Intensive Surrogate Model Exploitation in Self-adaptive Surrogate-assisted CMA-ES (saACM-ES)

Ilya Loshchilov
Laboratory of Intelligent Systems (LIS)
EPFL, Lausanne, Switzerland
ilya.loshchilov@epfl.ch

Marc Schoenauer
TAO, INRIA Saclay
Univ. Paris-Sud, Orsay, France
marc.schoenauer@inria.fr

Michèle Sebag
CNRS, LRI UMR 8623
Univ. Paris-Sud, Orsay, France
michele.sebag@inria.fr

ABSTRACT

This paper presents a new mechanism for a better exploitation of surrogate models in the framework of Evolution Strategies (ESs). This mechanism is instantiated here on the self-adaptive surrogate-assisted Covariance Matrix Adaptation Evolution Strategy (^{s*}ACM-ES), a recently proposed surrogate-assisted variant of CMA-ES. As well as in the original ^{s*}ACM-ES, the expensive function is optimized by exploiting the surrogate model, whose hyper-parameters are also optimized online. The main novelty concerns a more intensive exploitation of the surrogate model by using much larger population sizes for its optimization.

The new variant of ^{s*}ACM-ES significantly improves the original ^{s*}ACM-ES and further increases the speed-up compared to the CMA-ES, especially on unimodal functions (e.g., on 20-dimensional Rotated Ellipsoid, ^{s*}ACM-ES is 6 times faster than aCMA-ES and almost by one order of magnitude faster than CMA-ES). The empirical validation on the BBOB-2013 noiseless testbed demonstrates the efficiency and the robustness of the proposed mechanism.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Intelligence Problem Solving, Control Methods, and Search

General Terms

Algorithms

Keywords

Evolution Strategies, CMA-ES, self-adaptation, surrogate-assisted optimization, surrogate models, ranking support vector machine, black-box optimization

1. INTRODUCTION

Evolutionary Algorithms (EAs) have received a lot of attention regarding their potential to solve *black-box* optimization problems, where typically no additional information is available apart from the quality of evaluated solutions. By closely looking at the most successful EAs w.r.t. results on *both* artificial benchmark and real-world optimization problems, one may observe that invariance properties play a crucial role and represent a source of robustness. A good illustrative example is Covariance Matrix Adaptation Evolution Strategy (CMA-ES [10]) which exhibits invariance w.r.t. rank-preserving transformations of the objective function and invariance w.r.t. orthogonal transformations of the search space (if the initial search point(s) are transformed accordingly). The algorithm has demonstrated the state-of-the-art performance on benchmark problems (see, e.g., CEC-2005 [4], BBOB-2010 [8]) and real-world optimization problems [6].

One of the main limitations of EAs, the large number of function evaluations required for a reasonable accuracy of optimization, prevents EAs from being widely used on optimization problems which are expensive in terms of time (one evaluation takes several seconds or even hours) and/or in terms of money. This limitation is especially observable in the special, but quite common case of the unimodal noiseless continuous optimization, where gradient information is useful and quasi-Newton methods such as BFGS [28], proposed 40 years ago, usually outperform most of advanced EAs [8]. The latter is also the case for CMA-ES on some interesting (for benchmarking) optimization problems such as Rosenbrock function.

To address this limitation, numerous approaches to integrate surrogate/approximate models into EAs have been proposed (see, e.g., [15] for a recent overview of the state-of-the-art). In most of these approaches, a surrogate model $\hat{f}(\boldsymbol{x})$ (or simply $\hat{f}$) of the objective function $f(\boldsymbol{x})$ (or simply f) is built using the information about evaluated (*training*) solutions and corresponding objective values. Then, $\hat{f}$ is used to predict the quality of some promising *test* solutions in order to proceed towards the optimum of f faster in terms of number of function evaluations. The use of this approach usually leads to a loss of invariance w.r.t. rank-preserving transformations of f since the building of $\hat{f}$ is sensitive to these kinds of transformations. The property of invariance w.r.t. orthogonal transformations of the search space is also not preserved in most of the surrogate-assisted approaches, since the surrogate learning phase does not take into account the information about a coordinate system (if any) adapted during the search.

Until recently, surrogate-assisted CMA-ES algorithms preserved at most one of the above described invariance prop-

GECCO'13, July 6–10, 2013, Amsterdam, The Netherlands.

erties (see, e.g., [27, 17]). The first surrogate-assisted CMA-ES which preserves both invariance properties, referred to as ACM-ES, was proposed in [19]. Its extension, referred to as self-adaptive surrogate-assisted CMA-ES (^{s*}ACM-ES [21]), has demonstrated an ability to adapt during the search the hyper-parameters used to build the surrogate model, and this can be viewed as a sort of invariance. ^{s*}ACM-ES usually speed-ups the original CMA-ES by a factor between 2 and 4 on unimodal functions. However, it was surprisingly outperformed by ACM-ES algorithm on 10-dimensional Rosenbrock function [19]. An analysis of this observation led to an hypothesis that surrogate model exploitation is not sufficiently intensive in ^{s*}ACM-ES.

This paper investigates a simple mechanism of a more intensive surrogate model exploitation by using much larger population sizes, while keeping the default population size when optimizing the original expensive function. Such exploitation is useful when the surrogate model is sufficiently accurate and may lead to a certain divergence when the surrogate badly approximates the expensive function. Thus, a trade-off between between two cases should be respected.

The paper is organized as follows. First, Section 2 reviews surrogate-assisted Evolution Strategies and presents ^{s*}ACM-ES algorithm with a discussion of its main principles. Next, Section 3 introduces our new mechanism to surrogate model exploitation. The experimental validation of the proposed mechanism is reported and discussed in Section 4.1. Section 5 concludes the paper.

2. SURROGATE-ASSISTED EVOLUTION STRATEGIES

First, we discuss the various techniques used to learn and exploit surrogate models within Evolution Strategies. Then, we briefly describe ^{s*}ACM-ES algorithm whose mechanism of surrogate model exploitation will be further studied and modified in Section 3.

2.1 Background

One of the first surrogate-assisted $(\mu + \lambda)$-ES and (μ, λ)-ES were proposed in [23], where evaluations of an expensive structural optimization problem were replaced by a hidden-layer Artificial Neural Network (ANN) trained by back-propagation. The authors suggested to re-learn the model at each iteration by adding new training points randomly drawn from a Gaussian distribution with the mean located in the center of the decision space.

Jin et al. [16] proposed *individual-based* and *generation-based* evolution control strategies for surrogate-assisted search with CMA-ES. Individual-based strategy corresponds to a pre-selection strategy, where λ' random (*random strategy*) or best (*best strategy*) *controlled* individuals out of λ_{Pre} pre-selected individuals are evaluated with the expensive function f. In generation-based strategy, the whole population will be evaluated with the expensive function for η generations every κ generations, where $\eta \leq \kappa$. The authors found that in both cases about 50% of individuals should be evaluated in order to have a good final speed-up, but they employed only the generation-based strategy, because they found it more suitable for parallel implementation. It is suggested to set the fraction of generations $\frac{\eta}{\kappa}$ that $\hat{f}$ is optimized to be proportional to $\frac{E(k)}{E_{max}}$, where $E(k)$ and E_{max} are current and maximum model errors, respectively.

Emmerich et al. [3] studied Kriging model-based pre-selection strategy within (μ, κ, λ)-ES with $\mu = 15$, $\lambda = 100$ and $\kappa = 5$, where the individuals that exceed the age of $\kappa = 5$ generations are eliminated from the selection procedure. The authors suggested to evaluate the best $\lambda' = 10$ individuals w.r.t. a criterion based on both the estimated value $\hat{f}(\boldsymbol{x})$ and the estimated local standard deviation $\hat{\sigma}(\boldsymbol{x})$ of the prediction Kriging model as in [29]:

$$S_c(\boldsymbol{x}) = \hat{f}(\boldsymbol{x}) - \rho\hat{\sigma}(\boldsymbol{x}), \tag{1}$$

where $\rho \geq 0$ defines the selection trade-off between the most promising solutions with $\rho = 0$ and promising solutions in still unexplored search areas with $\rho > 0$. The experimental validation of the proposed surrogate-assisted algorithm showed that on unimodal functions the exploration ($\rho = 1$) does not harm, but leads to a better convergence on multimodal functions than $\hat{f}$-based selection ($\rho = 0$).

Ulmer et al. [31] studied RBF networks-based pre-selection strategy for (2,8)-ES with $\lambda_{Pre} = 30$ and (1+1)-ES with $\lambda_{Pre} = 10$ and concluded that the proposed meta-model assisted ES (MAES) performs better, especially on unimodal functions. Later, Ulmer et al. [30] analyzed GP-based pre-selection strategy similar to the one of [3] for CMA-ES, where an exploration-based selection criterion was chosen. The results confirmed the observations of [3] that on multimodal functions the exploration-based pre-selection should be preferred to "greedy" $\hat{f}$-based pre-selection. Ulmer et al. [32] further studied Support Vector Regression (SVR)-based pre-selection strategy for (μ, λ) Main Vector Adaptation (MVA [25], a CMA-ES variant with linear time and space complexity). They proposed to adjust λ_{Pre}, depending on a model quality measure similar to the one proposed by [14] and defined as summed rank of all correctly selected individuals. At each generation t, the actual model quality Q^t is compared to a quality Q^{rand} measured for the random model. The update procedure is controlled by a parameter $\delta_{\lambda_{Pre}}$ as follows:

$$\lambda_{Pre}^{t+1} = \begin{cases} \lambda_{Pre}^t + \frac{Q^{max} - Q^t}{Q^{max} - Q^{rand}}\delta_{\lambda_{Pre}} & \text{if } Q^t > Q^{rand} \\ \lambda_{Pre}^t - \frac{Q^{rand} - Q^t}{Q^{rand}}\delta_{\lambda_{Pre}} & \text{otherwise,} \end{cases} \tag{2}$$

where Q^{max} is the maximum possible quality. Experimental results of the proposed *C-MAES* on unimodal and multimodal functions confirmed that the adaptation works well even in dynamically changing noisy environment, where the speed-up up to a factor of 2 also can be achieved.

Runarsson [26] found that it is always desirable to evaluate with f at least one best point among λ evaluated with $\hat{f}$, and proposed *approximate ranking procedure* which suggests to evaluate with f several additional points only if the model changes its ranking prediction of λ points.

Poland [24] proposed an ES-like algorithm with quadratic meta-models, which showed the best results (in that time) in the literature for an Evolutionary Algorithm on popular Rosenbrock benchmark problem, where it outperformed CMA-ES by a factor of 4. Unfortunately, the algorithm is not invariant to rank-preserving transformations of f and its performance probably will degrade significantly if f is scaled differently (also true for all regression to value-based surrogate-assisted optimizers).

Büche et al. [2] proposed Gaussian Process Optimization Procedure (GPOP) which suggests to repeat the following

procedure. First, build a GP-based model and directly optimize Eq. (1) by CMA-ES. When a local optimum for a given ρ is found, it can be evaluated with f and added to the training set. The training set consists of the union of the N_C closest points to the current best solution and the N_R most recently evaluated points. The search space for each local search is constrained by the hyper-rectangle around the current best solution. The GPOP outperforms CMA-ES on Sphere and Schwefel functions, but already for Rosenbrock the speed-up is less than 2.0 for $n = 8, 16$ and less than 0.4 for $n = 32$, where larger training sets are probably needed. On multimodal Rastrigin function GPOP in most cases is outperformed by CMA-ES.

Runarsson [27] first proposed to use *ordinal regression* and Ranking Support Vector Machine (SVM) as surrogate models in Evolutionary Computation. The author studied the performance of Ranking SVM-based CMA-ES within the approximate ranking procedure and found that the surrogate-assisted version outperforms the original CMA-ES for $n \leq 5$, but shows no improvements for $n > 5$. The latter can be explained by the use of very small training set of only 60 training points. However, this paper has played an important role in the development of comparison-based surrogates (see, e.g., [19]).

Hoffmann and Holemann [12] suggested to keep λ_{Pre} fixed, but adapt λ using the following formula, similar to (2):

$$\lambda^{t+1} = \begin{cases} max(\lambda^t - \frac{Q^{max}-Q^t}{Q^{rand}}\delta_\lambda, \mu) & \text{if } Q^t > Q^{rand} \\ min(\lambda^t + \frac{Q^{max}-Q^t}{Q^{max}-Q^{rand}}\delta_\lambda, \lambda_{Pre}) & \text{otherwise,} \end{cases} \quad (3)$$

Thus, the better the model - the smaller λ^{t+1}, and vice versa. The results of the proposed λ-controlled surrogate-assisted λ-*CMA-ES* are similar, but not directly comparable to [32], where MVA was used as a baseline algorithm.

The local meta-model CMA-ES based on locally weighted regression was first proposed by [17] and later extended for large populations by [1]. In lmm-CMA-ES, the surrogate model exploitation is controlled by the *approximate ranking procedure* [26], which suggests to evaluate with f only a fraction of λ new individuals if the ranking of some of λ individuals on surrogate models of the current and previous generations remains unchanged. The algorithm and its extensions have demonstrated a relatively good performance, but the time complexity usually scales between $O(n^4)$ and $O(n^6)$ depending on the version of the algorithm [18], limiting the range of application to problems with $n \lesssim 15$ [18].

2.2 Self-adaptive Surrogate-assisted CMA-ES

In the following, we outline ^{s*}ACM-ES algorithm according to [21], see Figure 1.

In ^{s*}ACM-ES, two optimization procedures are performed:

1. Optimization of the objective function f using CMA-ES (so-called "CMA-ES #1" in Figure 1) assisted by a surrogate model $\widehat{f}_\alpha$, built using a vector α of some surrogate model (learning) hyper-parameters.
2. Optimization of the surrogate model error $\mathrm{Err}(\alpha)$ (using so-called "CMA-ES #2" in Figure 1) and corresponding surrogate model hyper-parameters α.

In both cases, the original *generation/iteration* procedure of CMA-ES is not modified, but called when necessary in order to change the state of CMA-ES in the search space, that also leads to a change of the internal parameters of CMA-ES (mean of mutation distribution $\boldsymbol{m}$, covariance matrix $\mathbf{C}$, step-size σ, evolution paths for σ and $\boldsymbol{m}$).

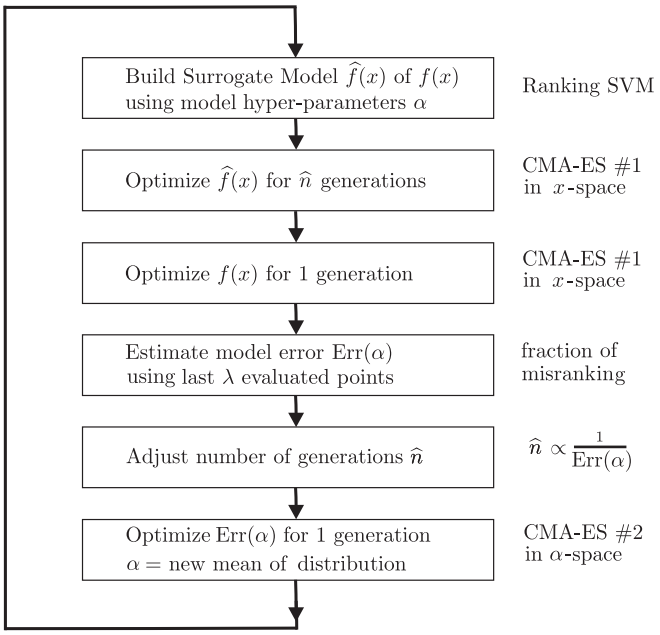

Figure 1: Optimization loop of the ^{s*}ACM-ES (adopted from [21]).

First, the algorithm proceeds the optimization of f for a given number of generations g_{start} till a sufficient number of evaluated points is stored to build the surrogate model $\widehat{f}_\alpha$. The model is built using Ranking SVM and model hyper-parameters α such that $\widehat{f}_\alpha$ predicts only the ordering of test points, this allows $\widehat{f}_\alpha$-assisted search to be invariant w.r.t. rank-preserving transformations of f. The second invariance property, invariance w.r.t. orthogonal transformations of the search space, is preserved thanks to the following transformation applied to each of the training and test points $\boldsymbol{x}$:

$$\boldsymbol{x}' = \mathbf{C}^{-1/2}(\boldsymbol{x} - \boldsymbol{m}), \quad (4)$$

where $\mathbf{C}$ is the covariance matrix adapted by CMA-ES.

When the model is built, it is optimized by CMA-ES for a given number of generations $\hat{n}$, where $\hat{n}$ is defined in [21] by a linear function to be inversely proportional to a *global* model error $\mathrm{Err}(\alpha)$. This global error is incrementally updated (with some relaxation factor) from a *local* error estimated with λ recently evaluated solutions stored in Λ as follows:

$$\mathrm{Err}(\alpha) = \frac{2}{|\Lambda|(|\Lambda|-1)} \sum_{i=1}^{|\Lambda|} \sum_{j=i+1}^{|\Lambda|} w_{ij} . 1_{\widehat{f}_\alpha, i, j} \quad (5)$$

where $1_{\widehat{f}_\alpha, i, j}$ holds true if $\widehat{f}_\alpha$ violates the f-based ordering constraint on pair (i, j) and w_{ij} defines the weight of this violation ($w_{ij} = 1$ by default).

First (see Figure 1), ^{s*}ACM-ES i). builds the surrogate model $\widehat{f}_\alpha$, ii). optimizes it for $\hat{n}$ generations, then iii). optimizes the objective function f for 1 generation, and iv). estimates the surrogated model error by computing with Eq. (5) a fraction of comparison relations which were incorrectly

predicted by $\widehat{f}_\alpha$ with λ recently evaluated solutions, and finally v). adjusts $\hat{n}$ for the next iteration.

An important novelty of ^{s*}ACM-ES, apart from the preservation of the invariance properties of CMA-ES, is an ability to adapt hyper-parameters of surrogate learning procedure during the optimization of the objective function f. This procedure corresponds to the last step, illustrated in Figure 1, where an additional instance of CMA-ES (initialed before the loop of Figure 1) performs one optimization generation in the space of surrogate model hyper-parameters. In this step, CMA-ES generates λ_{hyp} ($\lambda_{hyp} = 20$ by default) different α-vectors of hyper-parameters and builds λ_{hyp} corresponding surrogate models. These models are evaluated using $\mathrm{Err}(\alpha)$ and $\mu_{hyp} = \lfloor \lambda_{hyp}/2 \rfloor$ the most successful (with the smallest model error) out of λ_{hyp} vectors of hyper-parameters are used to updates internal parameters of CMA-ES. The new mean of the mutation distribution of CMA-ES is then used as an estimate of the optimum in the space of surrogate model hyper-parameters. This estimate is finally used as α to build the surrogate model in the next iteration of ^{s*}ACM-ES, gradually adapting the surrogate model to the local topography of the objective function.

Finally, the algorithm optimizes the objective function f together with the hyper-parameters used to build its surrogate model $\hat{f}$, that allows the user to define only the range of hyper-parameters and let ^{s*}ACM-ES to find their optimal values online.

3. NEW MECHANISM OF SURROGATE MODEL EXPLOITATION

As was mentioned before, in our experiments we observed that ACM-ES [19] outperforms ^{s*}ACM-ES on 10-dimensional Rosenbrock problem (but this is not the case in 20-D, where ^{s*}ACM-ES is better thanks to the adaptation of hyper-parameters). This result is quite surprising given that the surrogate model learning phases are very similar in both algorithms. These observations led to an hypothesis that the surrogate model exploitation is more intensive in ACM-ES on this particular problem and this is also beneficial because the surrogate is relatively accurate.

3.1 Intensive Exploitation

In the original ACM-ES algorithm, the surrogate model exploitation is independent on the model quality, that in certain cases may lead to a divergence of the algorithm when the surrogate provides almost random predictions. However, when the surrogate model is accurate enough, its exploitation by pre-selection substantially speed-ups CMA-ES [19]. In ACM-ES, $\lambda_{Pre} = 500$ individuals are evaluated with $\hat{f}$ and only $\lambda' = \left\lfloor \frac{\lambda_{default}}{3} \right\rfloor$ individuals among them ($\lambda_{default}$ is the default population size) are selected through a two-step selection procedure to be evaluated with the expensive function f. In ^{s*}ACM-ES, usually a smaller number of individuals than λ_{Pre} is evaluated with $\hat{f}$, but the change of the mean of the mutation distribution, and sometimes that is more important, of the step-size during $\hat{n}$ generations may lead to a larger divergence between the original and final multivariate normal distributions than with the pre-selection. Thus, ^{s*}ACM-ES is able to exploit $\hat{f}$ in the same way as CMA-ES exploits f, and if $\hat{f}$ ideally approximates f, then a speed-up of a factor of $\widehat{n_{max}}$ (the maximum number of generations $\hat{n}$ when $\mathrm{Err}(\alpha) = 0$) is expected.

The first attempt to increase the intensity of exploitation by increasing $\widehat{n_{max}}$ showed that larger values of $\widehat{n_{max}}$ lead to moderate improvements on unimodal functions, but may increase the chance of premature divergence when the surrogate provides random predictions since a larger number of generations $\hat{n}$ is allowed. When the suggested $\hat{n}$ is much larger than some expected speed-up k (e.g., by a factor larger than 3), the search with the surrogate may start to deteriorate after ca. k generations by slowly becoming a random walk and, thus, destroying the evolution paths of CMA-ES.

Since longer runs may deteriorate the search, more attention should be paid to a single generation with $\hat{f}$. In this generation, $\hat{f}$ can be more intensively exploited by estimating a local covariance matrix $\boldsymbol{C}_{loc}$ constructed from a large number of points (e.g., in order of 10^5), sampled by CMA-ES and ranked according to $\hat{f}$. This covariance matrix $\boldsymbol{C}_{loc}$ usually is very similar to the one of CMA-ES, which was used for sampling, but in the same time, it stores "all" the information about covariances available from $\hat{f}$ for a given scale defined by the step-size. The use of $\boldsymbol{C}_{loc}$ instead of the original estimate of CMA-ES allows to faster learn an appropriate covariance matrix, that in many cases leads to a faster convergence. However, the influence of $\boldsymbol{C}_{loc}$ is somehow limited by the learning rate used to adapt the covariance matrix $\boldsymbol{C}$ of CMA-ES. In the same time, this influence should be controlled by the model error in order to avoid a possible degradation of $\boldsymbol{C}$. The mechanism of this control might be very similar to the one already used in ^{s*}ACM-ES to control $\hat{n}$.

Finally, we found a very simple exploitation mechanism which solves the above described problems and generalizes the second and third steps shown in Figure 1:

1. Optimize $\hat{f}$ for $\hat{n}$ generations by CMA-ES with population size $\lambda = k_\lambda \lambda_{default}$ and number of parents $\mu = k_\mu \mu_{default}$, where $k_\lambda \geq 1$ and $k_\mu \geq 1$.
2. Optimize f for 1 generation by CMA-ES with population size $\lambda = \lambda_{default}$ and number of parents $\mu = \mu_{default}$.

This simple mechanism allows us to keep $\hat{n}$ in the order of the expected speed-up and, in the same time, to recover more information from $\hat{f}$ using a larger population size (when $k_\lambda > 1$). In contrast to the estimation of $\boldsymbol{C}_{loc}$, the mean of the mutation distribution also changes (as well as other internal parameters) taking into account the information from $\lambda = k_\lambda \lambda_{default}$ evaluated solutions. While, k_λ and k_μ can be adjusted depending on the model error, the control mechanism of $\hat{n}$ is already presented in ^{s*}ACM-ES.

Is should be noted that a more intensive exploitation occurs not when k_λ is large and $k_\mu = k_\lambda$, but rather when k_λ is large and $k_\lambda \gg k_\mu$ (e.g., $k_\mu = 1$).

3.2 Divergence Prevention

While the control mechanism of ^{s*}ACM-ES sets $\hat{n}$ to 0 when $\mathrm{Err}(\alpha)$ reaches some surrogate error threshold value τ_{err} (τ_{err}= 0.45 by default [21]), some oscillation between $\hat{n} = 0$ and $\hat{n} = 1$ is possible when $\mathrm{Err}(\alpha)$ is very close to the threshold. In this case, a more intensive exploitation of $\hat{f}$ with a larger k_λ might lead to a faster divergence of ^{s*}ACM-ES. The above described issue can be relevant for multimodal and noisy problems, where the surrogate model is typically inaccurate. To avoid the divergence, k_λ can be

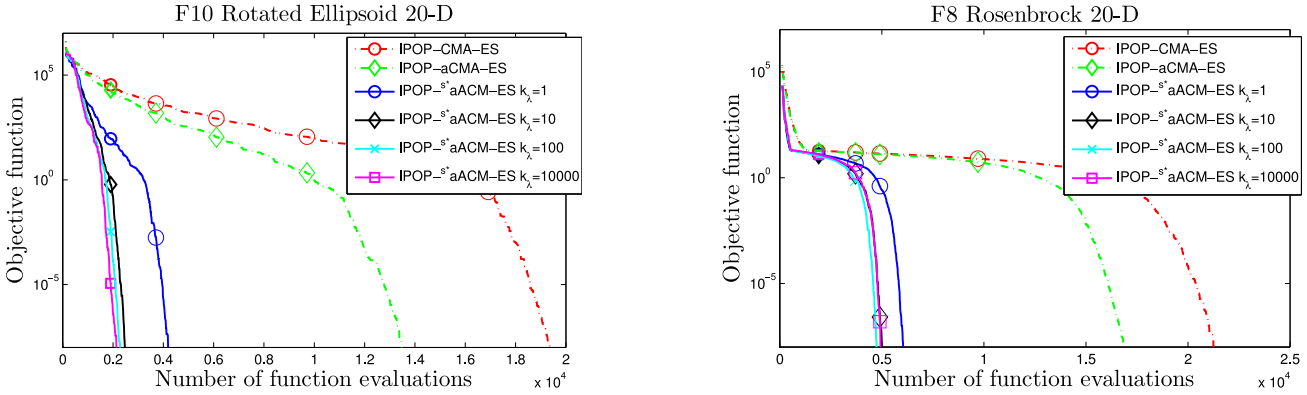

Figure 2: Comparison of the proposed surrogate-assisted versions of the original and active IPOP-CMA-ES algorithms on 20-dimensional Rotated Ellipsoid (Left) and Rosenbrock (Right) functions. The trajectories show the median of 15 runs.

adjusted according to $\mathrm{Err}(\alpha)$ or $\hat{n}$, e.g., k_λ should increase together with $\hat{n}$. Indeed, the optimal rule would be problem-dependent and is difficult to choose a priori, but probably can be learned online. In this paper, in order to avoid a possible overfitting, we study a simple rule which states that

- $k_\lambda > 1$ is used only if $\hat{n} \geq \hat{n}_{k_\lambda}$,

where $\hat{n}_{k_\lambda}$ is the number of generations which corresponds to a model error much smaller than τ_{err}, such that the model is "accurate enough" to be intensively exploited.

4. EXPERIMENTAL VALIDATION

This Section presents an experimental validation of the proposed mechanism to surrogate model exploitation on a set of noiseless black-box optimization problems from the BBOB-2012 framework [7]. We compare the original CMA-ES [9] and its so-called active version aCMA-ES [13, 11] in the IPOP and BIPOP scenarios of restarts [11, 5, 20] to the results of ^{s*}ACM-ES in its original version [21] and our modified version with the new mechanism of surrogate model exploitation. For all experiments, we use the Octave/Matlab source code provided[1] by the authors of [21] and, we make our modified version of this code also available online[2].

4.1 Experimental Setting

For all original algorithms the default parameters are used as given in [21]. In our modified version of ^{s*}ACM-ES, apart from the k_λ, k_μ and $\hat{n}_{k_\lambda}$, whose parameter settings will be discussed later in this Section, we also introduce an additional hyper-parameter to tune during the search: a stopping criterion n_{iter} of the quadratic programming solver of SVM. This parameter is set in [21] to $1000N_{training}$, where $N_{training}$ is the number of training points. The online tuning of this parameter may potentially improve the quality of surrogate model learning and, in the same time, reduce the CPU complexity of the learning. A completed list of surrogate model hyper-parameters is summarized in Table 1, where offline tuned values and their ranges of variation for online tuning are given. A detailed description of these parameters can be found in [21].

[1] `https://sites.google.com/site/acmesgecco/`
[2] `https://sites.google.com/site/newacmes/`

Parameter	Range for online tuning	Offline tuned value
$N_{training}$	$\lceil 4n, 2(40 + \lfloor 4n^{1.7} \rfloor) \rceil$	$40 + \lfloor 4n^{1.7} \rfloor$
C_{base}	$[0, 10]$	6
C_{pow}	$[0, 6]$	3
c_{sigma}	$[0.5, 2]$	1
n_{iter}	$[100N_{training}, 1500N_{training}]$	$1000\ N_{training}$

Table 1: Surrogate hyper-parameters, default value and range of variation (extended from [21]).

4.2 Results on Ellipsoid and Rosenbrock 20-D

First experiments were conducted on 20-dimensional Ellipsoid and Rosenbrock functions in order to investigate how the performance changes by changing k_λ and k_μ. We found that the greater the k_λ, the better the performance is observed. Moreover, the results with $k_\mu = 1$ are usually better than with $k_\mu = k_\lambda$, and this may be viewed as a more stronger pre-selection of individuals. This also can be interpreted as a trust-region method-based search, where the region is defined by the current covariance matrix and the step-size. In this context, $\hat{n}$ corresponds to the number of trust-region searches before the model is updated using new evaluated points.

Figure 2 shows the results for $k_\lambda = 1$ (the original IPOP-^{s*}aACM-ES), $k_\lambda = 10, 100, 10000$ (three modified versions with $k_\mu = 1$) as well as for the original and active versions of IPOP-CMA-ES. As can be clearly seen, the original IPOP-^{s*}aACM-ES can be improved by a factor of about 2 on Ellipsoid function and by a factor of about 1.2 on Rosenbrock function for $k_\lambda \geq 10$. Thus, the new mechanism of surrogate model exploitation allows to be about 6 times faster than IPOP-aCMA-ES and almost by one order of magnitude faster than IPOP-CMA-ES. The algorithm demonstrates the speed-up of a factor of about 3.5 on Rosenbrock function, and, thus, performs as well as BFGS algorithm and by a factor between 1.1 and 1.4 slower than NEWUOA algorithm [8]. The performance of the latter algorithms, however, may degrade significantly for certain rank-preserving transformations of f (e.g., scaling of f).

It should be noted that the computational complexity is growing linearly with k_λ, and the evaluation of candidate solutions with $\hat{f}$ becomes relatively expensive (w.r.t. the cost of surrogate model learning) when k_λ in 20-D is ca. 1000.

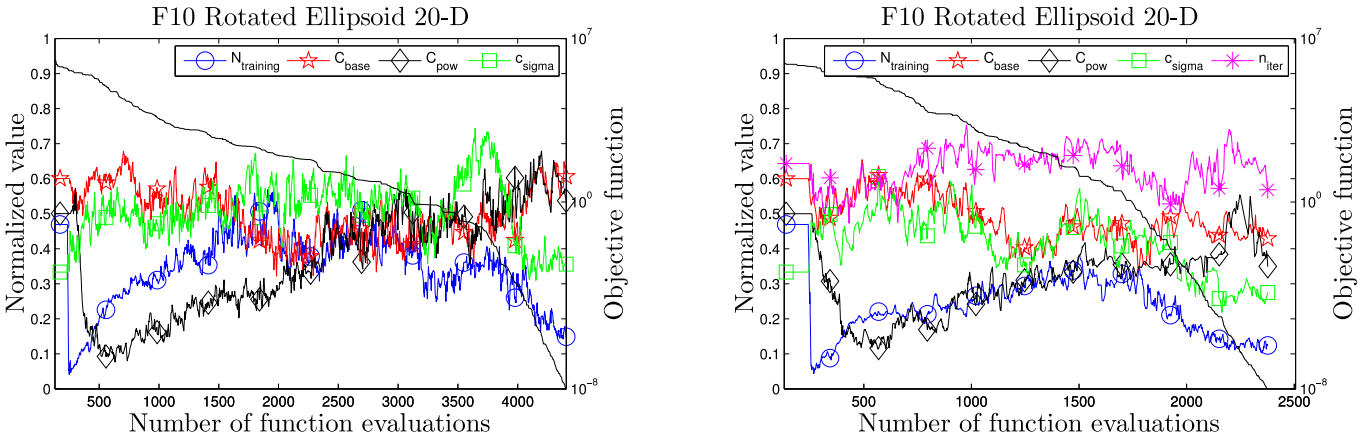

Figure 3: The median trajectories of normalized surrogate hyper-parameters estimated on 15 runs of the original (Left) and modified (Right) versions of IPOP-^{s*}aACM-ES on 20-dimensional Rotated Ellipsoid.

The CPU time per function evaluation on F10 Rotated Ellipsoid 20-D (on 2.4 GHz processor and MatLab 2006) is 1.22, 0.313, 0.311, 0.367 and 1.52 sec. for $k_\lambda = 1, 10, 100, 1000$ and 10000, respectively (note that with fixed hyper-parameters the cost would be $\lambda_{hyp} = 20$ times smaller). Thus, $k_\lambda = 10$ or $k_\lambda = 100$ can be viewed as a reasonable trade-off for these problems given that the performance is (and if) growing only marginally for $k_\lambda > 10$ (see Figure 2).

Figure 3 illustrates the dynamic of hyper-parameters adaptation on 20-dimensional Ellipsoid function for the original and modified (when $k_\lambda = 100$) versions of IPOP-^{s*}aACM-ES. As was expected, the number of training points also grows in the beginning, then becomes stable and finally decreases when the covariance matrix well approximates the optimal covariance matrix and the optimization of f becomes similar to the optimization of Sphere function. However, a smaller number of training points (and therefore CPU time) is usually needed for the new algorithm, that might be explained by the fact that the algorithm optimizes f faster and, therefore, outdated points are probably of a lesser importance.

Figure 3 also shows the adaptation of the number of iterations n_{iter} used to solve the quadratic programming problem of SVM learning. The results partially confirm our expectations that the offline tuned value of n_{iter} (see Table 1) was reasonably well chosen in [21]. We also do not observe any moderate difference due to this adaptation on other BBOB benchmarks problems. This conjecture, however, should be more carefully verified by using a larger search range for n_{init}, and should not prevent from studying alternative strategies for SVM problem learning and corresponding stopping criteria.

The adaptation of other hyper-parameters is more difficult to interpret, however, in both cases, C_{pow} steadily increases as it was pointed out already in [21].

4.3 Results on Noiseless BBOB Functions

The proposed modification of ^{s*}ACM-ES was implemented in BIPOP scenario of restarts and compared with BIPOP-CMA-ES [5], BIPOP-aCMA-ES [20] and BIPOP-^{s*}aACM-ES [21] on 20-dimensional noiseless BBOB problems. In all experiments, the maximum number of function evaluations was set to $10^6 n$. The number of function evaluations when surrogate models are used was set to $10^4 n$ to fit to the BBOB-2013 context of expensive optimization in contrast to $10^6 n$ used in BIPOP-^{s*}aACM-ES [21].

We found that the use of $k_\lambda = 10$ for 10-D, $k_\lambda = 100$ for 20-D and $k_\lambda = 1000$ for 40-D (see [22], here only 20-D case is shown) represents a reasonable trade-off between the speed-up and the computational complexity (a simple formula to adjust k_λ for other dimensions can be used). For problem dimensions smaller than 10, the use of large k_λ may lead to a certain divergence even on unimodal functions. This can be interpreted as follows: when k_λ is large, some of points sampled in 2-dimensional space might lie "far" from the current mean of the mutation distribution and "far" from the training set used to build the model. Thus, the model may incorrectly suggests these points, that may lead to a certain divergence. The use of some truncated sampling of points might resolve this issue, but this would lead to a change of mechanism of CMA-ES which expects that all points are sampled from multivariate normal distribution.

In order to avoid a possible divergence on multimodal functions, $\hat{n}_{k_\lambda}$ usually should be larger than 1. By some offline tuning we found that the use of $\hat{n}_{k_\lambda} = 3$ and $\hat{n}_{k_\lambda} = 4$ in most of the cases prevents the divergence (at least on tested BBOB problems), and we use $\hat{n}_{k_\lambda} = 4$ in our experiments on BBOB. A simple alternative to this would be to decrease $\widehat{n_{max}}$ or τ_{err}.

Figure 4 shows the aggregated results for all 24 noiseless BBOB problems and different subgroups of these problems. The proposed version of BIPOP-^{s*}aACM-ES is depicted as BIPOP-^{s*}aACM-ES-k. The best improvements of results is observed on moderate and ill-conditioned functions, where the speed-up up to a factor of 2 is observed. On ill-conditioned problems, BIPOP-^{s*}aACM-ES-k is about 3-6 times faster than BIPOP-aCMA-ES and by almost one order of magnitude faster than BIPOP-CMA-ES. The results of BIPOP-^{s*}aACM-ES and BIPOP-^{s*}aACM-ES-k are similar on multimodal functions, while BIPOP-^{s*}aACM-ES slightly outperforms the latter for the budgets of function evaluations larger than $10^4 n$, since the surrogate is still in use. The overall ranking of all algorithms is comparable and a difference is observed most likely due to the stochasticity of the process (at least for BIPOP-aCMA-ES and BIPOP-^{s*}

aACM-ES-k, which represent the same algorithm after $10^4 n$ function evaluations).

5. CONCLUSION AND PERSPECTIVES

We presented a new mechanism for surrogate model exploitation, where a larger population size is used to optimize the surrogate, while the default population size is used to optimize the expensive function. We found that while a more intensive exploitation of the surrogate is beneficial on unimodal functions, this also might lead to a certain divergence on multimodal functions. To avoid this potential issue, we suggested to more intensively exploit the surrogate only if it is sufficiently accurate. The proposed mechanism implemented in ^{s*}ACM-ES algorithm and its BIPOP extension demonstrates the speed-up of a factor up to 2 (compared with the original version of ^{s*}ACM-ES) on ill-conditioned BBOB problems. The resulting algorithm also preserves the properties of invariance of CMA-ES and might become a baseline version of ^{s*}ACM-ES in some near future.

The main perspective for further research is to replace the newly introduced offline-tuned parameters by some online procedure, which will be able to optimally control surrogate model exploitation. Such procedure might be based on some a posteriori analysis of what strategy was optimal in previous generations by analyzing (running experiments on) previous surrogate models and alternative exploitation strategies. Another perspective is to reduce the time complexity of the algorithm which limits its application to $n \lesssim 50 - 100$ and up to ca. 5000 training points (see, e.g., [20]).

6. REFERENCES

[1] Z. Bouzarkouna, A. Auger, and D. Y. Ding. Investigating the Local-Meta-Model CMA-ES for Large Population Sizes. In *EvoApplications (1)*, pages 402–411, 2010.

[2] D. Buche, N. N. Schraudolph, and P. Koumoutsakos. Accelerating evolutionary algorithms with Gaussian process fitness function models. *Systems, Man and Cybernetics, Part C, IEEE Transactions on*, 35(2):183–194, 2005.

[3] M. Emmerich, A. Giotis, M. Özdemir, T. Bäck, and K. Giannakoglou. Metamodel-Assisted Evolution Strategies. In *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, PPSN VII, pages 361–370, London, UK, UK, 2002. Springer-Verlag.

[4] 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*, 15:617–644, 2009.

[5] N. Hansen. Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In *GECCO Companion*, pages 2389–2396, 2009.

[6] N. Hansen. References to CMA-ES Applications. Website, January 2013. Available online at http://www.lri.fr/ hansen/cmaapplications.pdf.

[7] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-Parameter Black-Box Optimization Benchmarking 2012: Experimental Setup. Technical report, INRIA, 2012.

[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 *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, GECCO 2010, pages 1689–1696, New York, NY, USA, 2010. ACM.

[9] N. Hansen, S. Müller, 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.

[10] N. Hansen and A. Ostermeier. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evol. Comput.*, 9(2):159–195, June 2001.

[11] N. Hansen and R. Ros. Benchmarking a weighted negative covariance matrix update on the BBOB-2010 noiseless testbed. In *GECCO '10: Proceedings of the 12th annual conference comp on Genetic and evolutionary computation*, pages 1673–1680, New York, NY, USA, 2010. ACM.

[12] F. Hoffmann and S. Holemann. Controlled Model Assisted Evolution Strategy with Adaptive Preselection. In *International Symposium on Evolving Fuzzy Systems*, pages 182–187. IEEE, 2006.

[13] G. A. Jastrebski and D. V. Arnold. Improving Evolution Strategies through Active Covariance Matrix Adaptation. In *IEEE Congress on Evolutionary Computation*, pages 2814–2821, 2006.

[14] Y. Jin. Quality measures for approximate models in evolutionary computation. In *GECCO 2003: Proceedings of the Bird of a Feather Workshop, Genetic and Evolutionary Computation Conference*, pages 170–173. AAAI, 2003.

[15] Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, pages 61–70, 2011.

[16] Y. Jin, M. Olhofer, and B. Sendhoff. Managing Approximate Models in Evolutionary Aerodynamic Design Optimization. In *IEEE Congress on Evolutionary Computation*, pages 592–599. IEEE Press, 2001.

[17] S. Kern, N. Hansen, and P. Koumoutsakos. Local Meta-Models for Optimization Using Evolution Strategies. In *PPSN IX*, pages 939–948. LNCS 4193, Springer, 2006.

[18] S. Kern, N. Hansen, and P. Koumoutsakos. Fast Quadratic Local Meta-Models for Evolutionary Optimization of Anguilliform Swimmers. In Neittaanmaki et al., editors, *EUROGEN 2007*, Helsinki, Finlande, 2007.

[19] I. Loshchilov, M. Schoenauer, and M. Sebag. Comparison-Based Optimizers Need Comparison-Based Surrogates. In R. S. et al., editor, *Parallel Problem Solving from Nature (PPSN XI)*, volume 6238 of *LNCS*, pages 364–373. Springer, 2010.

[20] I. Loshchilov, M. Schoenauer, and M. Sebag. Black-box Optimization Benchmarking of IPOP-saACM-ES and BIPOP-saACM-ES on the BBOB-2012 Noiseless Testbed. In *Genetic and Evolutionary Computation Conference (GECCO Companion)*, pages 175–182. ACM Press, July 2012.

[21] I. Loshchilov, M. Schoenauer, and M. Sebag. Self-Adaptive Surrogate-Assisted Covariance Matrix Adaptation Evolution Strategy. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 321–328. ACM Press, July 2012.

[22] I. Loshchilov, M. Schoenauer, and M. Sebag. BI-population CMA-ES Algorithms with Surrogate Models and Line Searches. In T. Soule and J. H. Moore, editors, *Genetic and Evolutionary Computation Conference (GECCO Companion)*, page Christian Blum and Enrique Alba. ACM Press, July 2013.

[23] M. Papadrakakis, N. D. Lagaros, and Y. Tsompanakis. Structural optimization using evolution strategies and neural networks. *Comput. Methods Appl. Mech. Eng.*, 156(1-4):309–333, 1998.

[24] J. Poland. Explicit Local Models: Towards "Optimal" Optimization Algorithms. In *ECML*, pages 569–571, 2004.

[25] J. Poland and A. Zell. Main Vector Adaptation: A CMA Variant with Linear Time and Space Complexity. In L. Spector, editor, *Genetic and Evolutionary Computation Conference (GECCO 2001)*, pages 1050–1055. Morgan Kaufmann, 2001.

[26] T. P. Runarsson. Constrained Evolutionary Optimization by Approximate Ranking and Surrogate Models. In *PPSN*, pages 401–410, 2004.

[27] T. P. Runarsson. Ordinal Regression in Evolutionary Computation. In Th. Runarsson et al., editor, *PPSN IX*, pages 1048–1057. LNCS 4193, Springer Verlag, 2006.

[28] D. F. Shanno. Conditioning of Quasi-Newton Methods for Function Minimization. *Mathematics of Computation*, 24(111):647–656, 1970.

[29] M. W. Trosset and V. Torczon. Numerical optimization using computer experiments. Technical report, DTIC, 1997.

[30] H. Ulmer, F. Streichert, and A. Zell. Evolution Strategies assisted by Gaussian Processes with Improved Pre-Selection Criterion. In *IEEE Congress on Evolutionary Computation*, pages 692–699, 2003.

[31] H. Ulmer, F. Streichert, and A. Zell. Model-assisted steady-state evolution strategies. In *Proceedings of the 2003 international conference on Genetic and evolutionary computation*, GECCO 2003, pages 610–621. Springer-Verlag, 2003.

[32] H. Ulmer, F. Streichert, and A. Zell. Evolution strategies with controlled model assistance. In *IEEE Congress on Evolutionary Computation*, pages 1569 – 1576, 2004.

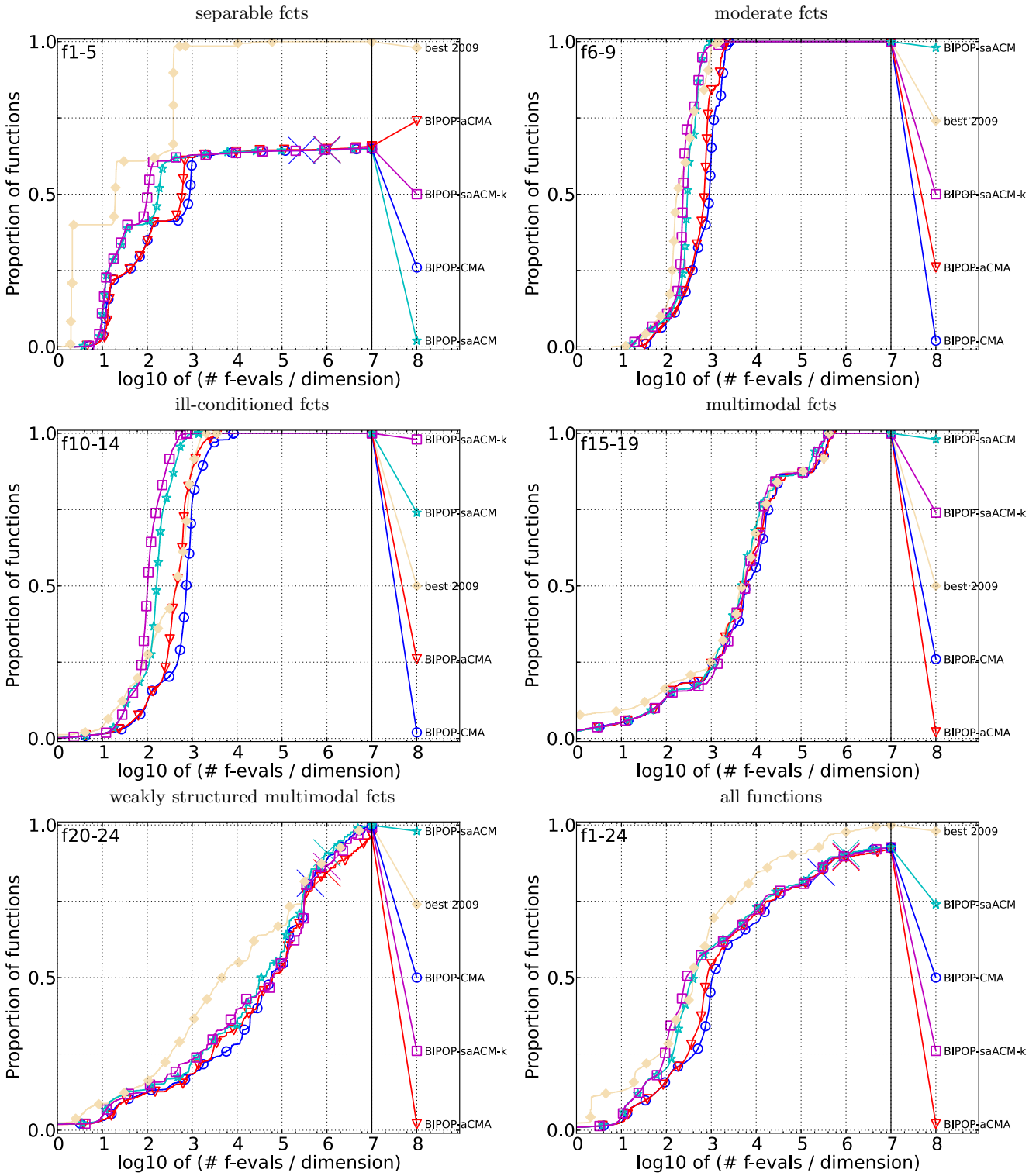

Figure 4: Bootstrapped empirical cumulative distribution of the number of objective function evaluations divided by dimension for 50 targets in $10^{[-8..2]}$ for all functions and subgroups in 20-D. The "best 2009" line corresponds to the best ERT observed during BBOB 2009 for *each single target*. The proposed algorithm is depicted as BIPOP-saACM-k. A detailed description how to read the figure is given in [8].