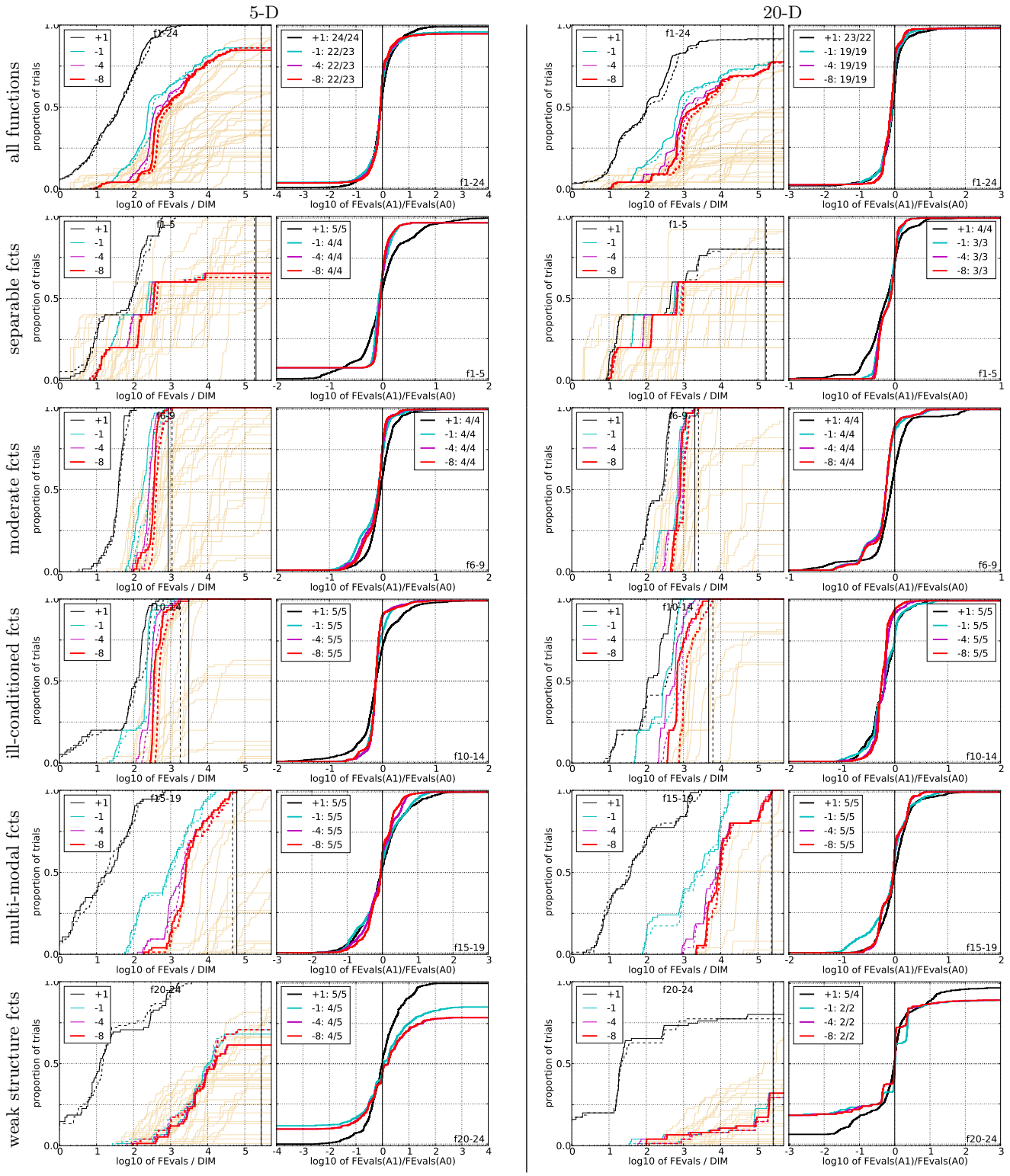

Figure 3: Empirical cumulative distributions (ECDF) of run lengths and speed-up ratios in 5-D (left) and 20-D (right). Left sub-columns: ECDF of the number of function evaluations divided by dimension D (FEvals/D) to reach a target value $f_{\mathrm{opt}} + \Delta f$ with $\Delta f = 10^k$, where $k \in \{1, -1, -4, -8\}$ is given by the first value in the legend, for IPOP-aCMA (solid) and IPOP-CMA (dashed). Light beige lines show the ECDF of FEvals for target value $\Delta f = 10^{-8}$ of algorithms benchmarked during BBOB-2009. Right sub-columns: ECDF of FEval ratios of IPOP-aCMA divided by IPOP-CMA, all trial pairs for each function. Pairs where both trials failed are disregarded, pairs where one trial failed are visible in the limits being > 0 or < 1. The legends indicate the number of functions that were solved in at least one trial (IPOP-aCMA first).

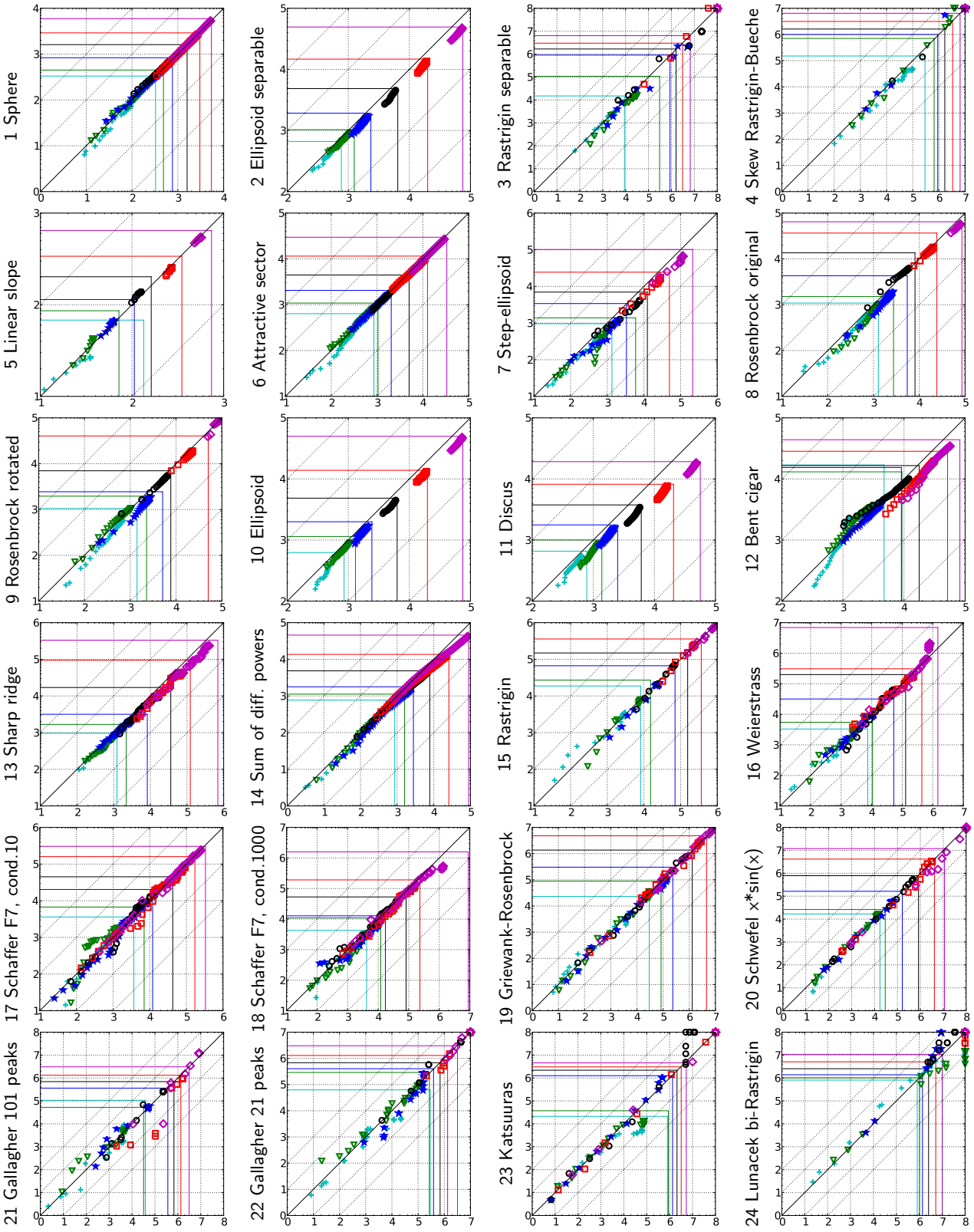

Figure 4: Expected running time (ERT in log10 of number of function evaluations) of IPOP-aCMA versus IPOP-CMA for 46 target values $\Delta f \in [10^{-8}, 10]$ in each dimension for functions f_1–f_{24}. Markers on the upper or right egde indicate that the target value was never reached by IPOP-aCMA or IPOP-CMA respectively. Markers represent dimension: 2:+, 3:▽, 5:⋆, 10:○, 20:□, 40:◇. Lines indicate the maximum number of function evaluations

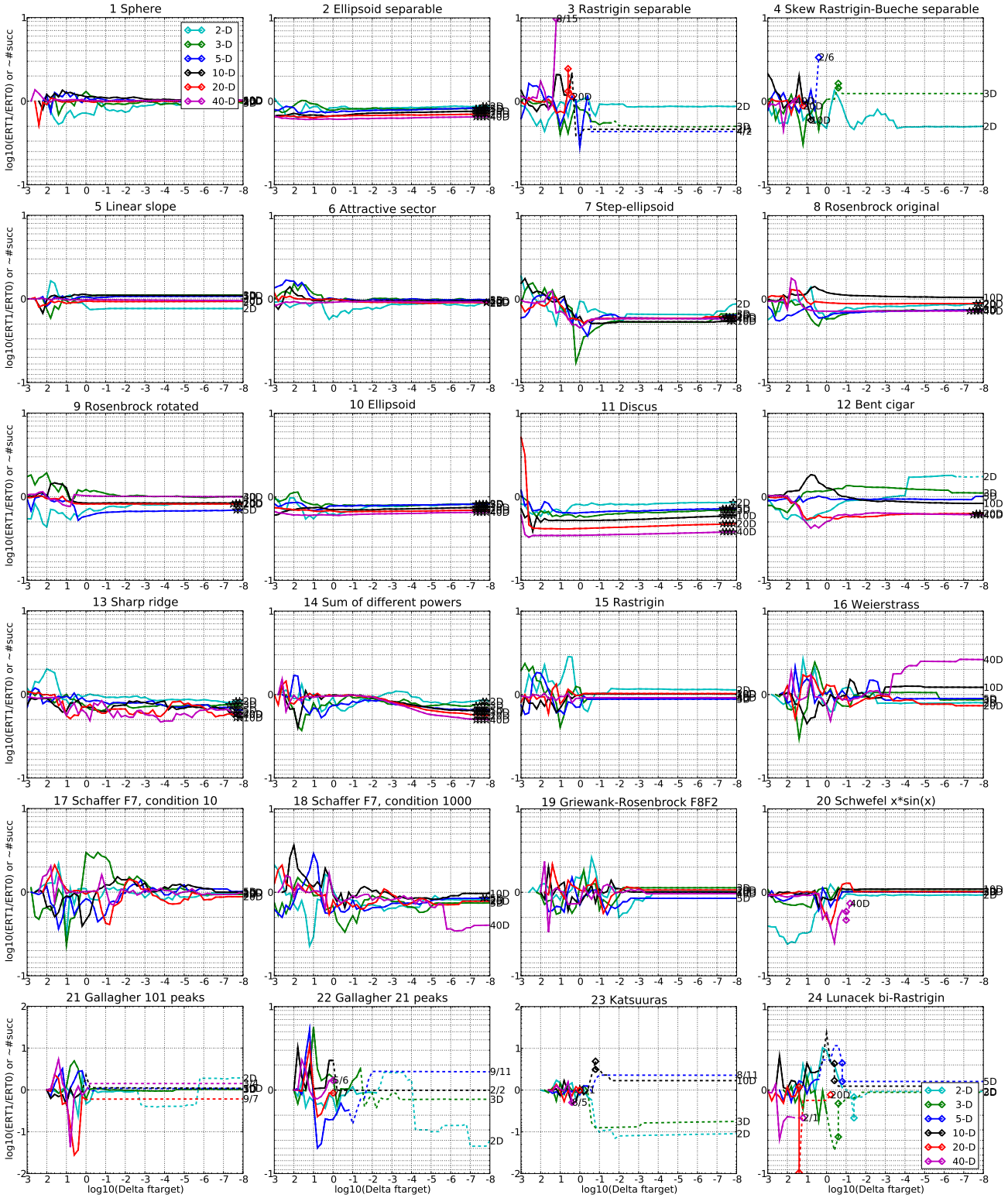

Figure 5: ERT ratio of IPOP-aCMA divided by IPOP-CMA versus $\log_{10}(\Delta f)$ for f_1–f_{24} in 2, 3, 5, 10, 20, 40-D. Ratios $< 10^0$ indicate an advantage of IPOP-aCMA, smaller values are always better. The line gets dashed when for any algorithm the ERT exceeds thrice the median of the trial-wise overall number of f-evaluations for the same algorithm on this function. Symbols indicate the best achieved Δf-value of one algorithm (ERT gets undefined to the right). The dashed line continues as the fraction of successful trials of the other algorithm, where 0 means 0% and the y-axis limits mean 100%, values below zero for IPOP-aCMA. The line ends when no algorithm reaches Δf anymore. The number of successful trials is given, only if it was in $\{1 \ldots 9\}$ for IPOP-aCMA (1st number) and non-zero for IPOP-CMA (2nd number). Results are significant with $p = 0.05$ for one star and $p = 10^{-\#\star}$ otherwise, with Bonferroni correction within each figure.

5-D

Δf	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
f_1	11	12	12	12	12	12	15/15
0: IPOP-CMA	2.5	8.0	14	27	39	51	15/15
1: IPOP-aCMA	3.2	8.9	15	27	39	51	15/15
f_2	83	87	88	90	92	94	15/15
0: IPOP-CMA	14	16	18	19	21	22	15/15
1: IPOP-aCMA	10	**12***	**14**$^{\star 3}$	**15**$^{\star 3}$	**16**$^{\star 3}$	**18**$^{\star 3}$	15/15
f_3	716	1622	1637	1646	1650	1654	15/15
0: IPOP-CMA	2.2	70	3130	3113	3106	3099	2/15
1: IPOP-aCMA	1.1	20	1359	1353	1350	1348	4/15
f_4	809	1633	1688	1817	1886	1903	15/15
0: IPOP-CMA	2.0	∞	∞	∞	∞	∞*8.5e5*	0/15
1: IPOP-aCMA	1.8	∞	∞	∞	∞	∞*8.7e5*	0/15
f_5	10	10	10	10	10	10	15/15
0: IPOP-CMA	4.6	6.0	6.3	6.3	6.3	6.3	15/15
1: IPOP-aCMA	4.6	6.3	6.8	6.8	6.8	6.8	15/15
f_6	114	214	281	580	1038	1332	15/15
0: IPOP-CMA	2.5	2.1	2.2	1.7	1.3	1.2	15/15
1: IPOP-aCMA	2.5	2.1	2.2	1.6	1.2	1.2	15/15
f_7	24	324	1171	1572	1572	1597	15/15
0: IPOP-CMA	4.4	1.7	1.2	1.2	1.2	1.2	15/15
1: IPOP-aCMA	4.0	0.87	0.70	0.69	0.69	0.70	15/15
f_8	73	273	336	391	410	422	15/15
0: IPOP-CMA	3.5	4.8	5.3	5.6	5.8	6.1	15/15
1: IPOP-aCMA	2.8	3.0	3.6	4.0	4.2	**4.5***	15/15
f_9	35	127	214	300	335	369	15/15
0: IPOP-CMA	6.0	11	8.7	7.5	7.3	7.2	15/15
1: IPOP-aCMA	5.4	6.2	5.7	5.0	5.0	**4.9***	15/15
f_{10}	349	500	574	626	829	880	15/15
0: IPOP-CMA	3.6	2.9	2.7	2.8	2.3	2.3	15/15
1: IPOP-aCMA	**2.5***	**2.2***	**2.1**$^{\star 2}$	**2.2**$^{\star 3}$	**1.8**$^{\star 3}$	**1.9**$^{\star 3}$	15/15
f_{11}	143	202	763	1177	1467	1673	15/15
0: IPOP-CMA	8.6	7.3	2.1	1.6	1.4	1.3	15/15
1: IPOP-aCMA	**5.6**$^{\star 3}$	**4.7**$^{\star 3}$	**1.4**$^{\star 3}$	**1.0**$^{\star 3}$	**0.95**$^{\star 3}$	**0.92**$^{\star 3}$	15/15
f_{12}	108	268	371	461	1303	1494	15/15
0: IPOP-CMA	9.4	6.1	6.2	6.3	2.8	2.8	15/15
1: IPOP-aCMA	8.8	5.9	5.7	6.0	2.6	2.6	15/15
f_{13}	132	195	250	1310	1752	2255	15/15
0: IPOP-CMA	3.1	5.0	5.3	1.4	1.6	1.6	15/15
1: IPOP-aCMA	3.0	4.1	4.2	1.2	**1.2**$^{\star 2}$	**1.1**$^{\star 2}$	15/15
f_{14}	10	41	58	139	251	476	15/15
0: IPOP-CMA	2.2	2.9	3.8	4.7	5.4	4.4	15/15
1: IPOP-aCMA	1.5	2.2	3.2	3.6	**3.8**$^{\star 3}$	**2.9**$^{\star 3}$	15/15
f_{15}	511	9310	19369	20073	20769	21359	14/15
0: IPOP-CMA	2.3	1.3	1.2	1.2	1.2	1.2	15/15
1: IPOP-aCMA	1.5	0.89	1.0	1.0	1.0	1.0	15/15
f_{16}	120	612	2662	10449	11644	12095	15/15
0: IPOP-CMA	2.5	2.3	1.7	0.96	0.94	0.95	15/15
1: IPOP-aCMA	3.9	2.4	1.7	0.82	0.84	0.85	15/15
f_{17}	5.2	215	899	3669	6351	7934	15/15
0: IPOP-CMA	4.8	1.1	0.97	0.77	0.81	1.0	15/15
1: IPOP-aCMA	4.3	0.89	0.53	0.77	1.00	1.1	15/15
f_{18}	103	378	3968	9280	10905	12469	15/15
0: IPOP-CMA	1.2	2.7	0.87	1.0	1.0	0.99	15/15
1: IPOP-aCMA	3.5	1.6	0.70	0.77	0.80	0.84	15/15
f_{19}	1	1	242	1.20e5	1.21e5	1.22e5	15/15
0: IPOP-CMA	21	1720	125	1.1	1.1	1.1	15/15
1: IPOP-aCMA	14	1207	123	0.95	0.96	0.96	15/15
f_{20}	16	851	38111	54470	54861	55313	14/15
0: IPOP-CMA	3.9	11	1.4	1.1	1.1	1.1	15/15
1: IPOP-aCMA	3.9	10	1.4	1.1	1.1	1.1	15/15
f_{21}	41	1157	1674	1705	1729	1757	14/15
0: IPOP-CMA	6.3	5.6	30	31	31	31	14/15
1: IPOP-aCMA	3.5	7.3	32	33	33	33	14/15
f_{22}	71	386	938	1008	1040	1068	14/15
0: IPOP-CMA	12	48	166	161	158	155	11/15
1: IPOP-aCMA	8.8	21	65	270	262	257	9/15
f_{23}	3.0	518	14249	31654	33030	34256	15/15
0: IPOP-CMA	2.2	26	33	15	14	14	11/15
1: IPOP-aCMA	1.6	20	76	34	33	32	8/15
f_{24}	1622	2.16e5	6.36e6	9.62e6	1.28e7	1.28e7	3/15
0: IPOP-CMA	2.9	18	1.4	0.94	0.70	0.70	2/15
1: IPOP-aCMA	2.6	41	∞	∞	∞	∞*1.3e6*	0/15

20-D

Δf	1e+1	1e+0	1e-1	1e-3	1e-5	1e-7	#succ
f_1	43	43	43	43	43	43	15/15
0: IPOP-CMA	8.0	14	20	33	46	58	15/15
1: IPOP-aCMA	7.9	14	20	33	45	58	15/15
f_2	385	386	387	390	391	393	15/15
0: IPOP-CMA	35	41	43	45	47	48	15/15
1: IPOP-aCMA	**22**$^{\star 3}$	**27**$^{\star 3}$	**29**$^{\star 3}$	**31**$^{\star 3}$	**33**$^{\star 3}$	**34**$^{\star 3}$	15/15
f_3	5066	7626	7635	7643	7646	7651	15/15
0: IPOP-CMA	13	∞	∞	∞	∞	∞*2.9e6*	0/15
1: IPOP-aCMA	10	∞	∞	∞	∞	∞*2.9e6*	0/15
f_4	4722	7628	7666	7700	7758	1.41e5	9/15
0: IPOP-CMA	∞	∞	∞	∞	∞	∞*2.8e6*	0/15
1: IPOP-aCMA	∞	∞	∞	∞	∞	∞*2.8e6*	0/15
f_5	41	41	41	41	41	41	15/15
0: IPOP-CMA	5.8	6.5	6.7	6.7	6.7	6.7	15/15
1: IPOP-aCMA	5.1	6.2	6.2	6.2	6.2	6.2	15/15
f_6	1296	2343	3413	5220	6728	8409	15/15
0: IPOP-CMA	1.7	1.3	1.2	1.2	1.2	1.2	15/15
1: IPOP-aCMA	1.6	1.3	1.1	1.1	**1.1***	1.1	15/15
f_7	1351	4274	9503	16524	16524	16969	15/15
0: IPOP-CMA	1.9	4.8	2.7	1.7	1.7	1.6	15/15
1: IPOP-aCMA	1.6	**2.7**$^{\star 2}$	**1.6**$^{\star 2}$	**0.99**$^{\star 2}$	**0.99**$^{\star 2}$	**1.0**$^{\star 2}$	15/15
f_8	2039	3871	4040	4219	4371	4484	15/15
0: IPOP-CMA	3.7	3.9	4.2	4.4	4.4	4.5	15/15
1: IPOP-aCMA	3.5	**3.5***	**3.7***	**3.9***	**3.9***	**4.0***	15/15
f_9	1716	3102	3277	3455	3594	3727	15/15
0: IPOP-CMA	4.6	5.7	6.0	6.1	6.1	6.1	15/15
1: IPOP-aCMA	4.1	**4.6**$^{\star 2}$	**4.9**$^{\star 2}$	**5.0**$^{\star 2}$	**5.0**$^{\star 2}$	**5.0**$^{\star 2}$	15/15
f_{10}	7413	8661	10735	14920	17073	17476	15/15
0: IPOP-CMA	1.8	1.8	1.5	1.2	1.1	1.1	15/15
1: IPOP-aCMA	**1.2**$^{\star 3}$	**1.2**$^{\star 3}$	**1.0**$^{\star 3}$	**0.80**$^{\star 3\downarrow 4}$	**0.73**$^{\star 3\downarrow 4}$	**0.75**$^{\star 3\downarrow 4}$	15/15
f_{11}	1002	2228	6278	9762	12285	14831	15/15
0: IPOP-CMA	11	5.4	2.1	1.4	1.2	1.1	15/15
1: IPOP-aCMA	**4.5**$^{\star 3}$	**2.3**$^{\star 3}$	**0.87**$^{\star 3}$	**0.64**$^{\star 3\downarrow 4}$	**0.56**$^{\star 3\downarrow 4}$	**0.50**$^{\star 3\downarrow 4}$	15/15
f_{12}	1042	1938	2740	4140	12407	13827	15/15
0: IPOP-CMA	4.8	5.3	5.5	5.1	2.1	2.2	15/15
1: IPOP-aCMA	2.6	3.0	3.2	3.1	1.3	**1.4***	15/15
f_{13}	652	2021	2751	18749	24455	30201	15/15
0: IPOP-CMA	6.5	4.8	6.2	1.4	1.7	2.3	15/15
1: IPOP-aCMA	3.6	3.4	3.7	0.80	1.3	**1.3***	15/15
f_{14}	75	239	304	932	1648	15661	15/15
0: IPOP-CMA	3.7	2.8	3.6	3.9	6.0	1.2	15/15
1: IPOP-aCMA	3.6	2.7	3.5	**3.2**$^{\star 3}$	**3.9**$^{\star 3}$	**0.67**$^{\star 3\downarrow 4}$	15/15
f_{15}	30378	1.47e5	3.12e5	3.20e5	4.49e5	4.59e5	15/15
0: IPOP-CMA	1.1	1.1	0.69	0.70	$0.52^{\downarrow 2}$	$0.53^{\downarrow 2}$	15/15
1: IPOP-aCMA	0.82	1.1	0.71	0.72	$0.53^{\downarrow 2}$	$0.54^{\downarrow 2}$	15/15
f_{16}	1384	27265	77015	1.88e5	1.98e5	2.20e5	15/15
0: IPOP-CMA	1.7	0.81	0.92	0.84	1.1	1.0	15/15
1: IPOP-aCMA	2.8	1.1	0.88	0.80	0.82	0.76	15/15
f_{17}	63	1030	4005	30677	56288	80472	15/15
0: IPOP-CMA	2.1	0.94	1.2	0.76	0.99	1.0	15/15
1: IPOP-aCMA	2.3	0.89	0.50	0.82	0.83	0.87	15/15
f_{18}	621	3972	19561	67569	1.31e5	1.47e5	15/15
0: IPOP-CMA	1.1	1.8	1.1	0.97	1.0	1.1	15/15
1: IPOP-aCMA	1.2	1.5	0.75	0.91	0.78	0.83	15/15
f_{19}	1	1	3.43e5	6.22e6	6.69e6	6.74e6	15/15
0: IPOP-CMA	161	27333	0.71	$0.38^{\downarrow 3}$	$0.41^{\downarrow 3}$	$0.41^{\downarrow 3}$	15/15
1: IPOP-aCMA	166	29179	0.63	$0.43^{\downarrow 3}$	$0.44^{\downarrow 3}$	$0.44^{\downarrow 3}$	15/15
f_{20}	82	46150	3.10e6	5.54e6	5.59e6	5.64e6	14/15
0: IPOP-CMA	4.6	6.4	0.65	0.57	0.58	0.58	15/15
1: IPOP-aCMA	4.7	**3.2**$^{\star 3}$	0.83	0.58	0.59	0.60	15/15
f_{21}	561	6541	14103	14643	15567	17589	15/15
0: IPOP-CMA	3.7	139	110	106	100	88	7/15
1: IPOP-aCMA	1.9	81	66	64	60	54	9/15
f_{22}	467	5580	23491	24948	26847	1.35e5	12/15
0: IPOP-CMA	445	287	∞	∞	∞	∞*1.3e6*	0/15
1: IPOP-aCMA	462	264	∞	∞	∞	∞*1.3e6*	0/15
f_{23}	3.2	1614	67457	4.89e5	8.11e5	8.38e5	15/15
0: IPOP-CMA	4.3	23082	∞	∞	∞	∞*2.5e6*	0/15
1: IPOP-aCMA	4.1	22809	∞	∞	∞	∞*2.5e6*	0/15
f_{24}	1.34e6	7.48e6	5.19e7	5.20e7	5.20e7	5.20e7	3/15
0: IPOP-CMA	∞	∞	∞	∞	∞	∞*5.1e6*	0/15
1: IPOP-aCMA	25	4.5	∞	∞	∞	∞*5.0e6*	0/15

Table 2: Expected running time (ERT in number of function evaluations) divided by the best ERT measured during BBOB-2009 (given in the respective first row) for different Δf values for functions f_1–f_{24}. The median number of conducted function evaluations is additionally given in *italics*, if $\mathrm{ERT}(10^{-7}) = \infty$. #succ is the number of trials that reached the final target $f_{\mathrm{opt}} + 10^{-8}$. Bold entries are statistically significantly better compared to the other algorithm, with $p = 0.05$ or $p = 10^{-k}$ where $k > 1$ is the number following the $\star$ symbol, with Bonferroni correction of 48.

self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[13] 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 8th annual conference on genetic and evolutionary computation GECCO*, pages 453–460. ACM, 2006.

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

[15] S. Kern, S. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos. Learning probability distributions in continuous evolutionary algorithms–a comparative review. *Natural Computing*, 3(1):77–112, 2004.

[16] K. Price. Differential evolution vs. the functions of the second ICEO. In *Proc. of the IEEE International Congress on Evolutionary Computation*, pages 153–157, 1997.

[17] F. Rothlauf, editor. *Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12, 2009, Companion Material*. ACM, 2009.