ITAT 2016 Proceedings, CEUR Workshop Proceedings Vol. 1649, pp. 138–146
http://ceur-ws.org/Vol-1649, Series ISSN 1613-0073, © 2016 N. Orekhov, L. Bajer, M. Holeňa

Testing Gaussian Process Surrogates on CEC'2013 multi-modal benchmark

Nikita Orekhov[1], Lukáš Bajer[2], and Martin Holeňa[3]

[1] Faculty of Information Technology, Czech Technical University, Prague, orekhnik@fit.cvut.cz
[2] Faculty of Mathematics and Physics, Charles University, Prague, bajeluk@matfyz.cz
[3] Czech Academy of Sciences, Prague, martin@cs.cas.cz

Abstract: This paper compares several Gaussian-process-based surrogate modeling methods applied to black-box optimization by means of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which is considered state-of-the-art in the area of continuous black-box optimization. Among the compared methods are the Model-assisted CMA-ES, the Robust Kriging Metamodel CMA-ES, and the Surrogate CMA-ES. In addition, a very successful surrogate-assisted self-adaptive CMA-ES, which is not based on Gaussian processes, but on ordinary regression by means of support vector machines has been included into the comparison. Those methods have been benchmarked using CEC'2013 testing functions. We show that the surrogate CMA-ES achieves best results at the beginning and later phases of optimization process, conceding in the middle to surrogate-assisted CMA-ES.

1 Introduction

Evolutionary computation has been successfully applied to a wide spectrum of engineering problems. The randomized exploration guided by a set of candidate solutions (population) may be resistant against most optimization obstacles such as noise or multi-modality. It makes evolutionary algorithms (EAs) suitable for black-box optimization where an analytic form of the objective function is not known. In such cases, values of the objective function evaluations are the only available information to optimize the function.

The evaluation of the objective function can often be costly, i.e., it takes a significant amount of time and/or money. Typically, a black-box objective function is evaluated empirically during some experiment such as gas turbine profiles optimization [3] or protein folding stability optimization [4].

Today, a state-of-the-art among evolutionary black-box optimizers is the Covariance Matrix Adaptation evolution strategy (CMA-ES), which was initially introduced in [7]. However, as any other evolutionary optimization methods, CMA-ES may suffer from insufficient convergence speed w.r.t. the budget of expensive function evaluations. This can be avoided by the introduction of so-called surrogate models (aka metamodels), which provide regression-based predictions of the expensive objective function values.

Past works on surrogate-assisted optimization showed that models based on Gaussian Processes (GP) can lead to a significant reduction as to the number of evaluations of the original objective function [2]. The aim of this paper is to compare several GP-based surrogate-modeling techniques against another successful surrogate modeling method used in connection with CMA-ES. In particular, we examine the POI-based approach [13], the robust kriging metamodel [9] and the surrogate CMA-ES (S-CMA-ES) [2] and compare them against the support vector machines-based method called ^{s*}ACM-ES [11] and the basic CMA-ES without a surrogate model. The comparison has been performed on the multi-modal benchmark from CEC'2013.

The remainder of the paper is organized as follows. Section 2 introduces the necessary background of the Covariance Matrix Adaptation ES and Gaussian processes in the context of black-box optimization. Section 3 briefly recalls tested surrogate models, while section 4 describes the performed experiments and comments on obtained results.

2 Background

2.1 Surrogate modeling in black-box optimization

In the context of black-box optimization by CMA-ES, a surrogate model is built using some points sampled by the CMA-ES and their objective values. Then, the model is used to predict the quality of the sampled points in order to reduce number of function evaluations. However, the points should be carefully selected. Generally, there are two main ways to select them. The first approach proposed in [3] means that in each iteration of the overall optimization algorithm, one replaces the original objective function by its surrogate model, *estimates the model's global optimum* and evaluates it with the original objective function.

The other way consists in selecting a *controlled fraction* of individuals that should be evaluated on the original objective function. Such approach is thus called *evolution control* (EC) [8]. More specifically, costly evaluating the original objective for the entire population only at certain generations is called *generation-based* EC, while evaluating it only for a part of the population at each generation is called *individual-based* EC.

Using a surrogate model to find the global optimum, one *exploits* model's knowledge regarding the unknown objective function. However, this can potentially prevent the optimization algorithm from convergence to an optimal solution due to unexplored regions that were incorrectly modeled. To protect the model from making such predictions there is a need for some mechanism which cares about the *exploration* of new regions in the solution space. The *merit functions* were introduced precisely for this reason: they incorporate both exploration and exploitation patterns into the decisions performed during the surrogate modeling.

The variance σ^2 of the model predictions may be used as a merit function. Thus, the larger the variance, the higher the uncertainty of the predicted objective function value. The merit function is expressed as:

$$f_{\mathrm{var}}(\boldsymbol{x}) = \sigma^2(\boldsymbol{x}),$$

where $\sigma^2(\boldsymbol{x})$ is the variance of the model's prediction at $\boldsymbol{x}$.

Lower confidence bound (LCB) is another merit function, which has the following form:

$$f_{\alpha}(\boldsymbol{x}) = \hat{f}(\boldsymbol{x}) - \alpha\sigma^2(\boldsymbol{x}),$$

where $\alpha \geq 0$ balances between the exploration and exploitation and $\hat{f}(\boldsymbol{x})$ is the model's prediction of the objective function value at $\boldsymbol{x}$.

The next merit function is a probability of improvement (POI). For any given solution vector $\boldsymbol{x}$, the model predictions about the function value at $\boldsymbol{x}$ is a realization of the random variable $Y(\boldsymbol{x})$. Having $f_{\min}$ as the current best obtained value of the original objective function, we define for any $T \leq f_{\min}$ the probability of improvement as:

$$f_{\mathrm{T}}(\boldsymbol{x}) = p(Y(\boldsymbol{x}) \leq T). \tag{1}$$

2.2 Gaussian processes

Gaussian process is a random process indexed by $\mathbb{R}^n$ such that its marginal indexed by a finite set of points $\boldsymbol{X}_N = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^n$ has an N-dimensional Gaussian distribution. Evaluating the objective function at those points results in a vector of function values $\boldsymbol{t}_N \in \mathbb{R}^N$. If that vector is viewed as a particular realization of the respective Gaussian distribution, the GP provides a prediction of the distribution of the function value t_{N+1} at some point $\boldsymbol{x}_{N+1}$.

Gaussian process models are determined by their mean and covariance matrix. This matrix is constructed by applying a covariance function $k : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ to each pair of points in $\boldsymbol{X}_N$. The covariance functions have a certain set of parameters, usually called hyper-parameters due to the fact that the covariance function itself is a parameter of the Gaussian process. Typically, the hyper-parameters are obtained through the maximum likelihood estimation.

We recall the covariance functions employed in the observed methods. Note that the hyper-parameters below are marked as θ. The first one is *exponential* covariance function. It is used in [9] and has the following form:

$$k_{\mathrm{E}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \exp\Big(-\frac{r}{\theta}\Big),$$

where $r = \|\boldsymbol{x}_p - \boldsymbol{x}_q\|$ is the distance between points.

The next covariance function is called *squared exponential*. This is used in [13] and its basic variant can be expressed as:

$$k_{\mathrm{SE}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \exp\Big(-\frac{r^2}{\theta}\Big).$$

In [2], the isotropic *Matérn* covariance function $k^{5/2}_{\text{Matérn}}$ is used in its form:

$$k^{5/2}_{\text{Matérn}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \theta_1\left(1 + \frac{\sqrt{5}r}{\theta_2} + \frac{5r^2}{3\theta_2^2}\right)\exp\left(-\frac{\sqrt{5}r}{\theta_2}\right).$$

Having defined the covariance matrix for N points, an extension of $\boldsymbol{C}$ after including an $(N+1)$th point reads:

$$\boldsymbol{C}_{N+1} = \begin{pmatrix} \boldsymbol{C}_N & \boldsymbol{k} \\ \boldsymbol{k}^{\mathrm{T}} & \kappa \end{pmatrix},$$

where $\boldsymbol{k} = k(\boldsymbol{x}_{N+1}, \boldsymbol{X}_N)$ is an N-dimensional real vector of covariances between the new point and points from training set, while $\kappa = k(\boldsymbol{x}_{N+1}, \boldsymbol{x}_{N+1})$ is a variance of the new point [3]. Consequently,

$$t_{N+1} \sim \mathcal{N}(\mu_{N+1}, \sigma^2_{N+1}), \tag{2}$$

where $\mu_{N+1} = \boldsymbol{k}^{\mathrm{T}}\boldsymbol{C}_N^{-1}\boldsymbol{t}_N$ is the predictive mean and $\sigma^2_{N+1} = \kappa - \boldsymbol{k}^{\mathrm{T}}\boldsymbol{C}_N^{-1}\boldsymbol{k}$ is the predictive variance.

In the context of Gaussian processes, we have $f_{\min} = \min(t_1, \ldots, t_N)$. Then, considering distribution from (2), the POI criterion can be rewritten from (1) as follows:

$$f_{\mathrm{T}}(\boldsymbol{x}) = \Phi\left(\frac{T - \hat{t}(\boldsymbol{x})}{\sigma(\boldsymbol{x})}\right),$$

where Φ is the cumulative distribution function of the normal distribution $\mathcal{N}(0,1)$.

Areas with high POI value have a high probability to sample point with objective value better than $f_{\min}$. Regions with model prediction $\hat{t}(\boldsymbol{x}) \gg f_{\min}$ will have POI value close to zero, which encourages the model to search somewhere else. The POI value becomes large when the variance (i.e. σ^2) is large, which is typical for unexplored areas. Therefore, the use of POI may be useful for dealing with multi-modal functions.

2.3 CMA-ES

The CMA-ES employs the concept of the adaptation of internal variables from the data. Particularly, it adapts several components such as mutation step size, distribution mean and the covariance matrix, which represents a local approximation of the function landscape.

In the CMA-ES, a generation of candidate solutions is usually obtained by sampling a *normally-distributed* random variable:

$$\boldsymbol{x}_k^{(g+1)} \sim \boldsymbol{m}^{(g)} + \sigma^{(g)}\mathcal{N}\Big(\boldsymbol{0}, \boldsymbol{C}^{(g)}\Big) \text{ for } k = 1, \ldots, \lambda,$$

where $\lambda \geq 2$ is a population size, $\boldsymbol{x}_k^{(g+1)} \in \mathbb{R}^n$ is the k-th offspring in $g+1$-th generation, $\boldsymbol{m}^{(g)} \in \mathbb{R}^n$ is the mean of the sampling distribution, $\sigma^{(g)} \in \mathbb{R}_+$ is a step-size, $\boldsymbol{C}^{(g)} \in \mathbb{R}_{n\times n}$ is a covariance matrix. Below we list basic equations for the adaptation of the mean, step-size and covariance matrix used in CMA-ES. For details we refer to [5].

The mean of the sampling distribution is a weighted average of μ best-ranked points from $\boldsymbol{x}_1^{(g+1)}, \dots, \boldsymbol{x}_\lambda^{(g+1)}$:

$$\boldsymbol{m}^{(g+1)} = \sum_{i=1}^{\mu} w_i\, \boldsymbol{x}_{i:\lambda}^{(g+1)},$$

where $\sum_{i=1}^{\mu} w_i = 1,\ w_1 \geq w_2 \geq \dots \geq w_\mu > 0$ are weight coefficients and $i:\lambda$ notation means i-th best individual out of $\boldsymbol{x}_1^{(g+1)}, \dots, \boldsymbol{x}_\lambda^{(g+1)}$.

The covariance matrix $\boldsymbol{C}$, being initially set to identity matrix, is updated by *rank-one-update* and *rank-μ-update* during each iteration. A *cumulation* strategy [5] is applied to combine consecutive steps in the search space.

$$\boldsymbol{C}^{(g+1)} = (1 - c_1 - c_\mu)\, \boldsymbol{C}^{(g)} + c_1 \underbrace{\boldsymbol{p}_c^{(g+1)} \boldsymbol{p}_c^{(g+1)\top}}_{\text{rank-one update}} + \underbrace{c_\mu \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda}^{(g+1)} \boldsymbol{y}_{i:\lambda}^{(g+1)\top}}_{\text{rank-}\mu\text{ update}},$$

where c_1 and c_μ are learning rates for rank-one-update and rank-μ-update, $\boldsymbol{p}_c^{(g+1)}$ is an evolution path and $\boldsymbol{y}_{i:\lambda}^{(g+1)} = \left(\boldsymbol{x}_{i:\lambda}^{(g+1)} - \boldsymbol{m}^{(g)}\right)/\sigma^{(g)}$.

The step length update reads:

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|}{\mathbb{E}\|\mathscr{N}(\mathbf{0},\mathbf{I})\|} - 1 \right) \right),$$

where c_σ is a learning rate and $\boldsymbol{p}_\sigma^{(g+1)}$ is an evolution path.

The CMA-ES is considered as a *local* optimizer. So in case of multi-modal functions, it tends to end up in a local optima. Hence, to reduce the influence of such behavior, several restart strategies were introduced. The perhaps best known one is the IPOP-CMA-ES [1]. In this modification, an optimization process is being interrupted several times and independently restarted with the population size increased by a certain factor (typically 2). IPOP version is set default for all of the CMA-ES algorithms throughout the article.

3 Tested methods

3.1 POI MAES

Ulmer et al. refer to this algorithm as Probability of Improvement Model-assisted Evolution Strategy (POI MAES) [13]. The method uses a modification of the covariance function SE, which has $n+2$ hyper-parameters, obtained by likelihood maximization:

$$k(\boldsymbol{x}_p, \boldsymbol{x}_q, \boldsymbol{\theta}) = \theta_1 \exp\left(-\frac{1}{2} \sum_{i=1}^{n} \frac{(x_{p_i} - x_{q_i})^2}{r_i^2} \right) + \delta_{pq}\theta_2,$$

where r_i^2 are length-scales for the individual dimensions and $\delta_{pq} = \begin{cases} 0, & \text{if } p \neq q \\ 1, & \text{if } p = q \end{cases}$ is the Kronecker delta.

The model is incorporated into the CMA-ES via individual-based EC, which the authors call *pre-selection*. In every iteration, $\lambda_{\text{pre}} > \lambda$ new individuals are sampled from μ parents and evaluated on the surrogate model using the POI merit function. Then, the λ offsprings with the highest POI are selected and evaluated on the objective function. In the end, the surrogate model is updated.

3.2 Robust CMA-ES

The approach below was designed to provide robustness approximations of the objective function values. It is also combined with a special method to select promising solutions [9].

One can imagine robustness approximation as an attempt to predict function values within noisy or imprecise environment. Considering expensive optimization, it becomes extremely important to make precise predictions for noisy problems as it requires a certain trade-off between limiting the noise and reducing the number of evaluations.

The robustness approximation is achieved in a way that the local model is trained around every point $\boldsymbol{x}_k$ to be estimated. To achieve this, the algorithm chooses n_{krig} pairs $(\boldsymbol{x}, t)$ in a specific way described below from a given archive $\mathscr{A}$ and stores them in a local training set $\mathscr{D}$. If the amount of points does not suffice, the algorithm returns also a set $\boldsymbol{X}_{\text{cand}} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_l\}$ of length $l = n_{\text{krig}} - |\mathscr{D}|$. Points from $\boldsymbol{X}_{\text{cand}}$ are then evaluated with the original objective function and added to both $\mathscr{A}$ and $\mathscr{D}$.

The procedure to select the pairs $(\boldsymbol{x}, t)$ from the archive $\mathscr{A}$ is as follows. First, the n_{krig} points are generated via the Latin hypercube sampling method and are stored in a reference set $\mathscr{R}$. Then, for every point $\boldsymbol{x} \in \mathscr{R}$ (subsequently called reference point), the closest $\boldsymbol{x}$ from $\mathscr{A}$ is assigned. An important note here is that the reference point from $\mathscr{R}$ must be closest to the archive point $\boldsymbol{x}$ as well. If this is the case, the archive point with its corresponding objective function value are added to the training set. Otherwise, the reference point is considered a suitable candidate for sampling and added to $\boldsymbol{X}_{\text{cand}}$. When all points from $\mathscr{R}$ are assigned, the reference point from the assigned pair (archive point, reference point), for which the distance between both points is the largest among all assigned pairs, is selected and evaluated.

Using the procedure above a separate training set is selected for every point $\boldsymbol{x}^{(g)}$ to be estimated at generation g and the model is trained. Then, the fitness value at $\boldsymbol{x}^{(g)}$ is estimated according to so-called *multi evaluation method*

(MEM). In this method the approximation is obtained as a mean value of several approximations at points with random perturbation in the input space, i.e.:

$$\hat{f}_{MEM}(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m}\hat{f}(\boldsymbol{x}+\boldsymbol{\delta}_i),$$

where m is the predefined amount of points to perform approximation and $\boldsymbol{\delta}_i$ denotes a random variation.

3.3 S-CMA-ES

Another surrogate-assisted approach is called *Surrogate CMA-ES* (S-CMA-ES). It was initially proposed in [2] and later extended to the *Doubly Trained* S-CMA-ES (DTS-CMA-ES) in [12]. The S-CMA-ES employs the Matérn covariance function mentioned in 2.2 with hyper-parameters $\boldsymbol{\theta} = \{\sigma_f^2, l, \sigma_n^2\}$ optimized by the maximum-likelihood approach.

The S-CMA-ES algorithm uses generation-based evolution control which means that an *original*-evaluated generation and *model*-evaluated generations interleave. During the *original* generations, all the points are evaluated by the expensive objective function. In every *model* generation, at least n_{MIN} archive points have to be selected to train the surrogate model. The selection process is restricted to points that do not have the Mahalanobis distance from the CMA-ES' mean value $\boldsymbol{m}$ larger than some prescribed limit r:

$$\mathscr{D} \leftarrow \{(\boldsymbol{x},t) \in \mathscr{A} | \sqrt{(\boldsymbol{m}-\boldsymbol{x})^{\top}(\sigma^2\boldsymbol{C})^{-1}(\boldsymbol{m}-\boldsymbol{x})} \leq r\},$$

where $\boldsymbol{C}$ is the CMA-ES covariance matrix and σ is the step-size at the considered generation.

If there are not enough points, building the model is postponed and all fitness evaluations are performed on the original objective function. When there are enough points, at most n_{MAX} points are selected via k-NN clustering to train a GP model.

The extension DTS-CMA-ES selects $n_{\text{orig}} < \lambda$ most uncertain points w.r.t. the employed merit function and evaluates them on the original objective function. Then, the surrogate model is being retrained on the archive extended with the newly evaluated n_{orig} points. Finally, the $\lambda - n_{\text{orig}}$ remaining points function values are predicted by the model and returned to the CMA-ES.

3.4 s*ACM-ES

This subsection describes a method called Self-adaptive Surrogate-assisted CMA-ES (^{s*}ACM-ES) [11], which is not GP-based, but is used for comparison.

This method employs the ordinal regression based on Ranking support vector machines. Evaluations made by such model provide only rankings of the points in order to achieve the invariance to the rank-preserving transformations of f. The method adapts the model hyper-parameters in an on-line manner. In addition, it also depends on much larger population size while operating on surrogate and preserving original population size for the original objective function evaluations.

A generation-based EC is used, the number $\hat{n}$ of model-evaluated generations is adjusted according to the model error, assessed in the interleaving generation in which the original objective function is evaluated.

The algorithm adjusts $\hat{n}$ by a linear function inversely proportional to a *global* model error $\text{Err}(\boldsymbol{\theta})$. The global error is updated with some relaxation from a *local* error on λ most recent evaluated points. Denoting Λ to be the set of those λ points, the local error is estimated as follows:

$$\text{Err}(\boldsymbol{\theta}) = \frac{2}{|\Lambda|(|\Lambda|-1)}\sum_{i=1}^{|\Lambda|}\sum_{j=i+1}^{|\Lambda|} w_{i,j}\cdot 1_{\hat{f}_{\boldsymbol{\theta},i,j}},$$

where $1_{\hat{f}_{\boldsymbol{\theta},i,j}}$ is true if $\hat{f}$ violates the ordering of pair (i,j) given by the real objective function f and $w_{i,j}$ defines the weights of such violations.

The procedure of the surrogate error optimization is done in the end of every ES generation, where the model hyper-parameters are optimized by one iteration of an additional CMA-ES (referred to as CMA-ES #2 in [11]). Here, the algorithm samples λ_{hyp} different points in a space of hyper-parameters and builds λ_{hyp} surrogate models. Then, those models are evaluated using the $\text{Err}(\boldsymbol{\theta})$ metric and $\mu_{\text{hyp}} = \lfloor\lambda_{\text{hyp}}/2\rfloor$ best performing are used to update internal variables of the CMA-ES #2. The resulting mean of the hyper-parameter distribution is used to obtain $\boldsymbol{\theta}$ for the next generation of ^{s*}ACM-ES.

Finally, the standard CMA-ES configurations differ if it optimizes the surrogate model $\hat{f}$. In such cases, it uses larger population sizes $\lambda = k_\lambda\lambda_{\text{default}}$ and $\mu = k_\mu\mu_{\text{default}}$, where $k_\lambda, k_\mu \geq 1$. In order to prevent degradation when the model is inaccurate, k_λ is also adjusted w.r.t. the $\text{Err}(\boldsymbol{\theta})$.

4 Experiments

4.1 Benchmark functions

The experiments has been conducted on a recently proposed benchmark, introduced on the special session for multi-modal function optimization during the Congress of Evolutionary Computation (CEC) in 2013 [10]. The benchmark contains 12 noiseless multi-modal functions being defined for dimensions 1 – 20D. Note that this test set differs from the one used in [9], where the noise was introduced by oscillations in the input space. In addition, the experiments has been conducted within the BBOB framework [6] via introducing 9 new benchmark functions. Since the CMA-ES is not able to run in 1D, we excluded 1D functions from our experiments. Altogether, the following set of functions was employed (dimensionality is shown in brackets): Himmelblau (2D), Six-Hump Camel Back (2D), Shubert (2D, 3D), Vincent (2D, 3D), Modified Rastrigin (2D), Composition Function

1 (2D), Composition Function 2 (2D), Composition Function 3 (2D, 3D, 5D, 10D), Composition Function 4 (3D, 5D, 10D, 20D).

4.2 Experimental setup

The experimental testing has been prepared according to the BBOB framework specifications [6]. First, for every benchmark function, the global minimum is denoted as f_{opt}. The target function value f_{target} was set to $f_{\text{opt}} + 10^{-8}$. This value is considered as a stopping criterion for tested algorithms. The performance of the tested methods is measured as a difference of the best objective value found so far t_{best} and global minimum of the function: $\Delta_f = t_{\text{best}} - f_{\text{opt}}$ for respective numbers of original fitness evaluations. The resulting Δ_f values are calculated for `Ntrials` $= 15$ algorithm runs for every benchmark function and every dimension.

The tested algorithms had the following settings:

- *CMA-ES*: the Matlab code version 3.62 of the IPOP-CMA-ES has been used, with the number of restarts $= 4$, *IncPopSize* $= 2$, $\sigma_{\text{start}} = \frac{8}{3}$, $\lambda = 4 + \lfloor 3 \log n \rfloor$ and setting the remaining parameters to its defaults [6]. Note that since the tested surrogate methods perform within the CMA-ES environment, they also had those settings.
- *POI MAES*: the only algorithm, which does not use the original author's implementation. We implemented it in Matlab according to its description in [13]. The λ_{pre} was set to 3λ and the number of points to train the model to 2λ. Here we weren't able to confirm or deny if the algorithm replicates the results from the original paper.
- *Robust CMA*: the n_{krig} is set to $2n$, the m parameter is 10.
- *S-CMA-ES*: the DTS-CMA-ES extension was used, the Mahalanobis distance limit r was set to 8, the covariance function $k^{5/2}_{\text{Matérn}}$ was used with the starting values $(\sigma_n^2, l, \sigma_f^2) = \log\,(0.01, 2, 0.5)$, n_{orig} was set to $\lceil 0.1\lambda \rceil$ and f_{var} was used as the merit function, see [12] for details.
- *^{s*}ACM-ES*: uses settings from [11].

4.3 Results and their assessment

The results of testing are now analyzed in two-stages. During the first stage, the algorithm's performance is analyzed for every benchmark function and every dimension separately. The second stage concentrates on the analysis of results aggregated over individual functions and dimensions, and finally over the entire benchmark set.

The results of testing over individual CEC functions are shown in Figure 1. The performance of every method has been calculated for `Ntrials` runs and is represented by the empirical median Δ_f^{med} (straight lines) and quartiles (translucent area) of the Δ_f.

It can be seen that the S-CMA-ES and ^{s*}ACM-ES methods have the best performance on the majority of benchmark functions. However, for the Shubert function in 2D and 3D and for the Composition Function 4 in 5D both methods seem to miss the global optimum and do not noticeably speed up the original CMA-ES. An example visualization of the experiment on 2D Shubert function is shown in Figure 2. In addition, the ^{s*}ACM-ES fails to find the global optimum for the Vincent function in 3D.

The other methods lead to a deceleration of the CMA-ES convergence speed in most cases. The global optimum remains undiscovered except for the Vincent, Himmelblau and Six-Hump Camel Back functions by POI MAES. POI MAES was the best method on the Shubert function. Also, this method outperformed the original CMA-ES on 20 dimensional Composite Function 4, which may indicate its ability to speed up the CMA-ES for higher dimensions.

The Robust CMA method achieved the worst results in our experiments; comparable performance was shown only for Shubert function. However, this can be due to the fact that the considered benchmark functions were noiseless whereas this method has been developed primarily for noisy objective functions.

Next, Figure 3 depicts results aggregated over different functions, while Figure 4 shows results aggregated over the entire benchmark set. To enable aggregation, we use scaled logarithms of medians. First of all, the evaluation budgets are normalized by the dimensionality. For benchmarking, the budgets were limited to 250 function evaluations per dimension (FE/D). Then, the scaled logarithm $\Delta_f^{\log}$ of the median Δ_f^{med} is calculated as follows:

$$\Delta_f^{\log} = \frac{\log \Delta_f^{\text{med}} - \Delta_f^{\text{MIN}}}{\Delta_f^{\text{MAX}} - \Delta_f^{\text{MIN}}} \log_{10}(1/10^{-8}) + \log_{10} 10^{-8},$$

where Δ_f^{MIN} and Δ_f^{MAX} are the lowest and the highest $\log\,\Delta_f^{\text{med}}$ values obtained by any of the compared algorithms for the particular benchmark function f and dimension D within the evaluation budget 250 FE/D.

Figures 3 and 4 show that S-CMA-ES and ^{s*}ACM-ES have the fastest convergence rates. S-CMA-ES converges fastest at the beginning of the optimization (till approx. 75FE/D) as well as at the later phases (125 – 250FE/D). However, it slows down from 75 till 125 FE/D, being outperformed by ^{s*}ACM-ES, especially due to the results on Shubert, CF3 (in 5D and 10D) and CF4 (in 5D and 20D).

It can be seen that POI MAES finally converges (at 250 FE/D) close to the other algorithms (outperforming CMA-ES and ^{s*}ACM-ES for 3D) for low-dimensional problems. An interesting case, however, is that POI MAES outperforms CMA-ES on 20-dimensional f_{12}. Such behavior may be associated with the fact that POI-based approach tends to explore areas with high model uncertainties which is useful for multi-modal problems [13], especially in case of high dimensionality, where the classic CMA-ES is not capable to learn the landscape sufficiently. Also, slower convergence rates of POI MAES may be explained by the fact that the model requires more time to move from exploration phase to exploitation. However, one can consider the method to be not flexible enough. The reason is that the model selects 2λ most recently evaluated points for training which is not the best decision if the algorithm moves far from the global optimum. Hence, a more sophisticated selection strategy may be required to obtain better results.

5 Conclusion

Experimental testing has shown that the best performing among all compared methods for smaller evaluation budgets (up to approx. $75\mathrm{FE}/D$) appears to be S-CMA-ES. However, it is being outperformed by the ^{s*}ACM-ES for the budgets $75-125\mathrm{FE}/D$. For even larger evaluation budgets the former method performed the best again. The POI MAES has shown a slightly worse performance compared to CMA-ES with rare improvements. However, we believe that the method can be further enhanced by introducing a new selection strategy. The method from [9] has shown inadequate performance in our experiments. However, that method was designed to perform in noisy environment, which was not the case of the employed CEC benchmark.

Today many experiments in the field of black-box optimization are conducted on the functions from the BBOB framework [2] [11]. Thereby, this paper is considered as an extension to the BBOB benchmark. The paper concentrates only on the relatively small set of test functions, which prevents from making clear conclusions. However, we believe that the performance of the methods may be clarified by the combination of this paper's results with other comparisons.

6 Acknowledgments

Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

References

[1] Anne Auger and Nikolaus Hansen. A restart CMA evolution strategy with increasing population size. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pages 1769–1776. IEEE, 2005.

[2] Lukáš Bajer, Zbyněk Pitra, and Martin Holeňa. Benchmarking Gaussian processes and random forests surrogate models on the BBOB noiseless testbed. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 1143–1150. ACM, 2015.

[3] Dirk Büche, Nicol N Schraudolph, and Petros Koumoutsakos. Accelerating evolutionary algorithms with Gaussian process fitness function models. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 35(2):183–194, 2005.

[4] John C Chaput and Jack W Szostak. Evolutionary optimization of a nonbiological ATP binding protein for improved folding stability. *Chemistry & biology*, 11(6):865–874, 2004.

[5] Nikolaus Hansen. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation*, pages 75–102. Springer, 2006.

[6] Nikolaus Hansen, Anne Auger, Steffen Finck, and Raymond Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. *INRIA*, 2010. <inria-00462481>.

[7] Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 312–317. IEEE, 1996.

[8] Yaochu Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft computing*, 9(1):3–12, 2005.

[9] Johannes W Kruisselbrink, Michael Emmerich, André H Deutz, and Thomas Bäck. A robust optimization approach using kriging metamodels for robustness approximation in the CMA-ES. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

[10] Xiaodong Li, Andries Engelbrecht, and Michael G Epitropakis. Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization.

[11] Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag. Intensive surrogate model exploitation in self-adaptive surrogate-assisted CMA-ES (saACM-ES). In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 439–446. ACM, 2013.

[12] Zbyněk Pitra, Lukáš Bajer, and Martin Holeňa. Doubly trained evolution control for the surrogate CMA-ES. Accepted for publication at PPSN 2016.

[13] Holger Ulmer, Felix Streichert, and Andreas Zell. Evolution strategies assisted by Gaussian processes with improved preselection criterion. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 1, pages 692–699. IEEE, 2003.

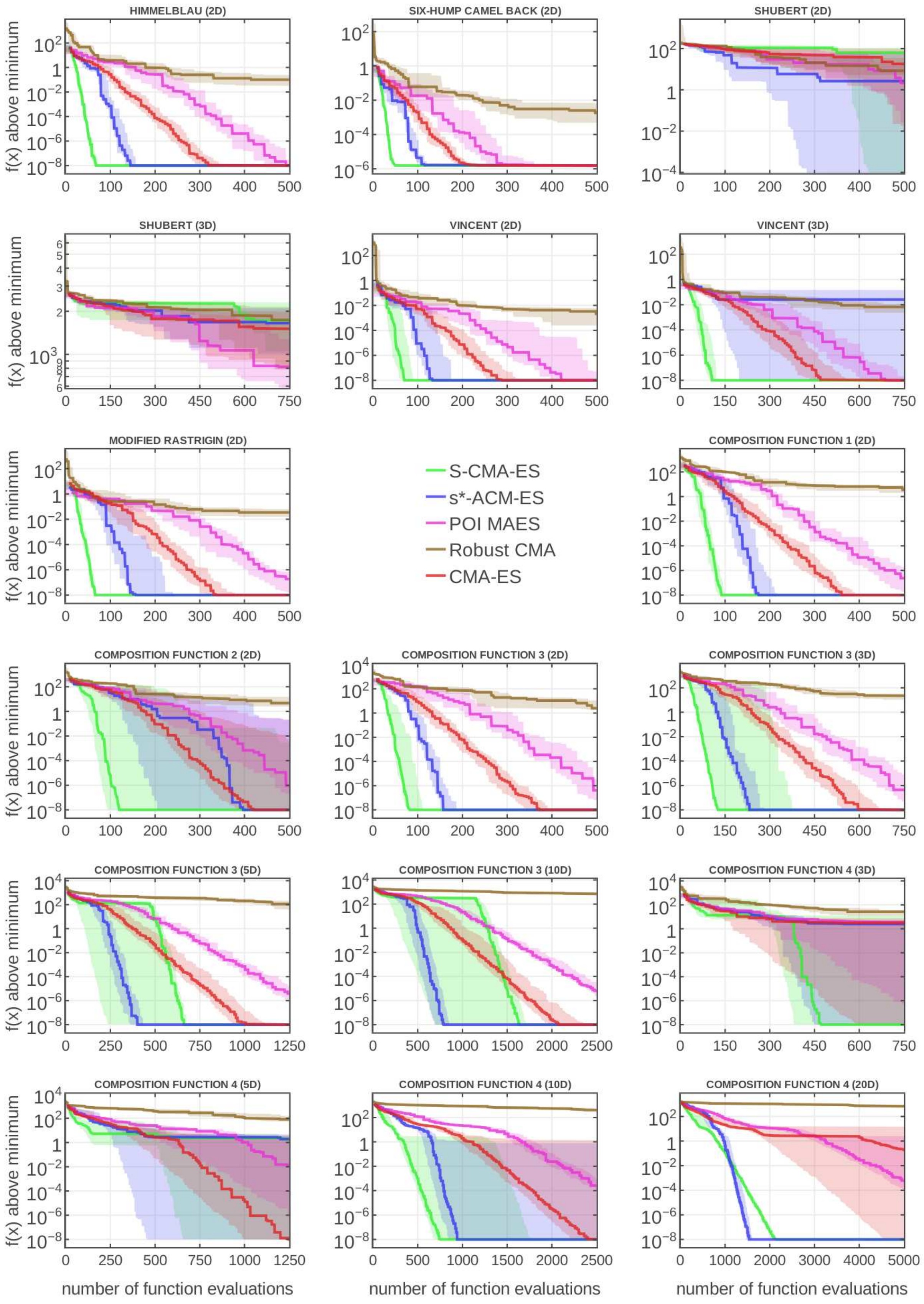

Figure 1: Convergence curves (median, first and third quartiles) computed from 15 repeated algorithm runs for feasible combinations of CEC functions and dimensions $n \in \{2,3,5,10,20\}$.

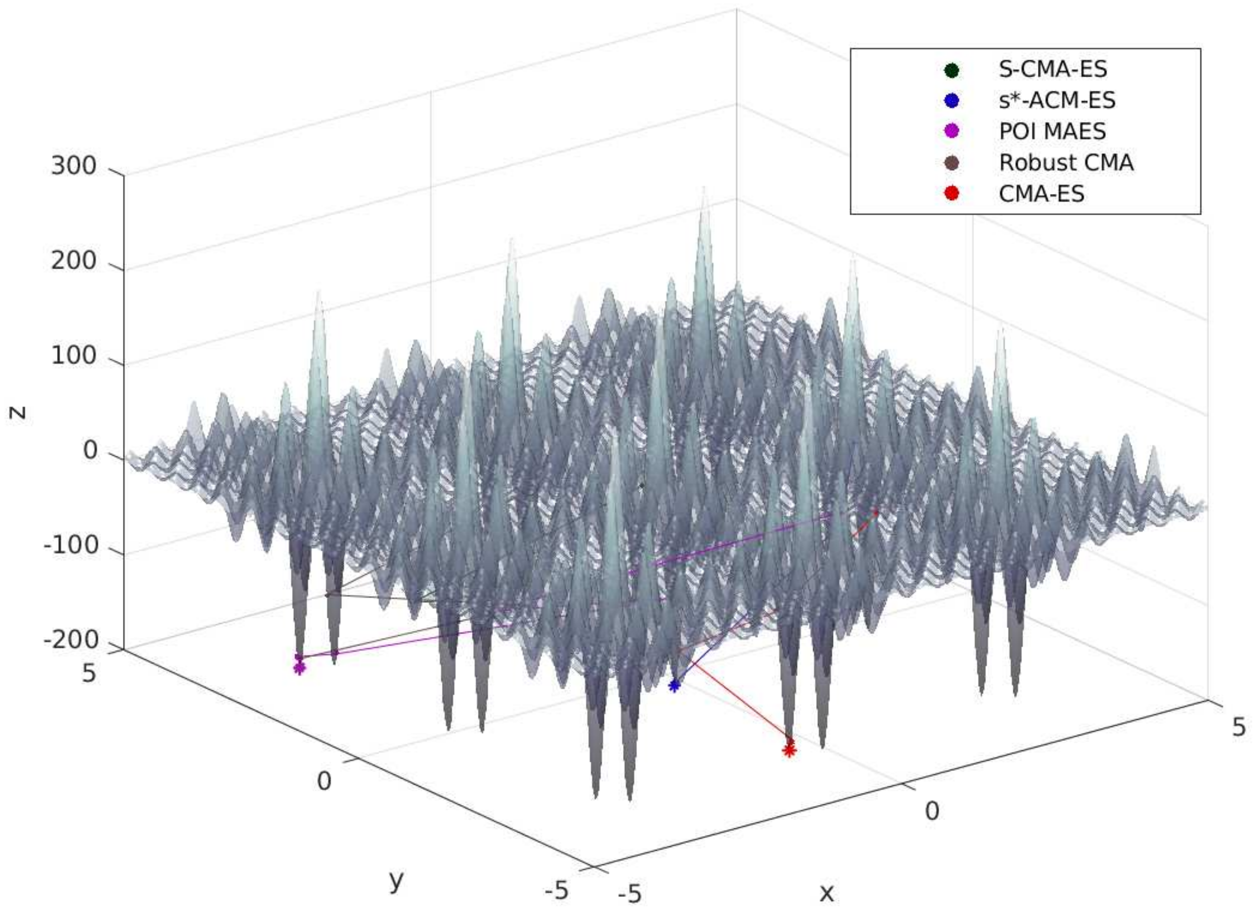

Figure 2: Best solutions found by algorithms mapped on 2D Shubert function landscape. Points represent transitional solutions found at some number of function evaluations during the optimization process. The solutions are sorted according to that number, the lines represent ordering of points. The resulting solutions found in the end of optimization are denoted by asterisks.

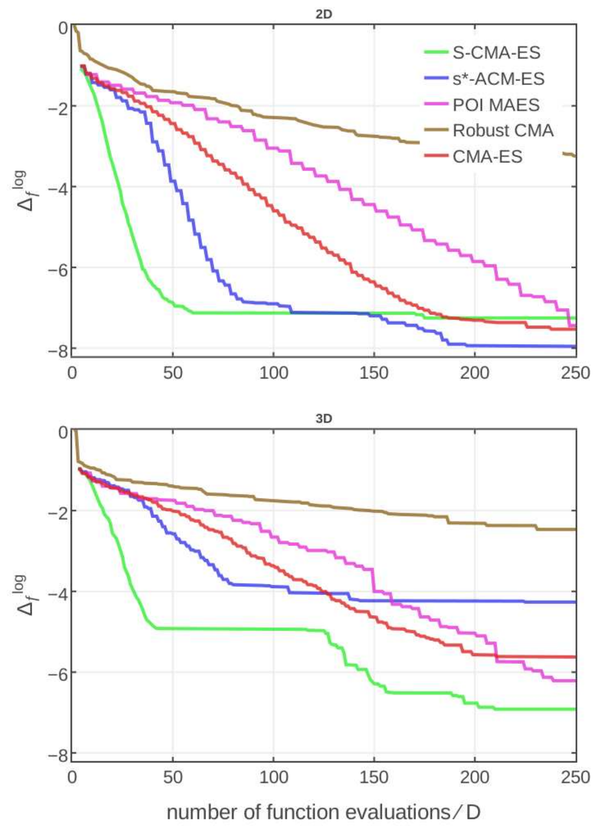

Figure 3: Scaled logarithms of the empirical medians (Δ_f^{med}) depending on FE/D. The graphs show the benchmark results achieved by averaging all functions defined in 2D and 3D.

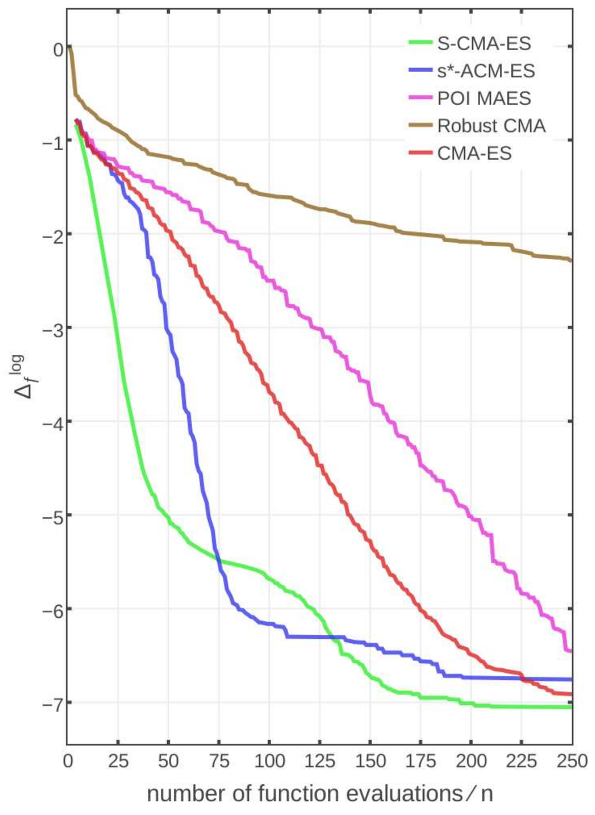

Figure 4: Scaled logarithms of the empirical medians (Δ_f^{med}) depending on FE/D. The results are aggregated over all benchmark functions.