Self-Adaptive Surrogate-Assisted Covariance Matrix Adaptation Evolution Strategy

Ilya Loshchilov
TAO, INRIA Saclay

Marc Schoenauer
TAO, INRIA Saclay

Michèle Sebag
CNRS, LRI UMR 8623

LRI, Univ. Paris-Sud, Orsay, France
firstname.lastname@inria.fr

ABSTRACT

This paper presents a novel mechanism to adapt surrogate-assisted population-based algorithms. This mechanism is applied to ACM-ES, a recently proposed surrogate-assisted variant of CMA-ES. The resulting algorithm, ^{s*}ACM-ES, adjusts online the lifelength of the current surrogate model (the number of CMA-ES generations before learning a new surrogate) and the surrogate hyper-parameters.

Both heuristics significantly improve the quality of the surrogate model, yielding a significant speed-up of ^{s*}ACM-ES compared to the ACM-ES and CMA-ES baselines. The empirical validation of ^{s*}ACM-ES on the BBOB-2012 noiseless testbed demonstrates the efficiency and the scalability w.r.t the problem dimension and the population size of the proposed approach, that reaches new best results on some of the benchmark problems.

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 black-box optimization, surrogate models, ranking support vector machine

1. INTRODUCTION

Evolutionary Algorithms (EAs) have become popular tools for optimization mostly thanks to their population-based properties and the ability to progress towards an optimum using problem-specific variation operators. A search directed by a population of candidate solutions is quite robust with respect to a moderate noise and multi-modality of the optimized function, in contrast to some classical optimization methods such as quasi-Newton methods (e.g. BFGS method [25]). Furthermore, many bio-inspired algorithms such as EAs, Differential Evolution (DE) and Particle Swarm Optimization (PSO) with rank-based selection are comparison-based algorithms, which makes their behavior invariant and robust under any monotonous transformation of the objective function. Another source of robustness is the invariance under orthogonal transformations of the search space, first introduced into the realm of continuous evolutionary optimization by Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [9]. CMA-ES, winner of the Congress on Evolutionary Computation (CEC) 2005 [5] and the Black-Box Optimization Benchmarking (BBOB) 2009 [8] competitions of continuous optimizers, has also demonstrated its robustness on real-world problems through more than one hundred applications [7].

When dealing with expensive optimization objectives, the well-known surrogate-assisted approaches proceed by learning a surrogate model of the objective, and using this surrogate to reduce the number of computations of the objective function in various ways. The best studied approach relies on the use of computationally cheap polynomial regression for the line search in gradient-based search methods, such as in the BFGS method [25]. More recent approaches rely on Machine Learning algorithms, modelling the objective function through e.g. Radial Basis Functions (RBF), Polynomial Regression, Support Vector Regression (SVR), Artificial Neural Network (ANN) and Gaussian Process (GP) a.k.a. Kriging. As could have been expected, there is no such thing as a best surrogate learning approach [14, 20]. Experimental comparisons also suffer from the fact that the results depend on the tuning of the surrogate hyper-parameters. Several approaches aimed at the adaptive selection of surrogate models during the search have been proposed [27, 28, 1]; these approaches focus on measuring the quality of the surrogate models and using the best one for the next evolutionary generation.

This paper, aimed at robust surrogate-assisted optimization, presents a surrogate-adaptation mechanism (^{s*}) which can be used on top of any surrogate optimization approach. ^{s*} adapts online the number of generations after which the surrogate is re-trained, referred to as the surrogate lifelength; further, it adaptively optimizes the surrogate hyper-parameters using an embedded CMA-ES module. A proof of principle of the approach is given by implementing ^{s*} on top of ACM-ES, a surrogate-assisted variant of CMA-ES, yielding the ^{s*}ACM-ES algorithm. To our best knowledge, the self-adaptation of the surrogate model within CMA-ES and by

Author's version.
GECCO'12, July 7–11, 2012, Philadelphia, Pennsylvania, USA.
.

CMA-ES is a new contribution. The merits of the approach are shown as ^{s*}ACM-ES show significant improvements compared to CMA-ES and ACM-ES on the BBOB-2012 noiseless testbed.

The paper is organized as follows. Section 2 reviews some surrogate-assisted variants of Evolution Strategies (ESs) and CMA-ES. For the sake of self-containedness, the ACM-ES combining CMA-ES with the use of a Rank-based Support Vector Machine is briefly described. Section 3 discusses the merits and weaknesses of ACM-ES and suggests that the online adjustment of the surrogate hyper-parameters is required to reach some robustness with respect to the optimization objective. The ^{s*}ACM-ES algorithm, including the online adjustment of the surrogate lifelength and hyper-parameters on top of ACM-ES, is described in section 4. The experimental validation of ^{s*}ACM-ES is reported and discussed in section 5. Section 6 concludes the paper.

2. SURROGATE MODELS

This section discusses the various techniques used to learn surrogate models, their use within EAs and specifically CMA-ES, and the properties of surrogate models in terms of invariance w.r.t. monotonous transformations of the objective function [24], and orthogonal transformations of the instance space [21].

2.1 Surrogate-assisted Evolution Strategies

As already mentioned, many surrogate modelling approaches have been used within ESs and CMA-ES: RBF network [11], GP [29, 4], ANN [6], SVR [30, 19], Local-Weighted Regression (LWR) [18, 3], Ranking SVM [24, 21, 12]. In most cases, the surrogate model is used as a filter (to select λ_{Pre} promising pre-children) and/or to estimate the objective function of some individuals in the current population. The impact of the surrogate, controlled by λ_{Pre}, should clearly depend on the surrogate accuracy; how to measure it ? As shown by [15], the standard Mean Square Error (MSE) used to measure a model accuracy in a regression context is ill-suited to surrogate-assisted optimization, as it is poorly correlated with the ability to select correct individuals. Another accuracy indicator, based on the (weighted) sum of ranks of the selected individuals, was proposed by [15], and used by [30, 11].

2.2 Comparison-based Surrogate Models

Taking advantage of the fact that some EAs, and particularly CMA-ES, are comparison-based algorithms, which only require the offspring to be correctly ranked, it thus comes naturally to learn a comparison-based surrogate. Comparison-based surrogate models, first introduced by Runarsson [24], rely on learning-to-rank algorithms, such as Ranking SVM [17]. Let us recall Ranking SVM, assuming the reader's familiarity with Support Vector Machines [26].

Let $(x_1, \ldots, x_N)$ denote an N-sample in instance space X, assuming with no loss of generality that point x_i has rank i. Rank-based SVM learning [17] aims at a real-valued function $\widehat{f}$ on X such that $\widehat{f}(x_i) < \widehat{f}(x_j)$ iff $i < j$. In the SVM framework, this goal is formalized through minimizing the norm of $\widehat{f}$ (regularization term) subject to the $N(N-1)/2$ ordering constraints. A more tractable formulation, also used in [24, 21], only involves the $N-1$ constraints related to consecutive points, $\widehat{f}(x_i) < \widehat{f}(x_{i+1})$ for $i = 1 \ldots N-1$.

Using the kernel trick[1], ranking function $\widehat{f}$ is defined as a linear function w w.r.t. some feature space $\Phi(X)$, i.e. $\widehat{f}(x) = \langle\, w, \Phi(x)\,\rangle$. With same notations as in [26], the primal minimization problem is defined as follows:

$$\begin{array}{l} \text{Minimize}_{\{w,\,\xi\}} \frac{1}{2}||w||^2 + \sum_{i=1}^{N} C_i \xi_i \\ \text{subj. to} \left\{ \begin{array}{l} \langle\, w, \Phi(x_i) - \Phi(x_{i+1})\,\rangle \geq 1 - \xi_i \;\; (i = 1, \; ...N-1) \\ \xi_i \geq 0 \;\; (i = 1 \ldots N-1) \end{array} \right. \end{array} \tag{1}$$

where slack variable ξ_i (respectively constant C_i) accounts for the violation of the i-th constraint (resp. the violation cost). The corresponding dual problem, quadratic in the Lagrangian multipliers α, can be solved easily by any quadratic programming solver. The rank-based surrogate $\widehat{f}$ is given as

$$\widehat{f}(x) = \sum_{i=1}^{N-1} \alpha_i (K(x_i, x) - K(x_{i+1}, x))$$

By construction, $\widehat{f}(x)$ is invariant to monotonous transformations of the objective function, which preserve the ranking of the training points.

2.3 Invariance w.r.t. Orthogonal Transformations

As already mentioned, CMA-ES is invariant w.r.t. orthogonal transformations of the search space, through adapting a covariance matrix during the search. This invariance property was borrowed by ACM-ES [21], using the covariance matrix C learned by CMA-ES within a Radial Basis Function (RBF) kernel K_C, where C^{-1} is used to compute Mahalanobis distance:

$$K_C(x_i, x_j) = e^{-\frac{(x_i - x_j)^t C^{-1} (x_i - x_j)}{2\sigma^2}} \tag{2}$$

For the sake of numerical stability, every training point x is mapped onto x' such as

$$x' = C^{-1/2}(x - m), \tag{3}$$

where m is the mean of the current CMA-ES distribution. The standard RBF kernel with Euclidean distance is used on top of this mapping. By construction, ACM-ES inherits from CMA-ES the property of invariance under orthogonal transformations of the search space; the use of the covariance matrix C brings significant improvements compared to standard Gaussian kernels after [21].

3. DISCUSSION

This section analyzes the weaknesses of ACM-ES. Following the characterization proposed in [16], ACM-ES is a surrogate-assisted optimizer with an individual-based evolution control. As in other pre-selection methods, at each generation ACM-ES generates λ_{Pre} individuals, where λ_{Pre} is much larger than population size λ. Then λ_{Pre} pre-children are evaluated and ranked using surrogate model $\widehat{f}$. The most promising λ' pre-children are selected and evaluated using the true expensive function, yielding new points $(x, f(x))$. When the objective function of λ' individuals is known, the ranking of other $\lambda - \lambda'$ points can be approximated.

[1] The so-called kernel trick supports the extension of the SVM approach from linear to non-linear model spaces, by mapping instance space X onto some *feature space* $\Phi(X)$. The actual mapping cost is avoided as the scalar product in feature space $\Phi(X)$ is computed on instance space X through a *kernel* function K: $\langle\, \Phi(x), \Phi(x')\,\rangle =_{def} K(x, x')$.

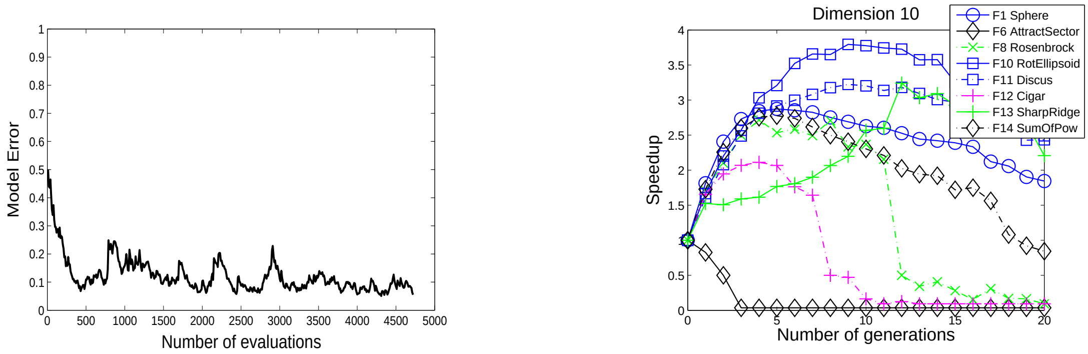

Figure 1: Left: Rank-based surrogate error vs number of evaluations, during a representative run of active CMA-ES on 10-D Rotated Ellipsoid. Right: The speedup of IPOP-aACM-ES over IPOP-aCMA-ES, where speedup = 2.0 means that IPOP-aACM-ES with a given lifelength $\widehat{n}$ of the surrogate model, requires 2.0 times less computational effort SP1 (i.e. average number of function evaluations of successful runs divided by proportion of successful runs) than IPOP-aCMA-ES to reach the target objective value of $f_{\mathrm{t}} = f_{\mathrm{opt}} + 10^{-8}$.

While our experimental results confirm the improvements brought by ACM-ES on some functions (about 2-4 times faster than CMA-ES on Rosenbrock, Ellipsoid, Schwefel, Noisy Sphere and Ackley functions up to dimension 20 [21]), they also show a loss of performance on the multi-modal Rastrigin function. Complementary experiments suggest that:

1. on highly multi-modal functions the surrogate model happens to suffer from a loss of accuracy; in such cases some control is required to prevent the surrogate model from misleading the search;
2. surrogate-assisted algorithms may require a larger population size for multi-modal problems.

The lack of surrogate control appears to be an important drawback in ACM-ES. This control should naturally reflect the current surrogate accuracy. A standard measure of the rank-based surrogate error is given as the fraction of violated ranking constraints on the test set [17]. Accuracy 0 (respectively .5) corresponds to a perfect surrogate (resp. random guessing).

However, before optimization one should be sure that the model gives a reasonable prediction of the optimized function. Fig. 1 (Left) illustrates the surrogate model error during a representative run of active CMA-ES (with re-training at each iteration, but without any exploitation of the model) on 10 dimensional Rotated Ellipsoid function. After the first generations, the surrogate error decreases to approximately 10%. This better than random prediction can be viewed as a source of information about the function which can be used to improve the search.

Let $\widehat{n}$ denote the number of generations a surrogate model is used, referred to as surrogate lifelength. In so-called generation-based evolution control methods [16], the surrogate $\widehat{f}$ is directly optimized for $\widehat{n}$ generations, without requiring any expensive objective computations. The following generation considers the objective function f, and yields instances to enrich the training set, relearn or refresh the surrogate and adjust some parameters of the algorithm. The surrogate lifelength $\widehat{n}$ is fixed or adapted.

The impact of $\widehat{n}$ is displayed on Fig. 1 (Right), showing the speedup reached by direct surrogate optimization on several 10 dimensional benchmark problems *vs* the number of generations $\widehat{n}$ the surrogate is used. A factor of speedup 1.7 is obtained for $\widehat{n}$= 1 on the Rotated Ellipsoid function, close to the optimal speedup 2.0. A speedup ranging from 2 to 4 is obtained for IPOP-aCMA-ES with surrogates for $\widehat{n}$ *in* $[5, 15]$. As could have been expected again, the optimal value of $\widehat{n}$ is problem-dependent and widely varies. In the case of the Attractive Sector problem for instance, the surrogate model is not useful and $\widehat{n} = 0$ should be used (thus falling back to the original aCMA-ES with no surrogate) to prevent the surrogate from misleading the search.

4. SELF-ADAPTIVE SURROGATE-BASED CMA-ES

In this section we propose a novel surrogate adaptation mechanism which can be used in principle on top of any iterative population-based optimizer without requiring any significant modifications thereof. The approach is illustrated on top of CMA-ES and ACM-ES. The resulting algorithm, ^{s*}ACM-ES, maintains a global hyper-parameter vector $\theta = (\theta_{opt}, \theta_{sur}, \widehat{n}, \mathcal{A}, \alpha)$, where:
• θ_{opt} stands for the optimization parameters of the CMA-ES used for expensive function optimization;
• θ_{sur} stands for the optimization parameters of the CMA-ES used for surrogate model hyper-parameters optimization;
• $\widehat{n}$ is the number of optimization generations during which the current surrogate model is used;
• $\mathcal{A}$ is the archive of all points $(x_i, f(x_i))$ for which the true objective function has been computed, exploited to train the surrogate function;
• α stands for the surrogate hyper-parameters.
All hyper-parameters are indexed by the current generation g; by abuse of notations, the subscript g is omitted when clear from the context.

The main two contributions of the paper regard the adjustment of the surrogate hyper-parameters (section 4.2) and of the surrogate lifelength $\widehat{n}$ (section 4.3).

4.1 Overview of ^{s*}ACM-ES

Let $GenCMA(h,\theta_h,\mathcal{A})$ denote the elementary optimization module (here one generation of CMA-ES) where h is the function to be optimized (the true objective f or the surrogate $\widehat{f}$), θ_h denotes the current optimization parameters

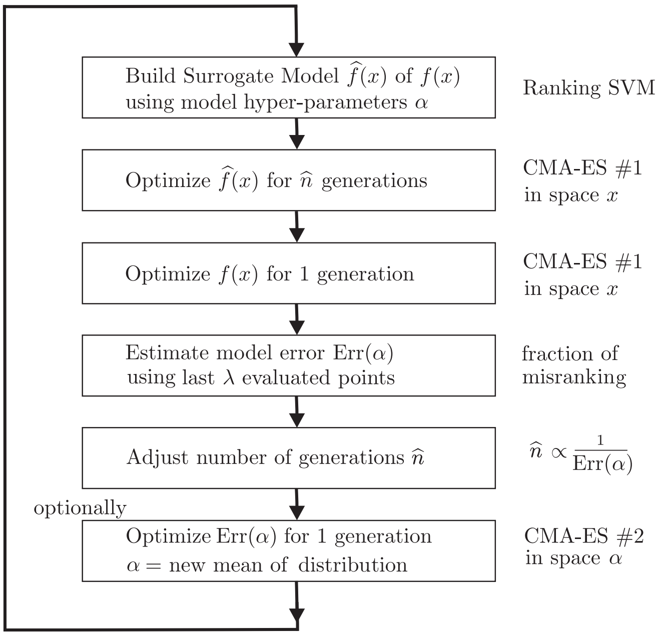
Build Surrogate Model $\widehat{f}(x)$ of $f(x)$
using model hyper-parameters α
Ranking SVM
Optimize $\widehat{f}(x)$ for $\widehat{n}$ generations
CMA-ES #1
in space x
Optimize $f(x)$ for 1 generation
CMA-ES #1
in space x
Estimate model error Err(α)
using last λ evaluated points
fraction of
misranking
Adjust number of generations $\widehat{n}$
$\widehat{n} \propto \frac{1}{\text{Err}(\alpha)}$
optionally
Optimize Err(α) for 1 generation
α = new mean of distribution
CMA-ES #2
in space α

Figure 2: Optimization loop of the ^{s*}ACM-ES.

(e.g. CMA-ES step-size and covariance matrix) associated to h, and $\mathcal{A}$ is the archive of f. After each call of *GenCMA*, optimization parameters θ_h are updated; and if *GenCMA* was called with the true objective function f, archive $\mathcal{A}$ is updated and augmented with the new points $(x, f(x))$. Note that *GenCMA* can be replaced by any black-box optimization procedure, able to update its own optimization parameters and the archive.

^{s*}ACM-ES starts by calling *GenCMA* for g_{start} number of generations with the true objective f, where θ_{opt} and $\mathcal{A}$ are respectively initialized to the default parameter of CMA-ES and the empty set (lines 4-7). In this starting phase, optimization parameter θ_{opt} and archive $\mathcal{A}$ are updated in each generation.

Then ^{s*}ACM-ES iterates a three-step process (Algorithm 1, illustrated on Fig. 2):

1 learning surrogate $\widehat{f}$ (procedure BuildSurrogateModel, line 9; section 4.2);

2 Optimizing surrogate $\widehat{f}$ during $\widehat{n}$ generations (lines 11-13). This step classically calls *GenCMA*($\widehat{f}$,θ_{opt},$\mathcal{A}$) for $\widehat{n}$ consecutive generations; θ_{opt} is updated accordingly while $\mathcal{A}$ is unchanged since this step does not involve any computation of the expensive f.

3 Adjusting the surrogate lifelength $\widehat{n}$ (section 4.3).

4.2 Learning a Surrogate and Adjusting its Hyper-parameters

The surrogate model learning phase proceeds as in ACM-ES (section 2.3). *GenCMA*(f,θ_{opt},$\mathcal{A}$) is launched for one generation with the true objective f, updating and augmenting archive $\mathcal{A}$ with new $(x, f(x))$ points.

$\widehat{f}$ is built using Ranking SVM [17] with archive $\mathcal{A}$ as training set, where the SVM kernel is tailored using the current optimization parameters θ_{opt} such as covariance matrix C.

Algorithm 1 ^{s*}ACM-ES

```
 1: $g \leftarrow 0$; Err $\leftarrow 0.5$; $\mathcal{A}_g \leftarrow \emptyset$;
 2: $\theta_{opt} \leftarrow$ InitializationCMA(); { to optimize $f(x)$ }
 3: $\theta_{sur} \leftarrow$ InitializationCMA(); { to optimize $h(\alpha)$ }
 4: repeat
 5:   $\{\theta_{opt}, \mathcal{A}_{g+1}\} \leftarrow$ GenCMA($f$,$\theta_{opt}$, $\mathcal{A}_g$);
 6:   $g \leftarrow g + 1$;
 7: until $g = g_{start}$ ;
 8: repeat
 9:   $\widehat{f}(x) \leftarrow$ BuildSurrogateModel($\alpha$, $\mathcal{A}_g$, $\theta_{opt}$);
10:   $g_{prev} \leftarrow g$;
11:   for $i = 1, \ldots, \widehat{n}$ do
12:     $\{\theta_{opt}, \mathcal{A}_{g+1} = \mathcal{A}_g\} \leftarrow$ GenCMA($\widehat{f}$,$\theta_{opt}$, $\mathcal{A}_g$);
13:     $g \leftarrow g + 1$;
14:   $\{\theta_{opt}, \mathcal{A}_{g+1}\} \leftarrow$ GenCMA($f$,$\theta_{opt}$, $\mathcal{A}_g$);
15:   $g \leftarrow g + 1$;
16:   Err($\alpha$) $\leftarrow$ MeasureSurrogateError($\widehat{f}$,$\theta_{opt}$);
17:   Err $\leftarrow (1 - \beta_{\text{Err}})$Err $+ \beta_{\text{Err}}$Err($\alpha$);
18:   $\widehat{n} \leftarrow \left\lfloor \frac{\tau_{err} - \text{Err}}{\tau_{err}} \widehat{n_{max}} \right\rfloor$ ;
19:   // adjust surrogate hyperparameters
20:   $\theta_{sur} \leftarrow$ GenCMA(Err,$\theta_{sur}$);
21:   $\alpha \leftarrow \theta_{sur}.m$;
22: until stopping criterion is met ;
```

Algorithm 2 Objective function Err(α) of surrogate model

```
1: Input: $\alpha$
2: $\widehat{f}(x) \leftarrow$ BuildSurrogateModel($\alpha$, $\mathcal{A}_{g_{prev}}$, $\theta_{sur,g_{prev}}$);
3: Err($\alpha$) $\leftarrow$ MeasureSurrogateError($\widehat{f}$, $\theta_{opt,g_{prev}}$);
4: Output: Err($\alpha$);
```

The contribution regards the adjustment of the surrogate hyper-parameters α (e.g. the number and selection of the training points in $\mathcal{A}$; the weights of the constraint violations in Ranking SVM, section 2.3), which are adjusted to optimize the quality of the surrogate Err (Eq. 4). Formally, to each surrogate hyper-parameter vector α is associated a surrogate error Err(α) defined as follows: hyper-parameter α is used to learn surrogate $\widehat{f}_\alpha$ using $\mathcal{A}_{g-1}$ as training set, and Err(α) is set to the ranking error of $\widehat{f}_\alpha$, using the most recent points $(\mathcal{A}_g - \mathcal{A}_{g-1})$ as test set.

Letting Λ denote the test set and assuming with no loss of generality that the points in Λ are ordered after f, Err($\widehat{f}_\alpha$) is measured as follows (procedure MeasureSurrogateError, line 16):

$$\text{Err}(\alpha) = \frac{2}{|\Lambda|(|\Lambda| - 1)} \sum_{i=1}^{|\Lambda|} \sum_{j=i+1}^{|\Lambda|} w_{ij}.1_{\widehat{f}_{\alpha,i,j}} \qquad (4)$$

where $1_{\widehat{f}_{\alpha,i,j}}$ holds true iff $\widehat{f}_\alpha$ violates the ordering constraint on pair (i,j). In all generality, the surrogate error can be tuned using weight coefficients w_{ij} to reflect the relative importance of ordering constraints. Only $w_{ij} = 1$ will be used in the remainder of the paper. For a better numerical stability, the surrogate error is updated using additive relaxation, with relaxation constant β_{Err} (lines 16-17).

Finally, the elementary optimization module *GenCMA*(Err,θ_{sur}) (in this study we do not use archive parameter here) is launched for one generation (line 20), and

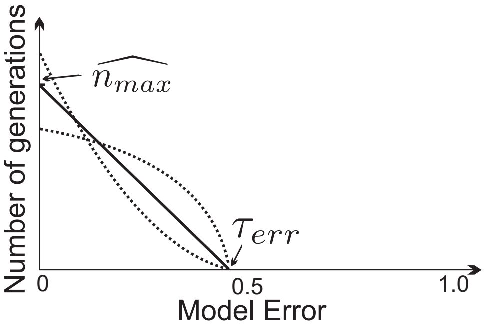

Figure 3: Number of generations $\widehat{n}$ versus surrogate error Err. Linear interpolation (bold curve) has been used in the experimental validation.

the mean of the CMA-ES mutation distribution is used (line 21) as surrogate hyper-parameter vector in the next surrogate building phase (line 9).

4.3 Adjusting Surrogate Lifelength

Lifelength $\widehat{n}_g$ is likewise adjusted depending on the error made by the previous surrogate $\widehat{f}_{g-1}$ on the new archive points $(\mathcal{A}_g - \mathcal{A}_{g-1})$. If this error is 0, then $\widehat{f}_{g-1}$ is perfectly accurate and could have been used for some more generations before learning $\widehat{f}_g$. In this case lifelength $\widehat{n}_g$ is set to the maximum value $\widehat{n_{max}}$, which corresponds to the maximum theoretical speedup of the ^{s*}ACM-ES.

If the error is circa .5, surrogate $\widehat{f}_{g-1}$ provides no better indications than random guessing and thus misleads the optimization; $\widehat{n}_g$ is set to 0. More generally, considering an error threshold τ_{err}, $\widehat{n}$ is adjusted between $\widehat{n_{max}}$ and 0, proportionally[2] to the ratio between the actual error and the error threshold τ_{err} (line 18, bold curve on Fig. 3).

5. EXPERIMENTAL VALIDATION

The experimental validation of the approach proceeds by comparing the performance of ^{s*}ACM-ES to the original [9] and active [13] CMA-ES versions, considering the restart scenario with increasing population size (IPOP [2, 10]).

The active IPOP-aCMA-ES [10] with weighted negative covariance matrix update is found to perform equally well or better than IPOP-CMA-ES, which is explained as it more efficiently exploits the information of the worst $\lambda/2$ points. We use IPOP-aCMA-ES as challenging baseline, more difficult to speed up than the original IPOP-CMA-ES.

Specifically, ^{s*}ACM-ES is validated on the noiseless BBOB testbed by comparing IPOP-aACM-ES with fixed hyper-parameters, and IPOP-^{s*}aACM-ES with online adaptation of hyperparameters of the surrogate model[3].

After detailing the experimental setting, this section reports on the offline tuning of the number $N_{training}$ of points used to learn the surrogate model, and the online tuning of the surrogate hyper-parameters.

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

Table 1: Surrogate hyper-parameters, default value and range of variation

5.1 Experimental Setting

The default BBOB stopping criterion is reaching target function value $f_t = f_{opt} + 10^{-8}$. Ranking SVM is trained using the most recent $N_{training}$ points (subsection 5.2); its stopping criterion is arbitrarily set to a maximum number of $1000 N_{training}$ iterations of the quadratic programming solver.

After a few preliminary experiments, the Ranking SVM constraint violation weights (Eq. 1) are set to

$$C_i = 10^{C_{base}} (N_{training} - i)^{C_{pow}}$$

with $C_{base} = 6$ and $C_{pow} = 3$ by default; the cost of constraint violation is thus cubically higher for top-ranked samples. The σ parameter of the RBF kernel is set to $\sigma = c_{sigma}\sigma_x$, where σ_x is the dispersion of the training points (their average distance after translation, Eq. 3) and c_{sigma} is set to 1 by default. The number g_{start} of CMA-ES calls in the initial phase is set to 10, the maximum lifelength $\widehat{n_{max}}$ of a surrogate model is set to 20. The error threshold τ_{err} is set to .45 and the error relaxation factor is set to .2.

The surrogate hyper-parameters θ_{sur} are summarized in Table 1, with offline tuned value (default for IPOP-aACM-ES) and their range of variation for online tuning, (where d stands for the problem dimension). Surrogate hyper-parameters are optimized with a population size 20 (20 surrogate models), where the Err function associated to a hyper-parameter vector is measured on the most recent λ points in archive $\mathcal{A}$, with λ the current optimization population size.

5.2 Offline Tuning: Number of Training Points

It is widely acknowledged that the selection of the training set is an essential ingredient of surrogate learning [16]. After some alternative experiments omitted for brevity, the training set includes simply the most recent $N_{training}$ points in the archive. The study thus focuses on the tuning of $N_{training}$. Its optimal tuning is of course problem- and surrogate learning algorithm-dependent. Several tunings have been considered in the literature, for instance for 10-dimensional problems: 3λ for SVR [30]; 30 for RBF [29]; 50 for ANN [6]; $\lambda, 2\lambda$ for Ranking SVM[24, 12]; $\frac{d(d+1)}{2} + 1 = 66$ for LWR in the lmm-CMA-ES [3]; $70\sqrt{d} = 221$ for Ranking SVM in the ACM-ES [21].

In all above cases but ACM-ES, the surrogate models aim at local approximation. These approaches might thus be biased toward small $N_{training}$ values, as a small number of training points are required to yield good local models (e.g. in the case of the Sphere function), and small $N_{training}$ values positively contribute to the speed-up. It is suggested

[2] Complementary experiments omitted for brevity, show that the best adjustment of $\widehat{n}$ depending on the surrogate error is again problem-dependent.

[3] For the sake of reproducibility we used the Octave/MatLab source code of IPOP-CMA-ES with default parameters, available from its author's page, with the active flag set to 1. The ^{s*}ACM-ES source code is available at `https://sites.google.com/site/acmesgecco/`.

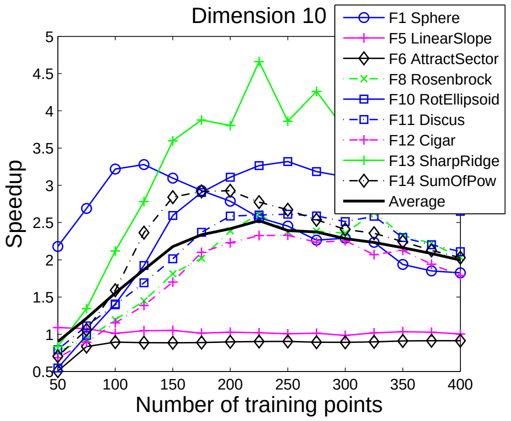

Figure 4: The speedup of IPOP-aACM-ES over IPOP-aCMA-ES w.r.t. (fixed) number of training points.

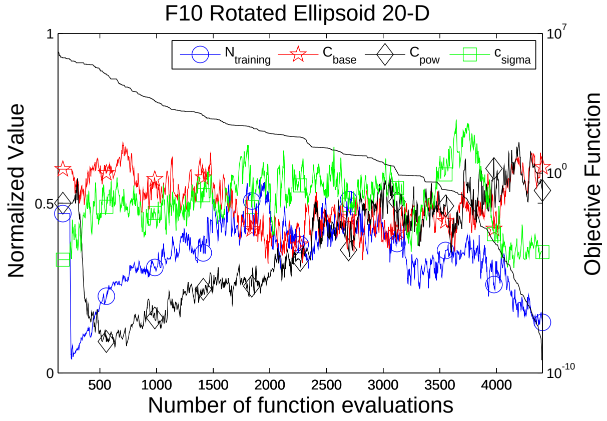

Figure 5: The median trajectories of normalized surrogate hyper-parameters estimated on 15 runs of the IPOP-^{s*}aACM-ES on Rotated Ellipsoid 20-D.

however that the Sphere function might be misleading, regarding the optimal adjustment of $N_{training}$.

Let us consider the surrogate speed-up of IPOP-aACM-ES w.r.t. IPOP-CMA-ES depending on (fixed) $N_{training}$, on uni-modal benchmark problems from the BBOB noiseless testbed (Fig. 4 for $d = 10$). While the optimal speed-up varies from 2 to 4, the actual speed-up strongly depends on the number $N_{training}$ of training points.

Complementary experiments on d-dimensional problems with $d = 2, 5, 10, 20, 40$ (Fig. 4) yield to propose an average best tuning of $N_{training}$ depending on dimension d:

$$N_{training} = \lfloor 40 + 4d^{1.7} \rfloor \qquad (5)$$

Eq. (5) is found to empirically outperform the one proposed in [21] ($N_{training} = \left\lfloor 70\sqrt{d} \right\rfloor$), which appears to be biased to 10-dimensional problems, and underestimates the number of training points required in higher dimensions. Experimentally however, $N_{training}$ must super-linearly increase with d; eq. (5) states that for large d the number of training points should triple when d doubles.

Further, Fig. 4 shows that the optimal $N_{training}$ value is significantly smaller for the Sphere function than for other functions, which experimentally supports our conjecture that the Sphere function might be misleading with regard to the tuning of surrogate hyper-parameters.

5.3 Online Tuning: Surrogate Hyper-parameters

The IPOP-^{s*}ACM-ES achieves the online adaptation of the surrogate hyper-parameters within a specified range (Table 1), yielding the surrogate hyper-parameter values to be used in the next surrogate learning step.

Note that a surrogate hyper-parameter individual might be non-viable, i.e. if it does not enable to learn a surrogate model (Ranking SVM fails due to an ill-conditioned kernel). Such non-viable individual is heavily penalized (Err(α) $>>$ 1). In case no usable hyper-parameter individual is found (which might happen in the very early generations as it is shown on Fig. 5), θ_{sur} is set to its default value.

The online adaptation of surrogate hyper-parameters however soon reaches usable hyper-parameter values. The trajectory of the surrogate hyper-parameter values vs the number of generations is depicted in Fig. 5, normalized in $[0, 1]$ and considering the median out of 15 runs optimizing 20 dimensional Rotated Ellipsoid function.

The trajectory of $N_{training}$ displays three stages. In a first stage, $N_{training}$ increases as the overall number of evaluated points (all points are required to build a good surrogate). In a second stage, $N_{training}$ reaches a plateau; its value is close to the one found by offline tuning (section 5.2). In a third stage, $N_{training}$ steadily decreases. This last stage is explained as CMA-ES approaches the optimum of f and gets a good estimate of the covariance matrix of the Ellipsoid function. At this point the optimization problem is close to the Sphere function, and a good surrogate can be learned from comparatively few training points.

The trajectories of other surrogate hyper-parameters are more difficult to interpret, although they clearly show non-random patterns (e.g. C_{pow}).

5.4 Comparative Performances

The comparative performance of ^{s*}ACM-ES combined with the original and the active variants of IPOP-CMA-ES is depicted on Fig. 6, on the 10-d Rotated Ellipsoid (Left) and Rosenbrock (Right) functions. In both cases, the online adaptation of the surrogate hyper-parameters yields a quasi constant speed-up, witnessing the robustness of ^{s*}ACM-ES. On the Ellipsoid function, the adaptation of the covariance matrix is much faster than for the baseline, yielding same convergence speed as for the Sphere function. On the Rosenbrock function the adaptation is also much faster, although there is clearly room for improvements.

The performance gain of ^{s*}ACM-ES, explained from the online adjustment of the surrogate hyper-parameters, in particular $N_{training}$, confirms the fact that the appropriate surrogate hyper-parameters vary along search, and can be adjusted based on the accuracy of the current surrogate model. Notably, IPOP-^{s*}ACM-ES almost always outperforms IPOP-ACM-ES, especially for $d > 10$.

5.5 Scalability w.r.t Population Size

The default population size $\lambda_{default}$ is suggested to be the only CMA-ES parameter to possibly require manual tuning. Actually, $\lambda_{default}$ is well tuned for uni-modal problems and only depends on the problem dimension. Increasing the pop-

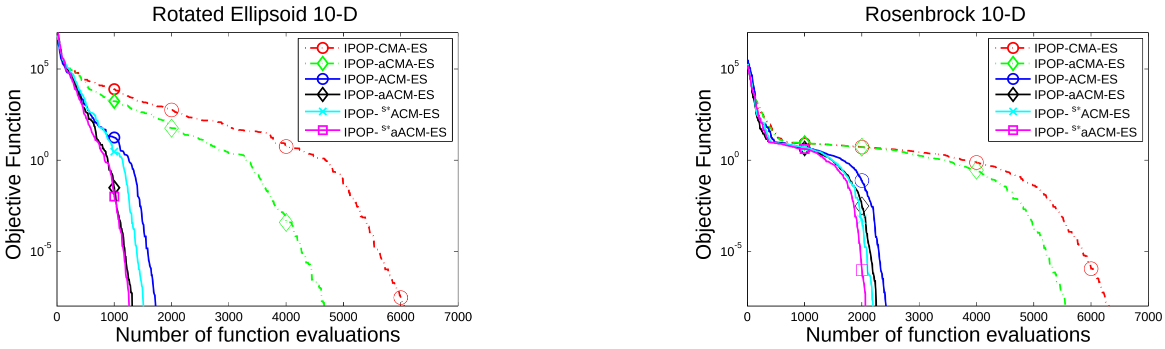

Figure 6: Comparison of the proposed surrogate-assisted versions of the original and active IPOP-CMA-ES algorithms on 10 dimensional Rotated Ellipsoid (Left) and Rosenbrock (Right) functions. The trajectories show the median of 15 runs.

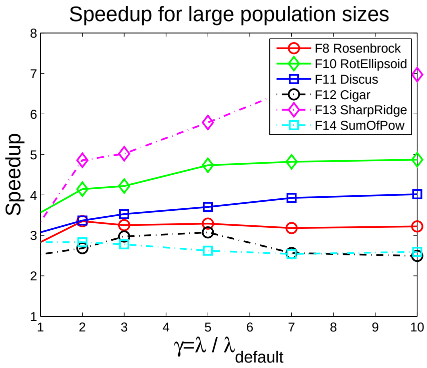

Figure 7: Speedup of the IPOP-^{s*}aACM-ES over IPOP-aCMA-ES for large population sizes $\lambda = \gamma\lambda_{default}$ on 10-D problems.

ulation size does not decrease the overall number of function evaluations needed to reach an optimum in general. Still, it allows one to reach the optimum after fewer generations. Increasing the population size and running the objective function computations in parallel is a source of speed-up, which raises the question of ^{s*}ACM-ES scalability w.r.t. the population size.

Fig. 7 shows the speedup of the IPOP-^{s*}aACM-ES compared to IPOP-aCMA-ES for unimodal 10 dimensional problems, when the population size λ is set to γ times the default population size $\lambda_{default}$. For F8 Rosenbrock, F12 Cigar and F14 Sum of Different Powers the speedup remains almost constant and independent of γ, while for F10 Rotated Ellipsoid, F11 Discus and F13 Sharp Ridge, it even increases with γ. We believe that with a larger population size, "younger" points are used to build the surrogate model, that is hence more accurate.

The experimental evidence suggests that ^{s*}ACM-ES can be applied on top of parallelized versions of IPOP-^{s*}aACM-ES, while preserving or even improving its speed-up. Note that the same does not hold true for all surrogate-assisted methods; for instance in trust region methods, one needs to sequentially evaluate the points.

It is thus conjectured that further improvements of CMA-ES (e.g. refined parameter tuning, noise handling) will translate to ^{s*}ACM-ES, without degrading its speed-up.

6. CONCLUSION AND PERSPECTIVES

This paper presents a generic framework for adaptive surrogate-assisted optimization, which can in principle be combined with any iterative population-based optimization, and surrogate learning, algorithms. This framework has been instantiated on top of surrogate-assisted ACM-ES, using CMA-ES as optimization algorithm and Ranking SVM as surrogate learning algorithm. The resulting algorithm, ^{s*}ACM-ES, inherits from CMA-ES and ACM-ES the property of invariance w.r.t. monotonous transformations of the objective function and orthogonal transformations of the search space.

The main contribution of the paper regards the online adjustment of i) the number $\widehat{n}$ of generations a surrogate model is used, called surrogate lifelength; ii) the surrogate hyper-parameters controlling the surrogate learning phase. The surrogate lifelength is adapted depending on the quality of the current surrogate model; the higher the quality, the longer the next surrogate model will be used. The adjustment of the surrogate hyper-parameters is likewise handled by optimizing them w.r.t. the quality of the surrogate model, without requiring any prior knowledge on the optimization problem at hand.

IPOP-^{s*}aACM-ES was found to improve on IPOP-aCMA-ES with a speed-up ranging from 2 to 3 on uni-modal d-dimensional functions from the BBOB-2012 noiseless testbed, with dimension d ranging from 2 to 40. On multi-modal functions, IPOP-^{s*}aACM-ES is equally good or sometimes better than IPOP-aCMA-ES, although the speed-up is less significant than for uni-modal problems. Further, IPOP-^{s*}aACM-ES also improves on IPOP-aCMA-ES on problems with moderate noise from BBOB-2012 noisy testbed. All these results as well as the computational complexity of the algorithm are discussed in details in BBOB-2012 workshop papers [22] and [23].

A long term perspective for further research is to better handle multi-modal and noisy functions. A shorter-term perspective is to consider a more comprehensive surrogate learning phase, involving a portfolio of learning algorithms and using the surrogate hyper-parameter optimization phase to achieve portfolio selection. Another perspective is to design a tighter coupling of the surrogate learning phase, and the CMA-ES optimization, e.g. using the surrogate model $\widehat{f}$ to adapt the CMA-ES hyper-parameters during the optimization of the expensive objective f.

7. ACKNOWLEDGMENTS

The authors would like to acknowledge Anne Auger, Zyed Bouzarkouna, Nikolaus Hansen and Thomas P. Runarsson for their valuable discussions. This work was partially funded by FUI of System@tic Paris-Region ICT cluster through contract DGT 117 407 *Complex Systems Design Lab* (CSDL).

8. REFERENCES

[1] G. T.-P. Alan Diaz-Manriquez and W. Gomez-Flores. On the selection of surrogate models in evolutionary optimization algorithms. In *Proc. CEC'2011*, pages 2143–2150. IEEE Press, 2011.

[2] A. Auger and N. Hansen. A restart CMA evolution strategy with increasing population size. In *Proc. CEC'2005*, pages 1769–1776. IEEE Press, 2005.

[3] Z. Bouzarkouna, A. Auger, and D. Ding. Investigating the local-meta-model CMA-ES for large population sizes. In C. Di Chio et al., editor, *Proc. EvoNUM'10*, pages 402–411. LNCS 6024, Springer, 2010.

[4] M. T. Emmerich, K. C. Giannakoglou, and B. Naujoks. Single- and Multiobjective Evolutionary Optimization Assisted by Gaussian Random Field Metamodels. *IEEE TEC*, 10(4):421–439, 2006.

[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*, 15:617–644, 2009.

[6] L. Graning, Y. Jin, and B. Sendhoff. Efficient evolutionary optimization using individual-based evolution control and neural networks: A comparative study. In *Proc. ESANN'2005*, pages 27–29, 2005.

[7] N. Hansen. References to CMA-ES applications. http://www.lri.fr/ hansen/cmaapplications.pdf, 2009.

[8] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík. Comparing results of 31 algorithms from the BBOB-2009. In *GECCO Workshop Proc.* ACM, 2010.

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

[10] N. Hansen and R. Ros. Benchmarking a weighted negative covariance matrix update on the bbob-2010 noiseless testbed. In *In GECCO Workshop Proc.*, pages 1673–1680. ACM, 2010.

[11] 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.

[12] H. Ingimundardottir and T. Runarsson. Sampling strategies in ordinal regression for surrogate assisted evolutionary optimization. In *Proc. ISDA'2011*, page To appear, 2011.

[13] G. A. Jastrebski and D. V. Arnold. Improving evolution strategies through active covariance matrix adaptation. In *Proc. CEC'2006*, pages 2814–2821, 2006.

[14] R. Jin, W. Chen, and T. W. Simpson. Comparative studies of metamodeling techniques under multiple modeling criteria. *Structural and Multidisciplinary Optimization*, 23:1–13, 2000.

[15] Y. Jin. Quality measures for approximate models in evolutionary computation. In *GECCO Workshop Proc.*, pages 170–173. AAAI, 2003.

[16] Y. Jin. A Comprehensive Survey of Fitness Approximation in Evolutionary Computation. *Soft Computing*, 9(1):3–12, 2005.

[17] T. Joachims. A support vector method for multivariate performance measures. In L. D. Raedt and S. Wrobel, editors, *Proc. ICML'05*, volume 119 of *ACM ICPS*, pages 377–384. ACM, 2005.

[18] S. Kern, N. Hansen, and P. Koumoutsakos. Local meta-models for optimization using evolution strategies. In Th. Runarsson et al., editor, *PPSN IX*, pages 939–948. LNCS 4193, Springer, 2006.

[19] O. Kramer. Covariance matrix self-adaptation and kernel regression - perspectives of evolutionary optimization in kernel machines. *Fundam. Inf.*, 98:87–106, 2010.

[20] D. Lim, Y.-S. Ong, Y. Jin, and B. Sendhoff. A study on metamodeling techniques, ensembles, and multi-surrogates in evolutionary computation. In *Proc. GECCO'2007*, pages 1288–1295. ACM, 2007.

[21] I. Loshchilov, M. Schoenauer, and M. Sebag. Comparison-Based Optimizers Need Comparison-Based Surrogates. In J. K. R. Schaefer, C. Cotta and G. Rudolph, editors, *Proc. PPSN XI*, pages 364–373. LNCS 6238, Springer, 2010.

[22] 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 *In GECCO Companion*, page To appear. ACM, 2012.

[23] I. Loshchilov, M. Schoenauer, and M. Sebag. Black-box optimization benchmarking of IPOP-saACM-ES on the BBOB-2012 noisy testbed. In *In GECCO Companion*, page To appear. ACM, 2012.

[24] T. P. Runarsson. Ordinal regression in evolutionary computation. In Th. Runarsson et al., editor, *PPSN IX*, pages 1048–1057. LNCS 4193, Springer, 2006.

[25] D. F. Shanno. Conditioning of Quasi-Newton methods for function minimization. *Mathematics of Computation*, 24(111):647–656, 1970.

[26] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge Univ. Press, 2004.

[27] L. Shi and K. Rasheed. Asaga: an adaptive surrogate-assisted genetic algorithm. In *Proc. GECCO'2008*, pages 1049–1056, 2008.

[28] Y. Tenne and S. W. Armfield. A versatile surrogate-assisted memetic algorithm for optimization of computationally expensive functions and its engineering applications. In *Success in Evolutionary Computation*, pages 43–72. Springer, 2008.

[29] H. Ulmer, F. Streichert, and A. Zell. Evolution strategies assisted by gaussian processes with improved pre-selection criterion. In *Proc. CEC'2003*, pages 692–699, 2003.

[30] H. Ulmer, F. Streichert, and A. Zell. Evolution strategies with controlled model assistance. In *Proc. CEC'2004*, pages 1569 – 1576, 2004.