Local Meta-Models for Optimization using Evolution Strategies

Stefan Kern, Nikolaus Hansen, and Petros Koumoutsakos

Computational Science and Engineering Laboratory,
Institute of Computational Science,
ETH Zurich, Switzerland
{skern,hansenn,petros}@inf.ethz.ch

Abstract. We employ local meta-models to enhance the efficiency of evolution strategies in the optimization of computationally expensive problems. The method involves the combination of second order local regression meta-models with the Covariance Matrix Adaptation Evolution Strategy. Experiments on benchmark problems demonstrate that the proposed meta-models have the potential to reliably account for the ranking of the offspring population resulting in significant computational savings. The results show that the use of local meta-models significantly increases the efficiency of already competitive evolution strategies.

1 Introduction

The optimization of a large number of engineering processes, ranging from multidisciplinary design to manufacturing, can only be formulated as black-box problems. The fitness function in this context is usually computationally expensive and may involve noise and multiple optima. Evolutionary Algorithms (EAs) have been shown to cope successfully with noise and multimodality, and there is an ongoing effort to further extend their efficiency for expensive problems by incorporating local or global meta-models of the fitness function [1]. The use of meta-models based on *global* function approximation, even for moderate dimension, is hindered by the inhomogeneity of the data collected during the optimization. Several methods have been proposed to overcome this difficulty ranging from restricting the training data to the closest, most recently evaluated points [2] to sophisticated sequential update techniques [3, 4]. Alternatively, *local* meta-models have been developed ranging from simple nearest neighborhood regression to local quadratic models [5–7].

Meta-models have been shown to improve the efficiency of EAs in many cases, but a number of open questions remain, including the choice of the meta-model complexity with regard to the underlying EA, as well as the balance between the use of the meta-model and the true objective. In this paper we address these open problems by investigating the use of *local* meta-models of varying complexity in conjunction with Covariance Matrix Adaptation (CMA-ES) [8–10]. The CMA-ES employs rank based selection, relaxing for the meta-model the requirement

of approximating the objective function. We propose a *local meta-model CMA-ES* (lmm-CMA) and investigate its performance on benchmark problems. We find that the lmm-CMA enhances significantly the performance of the standard CMA-ES.

The paper is organized as follows: In Sect. 2 we address model quality measures and the control of data points used as meta model support. Section 3 gives an introduction to Locally Weighted Regression as a class of meta models for EAs. In Sect. 4, the choice of complexity of the local model is investigated. In Sect. 5 we propose the lmm-CMA and determine the optimal bandwidth of the local model. In Sect. 6 the performance of the proposed lmm-CMA is examined on well known test-functions and compared to previous results. A summary and conclusions are presented in Sect. 7.

2 Meta-model quality and controlled model assistance

In meta-modeling the definition of optimal prediction needs to be consistent with the operators of the optimization algorithm [11]. Optimal prediction is usually associated with a minimum error in the quantitative approximation of the objective function by the meta-model. For rank-based EAs maintaining the fitness based ranking of the population is sufficient and therefore more appropriate.

Measuring meta-model quality. In this paper we use meta-model quality measures in order to: (i) investigate the optimal complexity of the local models to be learned, and (ii) control the adaptive use of the local models in the EA. In both cases we are interested in the deviation of the offspring ranking predicted by the meta-model $\hat{f}$ from the true ranking determined by the fitness function f in each generation g.

When the true fitness function values $y_i = f(\boldsymbol{x}_i)$ are known for the *complete* population of size λ, we propose a quality measure adopted from sorting that counts pair inversions in the approximate ranking. For an approximate ranking $\hat{F} = \langle \hat{y}_1, \ldots, \hat{y}_\lambda \rangle$, with $\hat{y}_i \leq \hat{y}_j, 1 \leq i < j \leq \lambda$, the normalized pair inversion count ρ_{inv} is defined as

$$\rho_{inv}(\hat{F}) = \frac{4}{\lambda(\lambda-1)} \left| \{(i,j) | 1 \leq i < j \leq \lambda \quad \text{and} \quad f(\boldsymbol{x}_{r(i)}) > f(\boldsymbol{x}_{r(j)})\} \right|, \quad (1)$$

where $r(i)$ is the index mapping function determined by the model based ranking, i.e. $\hat{y}_i = \hat{f}(\boldsymbol{x}_{r(i)})$. The normalization factor $\lambda(\lambda-1)/4$ is the expected pair inversion count for a randomly ranked population which can be easily proven by induction; it holds $0 \leq \rho_{inv} \leq 2$. If not all individuals in one generation are evaluated, the measure (1) cannot be applied since the correct ranking of the population is only partially known.

Meta-model assisted ranking procedure. An elegant way to control model quality without knowing the correct ranking of the complete population is the *approximate ranking procedure* [7]: In every generation, the offspring are successively

1 *approximate*: build $\hat{f}(\boldsymbol{x}_k), k = 1, \ldots, \lambda$ based on evaluations in training set $\mathcal{S}$
2 *rank*: based on $\hat{f}$ generate ranking$_0^\mu$ of the μ best individuals
4 *evaluate*: n_{init} best individuals based on $\hat{f}$, add to $\mathcal{S}$
5 **for** $i := 1$ **to** $(\lambda - n_{\text{init}})/n_b$ **do**
6 *approximate*: build $\hat{f}(\boldsymbol{x}_k), k = 1, \ldots, \lambda$ based on $\mathcal{S}$
7 *rank*: based on $\hat{f}$ generate ranking$_i^\mu$ of the μ best individuals
8 **if** (ranking$_{i-1}^\mu$ == ranking$_i^\mu$) **then** *(ranking of μ best remains unchanged)*
10 **break** *(exit for loop)*
11 **else** *(ranking of μ best individuals changed)*
12 *evaluate:* n_b next best unevaluated points based on $\hat{f}$, add to $\mathcal{S}$
13 **fi**
14 **od**
15 **if** $(i > 2)$ **then** $n_{\text{init}} = \min(n_{\text{init}} + n_b, \lambda - n_b)$
16 **elseif** $(i < 2)$ **then** $n_{\text{init}} = \max(n_b, n_{\text{init}} - n_b)$

Fig. 1. Approximate ranking procedure that is executed in every generation to determine the fraction of points evaluated on the fitness function. The procedure is not called until sufficiently many evaluations are stored in the training set $\mathcal{S}$ to build the model; initialization of $n_{\text{init}} = \lambda$.

evaluated and added to the training set of the fitness function model until the (deterministic) model based selection of the parents remains unchanged in two consecutive iteration cycles. This results in an adaptive control mechanism determining the number of evaluated individuals in every generation. The CMA-ES uses the ranked $\mu = \lambda/2$ best offspring to update its Gaussian mutation distribution [9]. We adapt the *approximate ranking procedure* to the requirements of CMA-ES: the predicted ranking in the μ first positions should not change for the meta-model iteration to stop. For large population sizes λ, often required to solve multimodal functions [9], the amount of information added in one iteration may result in insignificant changes even of a meta-model with bad ranking predictions. To overcome this deficiency we suggest to evaluate a batch of individuals in every meta-model iteration. We use a batch size n_b proportional to λ and choose $n_b = \max(1, \lfloor \lambda/10 \rfloor)$. The total cost of the meta model loop can be further reduced by introducing an adaptive parameter to specify the number of initial evaluations, n_{init}, performed before the model iteration loop is entered. The resulting meta-model assisted ranking procedure is outlined in Fig. 1.

3 Locally weighted regression

Locally weighted regression (LWR) [12] attempts to fit the training data (*here*: past evaluations of the fitness function stored in a database) only in a region around the location of the query. The local models are built consecutively as queries need to be answered and therefore are intrinsically designed for growing training data sets as they occur in the course of an optimization. In the following we give a brief introduction to LWR following the notation in [12].

For every offspring to be predicted an individual model is built. Given a set of points $(\boldsymbol{x}_j, y_j), j = 1, \ldots, m$, the training criterion C is minimized w.r.t. the parameters $\boldsymbol{\beta}$ of the local mode $\hat{f}$ at query point $\boldsymbol{q}$ and can be written as

$$C(\boldsymbol{q}) = \sum_{j=1}^{m} \left[(\hat{f}(\boldsymbol{x}_j, \boldsymbol{\beta}) - y_j)^2 K\left(\frac{d(\boldsymbol{x}_j, \boldsymbol{q})}{h}\right)\right], \tag{2}$$

where $K(.)$ is the kernel weighting function, $d(\boldsymbol{x}_j, \boldsymbol{q})$ the distance between data point $\boldsymbol{x}_j$ and $\boldsymbol{q}$, and h is the (local) bandwidth. We consider $\hat{f}$ linear in $\boldsymbol{\beta}$, i.e. $\hat{f}(\boldsymbol{x}, \boldsymbol{\beta}) = \tilde{\boldsymbol{x}}^T \boldsymbol{\beta}$ (cf. Table 1), and thus we can directly weight the training points and minimize (2) by solving the normal equations

$$\left((\boldsymbol{W}\tilde{\boldsymbol{X}})^T \boldsymbol{W}\tilde{\boldsymbol{X}}\right) \boldsymbol{\beta} = (\boldsymbol{W}\tilde{\boldsymbol{X}})^T \boldsymbol{W}\boldsymbol{y}, \tag{3}$$

where $\tilde{\boldsymbol{X}} = (\tilde{\boldsymbol{x}}_\mathbf{1}, \ldots, \tilde{\boldsymbol{x}}_{\boldsymbol{m}})^T$, $\boldsymbol{y} = (y_1, \ldots, y_m)^T$, and $\boldsymbol{W} = \mathrm{diag}(\sqrt{K(d(\boldsymbol{x}_i, \boldsymbol{q})/h)})$. For a given model structure, K, d, and h remain to be chosen determining the locality and smoothness of the model.

For the calculation of $d(\boldsymbol{x}_j, \boldsymbol{q})$ we propose to utilize the metric of the search distribution of the EA. Evolution strategies as the CMA-ES adapt a multivariate Gaussian mutation distribution $\mathcal{N}(\boldsymbol{m}, \boldsymbol{C})$ to the (local) topography of the function, and the covariance matrix $\boldsymbol{C}$ naturally defines a metric that can be exploited in the calculation of d as fully weighted Euclidean distance

$$d(\boldsymbol{x}_j, \boldsymbol{q}) = \sqrt{(\boldsymbol{x}_j - \boldsymbol{q})^T \boldsymbol{C}^{-1} (\boldsymbol{x}_j - \boldsymbol{q})}. \tag{4}$$

Experiments using different kernel functions K showed insignificant variation in prediction performance. We use a bi-quadratic kernel function defined as

$$K(\zeta) = \begin{cases} (1 - \zeta^2)^2 & \text{if} \quad \zeta < 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

for the remainder of this paper. Because the density of the data points collected in the course of an optimization run changes considerably, an adaptive choice of the bandwidth h is essential. We use a *nearest neighbor bandwidth selection*, where h is set to the distance of the kth nearest neighbor data point to $\boldsymbol{q}$ and thus the volume increases and decreases in size according to the density of nearby data. In this way changes in scale of the distance function d are canceled by the choice of h, giving a scale invariant distribution of the weights to the data. The optimal choice of k is addressed in Sect. 5.

4 Choice of Model Complexity

A meta model can speed up the convergence of an EA if it is capable to provide information about the fitness function not yet incorporated in the search distribution. The choice of a suitable complexity for the local model involves two

Table 1. Locally polynomial models tested in the LWR framework.

$\hat{f}_{\mathrm{x}} = \boldsymbol{\beta}_{\mathrm{x}}^T \tilde{\boldsymbol{x}}_{\mathrm{x}}$,	$\tilde{\boldsymbol{x}}_{\mathrm{x}}$,	$\boldsymbol{\beta}_{\mathrm{x}}$	dim
$\hat{f}_{\mathrm{wmean}}$	$\tilde{\boldsymbol{x}}_w = 1$	$\boldsymbol{\beta}_w = \sum_{j=1}^m w_j f_j$,	1
$\hat{f}_{\mathrm{linear}}$	$\tilde{\boldsymbol{x}}_l = (x_1, \ldots, x_n, 1)^T$	$\boldsymbol{\beta}_l$: minimize (2),	$n+1$
$\hat{f}_{\mathrm{dquad}}$	$\tilde{\boldsymbol{x}}_d = (x_1^2, \ldots, x_n^2, x_1, \ldots, x_n, 1)^T$	$\boldsymbol{\beta}_d$: minimize (2),	$2n+1$
$\hat{f}_{\mathrm{quad}}$	$\tilde{\boldsymbol{x}}_q = (x_1^2, \ldots, x_n^2,$ $x_1x_2, \ldots, x_{n-1}x_n, x_1, \ldots, x_n, 1)^T$	$\boldsymbol{\beta}_q$: minimize (2),	$\frac{n(n+3)}{2}+1$

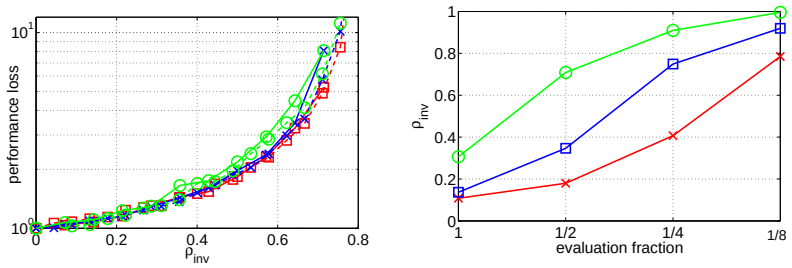
Table 2. Test-functions and coordinate-wise initialization intervals.

Name	Function	Init
Sphere	$f_{\mathrm{Sphere}}(\boldsymbol{x}) = \sum_{i=1}^n x_i^2$	$[-3, 7]^n$
Noisy Sphere	$f_{\mathrm{NoisySphere}}(\boldsymbol{x}) = f_{\mathrm{Sphere}}(\boldsymbol{x})\,(1 + \epsilon\mathcal{N}(0,1))$	$[-3, 7]^n$
Schwefel	$f_{\mathrm{Schwefel}}(\boldsymbol{x}) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j\right)^2$	$[-10, 10]^n$
Ellipsoid	$f_{\mathrm{Ellipse}}(\boldsymbol{x}) = \sum_{i=1}^n \left(100^{\frac{i-1}{n-1}} y_i\right)^2$	$[-3, 7]^n$
Rosenbrock	$f_{\mathrm{Rosenbrock}}(\boldsymbol{x}) = \sum_{i=1}^{n-1} \left(100 \cdot (x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right)$	$[-5, 5]^n$
Ackley	$f_{\mathrm{Ackley}}(\boldsymbol{x}) = 20 - 20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^n x_i^2}\right)$ $+e - \exp\left(\frac{1}{n}\sum_{i=1}^n \cos(2\pi x_i)\right)$	$[1, 30]^n$
Rastrigin	$f_{\mathrm{Rastrigin}}(\boldsymbol{x}) = 10n + \sum_{i=1}^n \left(y_i^2 - 10\cos(2\pi y_i)\right)$	$[1, 5]^n$

questions: (i) How good is the ranking prediction of a model, and (ii) how is the performance of the baseline optimization algorithm influenced by perturbations of the ranking introduced by erroneous models?

We investigate the performance loss for the CMA-ES caused by erroneous offspring rankings by running the strategy with artificially introduced ranking perturbations of given pair inversion count ρ_{inv} (1). The performance loss is computed as ratio between the number of function evaluations needed to reach the function value of $f_{\mathrm{stop}} = 10^{-10}$ with the erroneous and the correct ranking. A rank perturbation of fixed ρ_{inv} can be produced by uniformly sampling swaps of neighbors in the ranking and only conducting a swap if it increases ρ_{inv}. This procedure is repeated until the target ρ_{inv} is reached. Figure 2a shows the expected performance loss of the CMA-ES on f_{Sphere}, f_{Ellipse}, and $f_{\mathrm{Rosenbrock}}$ (see Table 2) for dimension $n = 5$ and 10 versus the pair inversion count ρ_{inv}. The data was obtained by averaging 20 runs with a uniformly random starting point from the interval given in Table 2. The performance loss shows only minor dependency on the function and the dimensionality.

We measured the quality of ranking predictions of constant, linear, and quadratic (with and without cross terms) local models as given in Table 1. The training data was obtained from independent runs of the CMA-ES. All four models where tested on 20 data sets for each of the three test-functions using fractions of $1, 1/2, 1/4$ and $1/8$ of the evaluated points, randomly chosen in every generation. The bandwidth parameter k was varied as 1, 2, and 4 times the number of free parameters of the model. Figure 2b gives the result for $\hat{f}_{\mathrm{quad}}$ on



(a) (b)

Fig. 2. (a) Mean performance loss of the CMA-ES with perturbed offspring ranking on f_{Sphere} (□), f_{Ellipse} (×), and $f_{\text{Rosenbrock}}$ (◯) for $n = 5$ (solid lines), and $n = 10$ (dashed lines). (b) Normalized pairwise inversion count of the predicted ranking versus evaluation fraction for $\hat{f}_{\text{quad}}$ on $f_{\text{Rosenbrock}}$ in 10D and bandwidth parameter $k = [1, 2, 4] \times \frac{n(n+3)+2}{2}$ (bottom to top).

$f_{\text{Rosenbrock}}$ in 10D, showing the loss in rank prediction quality as less function evaluations are used.

Combining the data of Fig. 2a & b, the speedup potential of a meta-model can be estimated[1] for a given evaluation fraction. Figure 3 depicts the speedup potential of the four investigated local models on f_{Sphere}, f_{Ellipse}, and $f_{\text{Rosenbrock}}$ with the optimal bandwidth parameter k. For none of the three functions $\hat{f}_{\text{wmean}}$ or $\hat{f}_{\text{linear}}$ a speedup factor of more than 1.5 is predicted; in 10D the results even predict a negative speedup for all three test-functions. $\hat{f}_{\text{dquad}}$ shows perfect speedup on f_{Sphere}, however already for randomly oriented misscaled convex quadratic functions the expected speedup does not exceed a factor of 2 in 10D. Only the full quadratic model $\hat{f}_{\text{quad}}$ is capable to predict reliable rankings to enhance convergence of CMA-ES on the non-quadratic $f_{\text{Rosenbrock}}$. The speedup potential is between 2 and 3 at an evaluation rate of about 1/3 to 1/4.

5 The lmm-CMA

The investigations in the previous section revealed that only *full quadratic* local models have the potential to significantly improve the convergence speed of the CMA-ES. We therefore enhance CMA-ES with a local quadratic meta-model using the adapted *approximate ranking procedure* presented in Fig. 1. The algorithm is referred to as lmm-CMA.

To complete lmm-CMA the optimal bandwidth for the locally weighted quadratic regression remains to be chosen. We investigate the influence of k for the k-th nearest neighbor bandwidth selection on the number of function evaluations of lmm-CMA to reach $f_{\text{stop}} = 10^{-10}$ on the non-quadratic $f_{\text{Rosenbrock}}$ for dimension $n = 2, 4, 8$, and 16. The parameter k is varied according to $k = \alpha\, k_{\text{min}}$

[1] making the (optimistic) assumption that the use of the surrogate does not change the optimization path

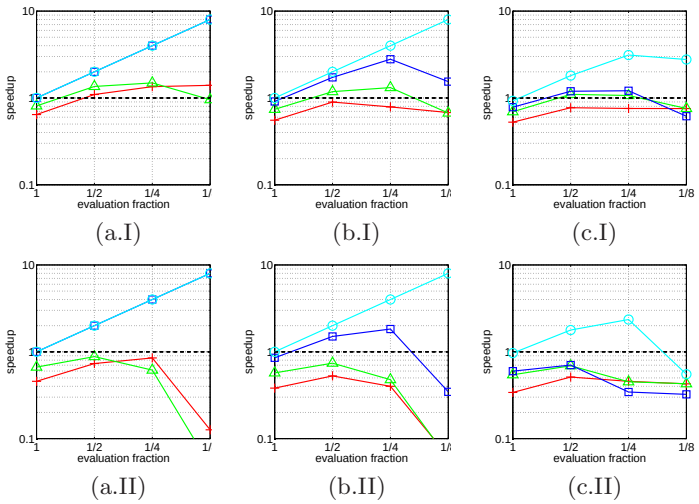


Fig. 3. Speedup potential of $\hat{f}_{\mathrm{wmean}}$ (+), $\hat{f}_{\mathrm{linear}}$ (Δ), $\hat{f}_{\mathrm{dquad}}$ ($\square$), and $\hat{f}_{\mathrm{quad}}$($\bigcirc$) on (a) f_{Sphere}, (b) f_{Ellipse}, and (c) $f_{\mathrm{Rosenbrock}}$ in dimension (I) $n = 5$, and (II) $n = 10$.

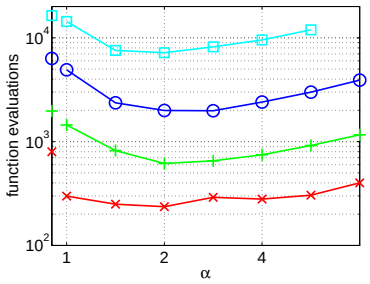


Fig. 4. Average number of function evaluations to reach f_{stop} of the lmm-CMA on $f_{\mathrm{Rosenbrock}}$ for varying bandwidth parameter $k = \alpha \cdot (n(n+3)/2+1)$ and $n = 2\ (\times), 4\ (+), 8\ (\bigcirc)$, and $16\ (\square)$. The points on the y-axis show the performance of the original CMA-ES. The optimal performance of lmm-CMA is observed for $\alpha = 2$.

with $\alpha = 2^{i/2}, i = 0, \ldots, 6$, where $k_{\mathrm{min}} = \frac{n(n+3)}{2} + 1$ is the number of free parameters of the local quadratic model. Every data point is obtained by averaging 20 independent runs of lmm-CMA. The results in Fig. 4 show a unique minimum for $\alpha = 2$ independent of dimension. Therefore, we set $k = n(n+3)+2$ for all experiments conducted in the following.

6 Performance of the lmm-CMA

The proposed lmm-CMA is investigated on a set of uni- and multimodal test-functions summarized in Table 2. The performance is assessed by averaging the number of function evaluations needed to reach $f_{\mathrm{stop}} = 10^{-10}$ from 20 inde-

Table 3. Average number of function evaluations and standard deviations to reach f_{stop} of lmm-CMA versus CMA-ES, GPOP [2], and `fminunc`. For the multimodal functions, the numbers are divided by the probability to find the global optimum given in brackets. `fminunc` diverges on $f_{\text{NoisySphere}}$ (†) and has a vanishing probability to converge to the global optimum on f_{Ackley} and $f_{\text{Rastrigin}}$ for the given initialization region.

Function	n	λ	ϵ	lmm-CMA	CMA-ES	GPOP	`fminunc`
$f_{\text{Schwefel}}(\boldsymbol{x})$	2	6		81 ±5	391 ±42	40	**24** ±5
	4	8		145 ±7	861 ±53	110	**96** ±7
	8	10		**282** ±11	2035 ±93	440	428 ±22
	16	12		**626** ±17	5263 ±115	6000	1684 ±37
$f_{\text{Rosenbrock}}(\boldsymbol{x})$	2	6		263 ±87 (1.0)	799 ±119 (1.0)	180	**119** ±38 (1.0)
	4	8		674 ±103 (1.0)	1973 ±291 (.95)	700	**344** ±52 (.85)
	8	10		2494 ±511 (.90)	6329 ±747 (.85)	2500	**1057** ±119 (.95)
	16	12		7299 ±1154 (1.0)	16388 ±1414 (.95)	14000	**3628** ±226 (.90)
$f_{\text{NoisySphere}}(\boldsymbol{x})$	2	6	0.35	**184** ±24	372 ±39	-	†
	4	8	0.25	**503** ±56	855 ±93	-	†
	8	10	0.18	**1179** ±103	1645 ±84	-	†
	16	12	0.13	**2700** ±112	3073 ±94	-	†
$f_{\text{Ackley}}(\boldsymbol{x})$	2	5		**308** ±33 (.95)	728 ±51 (.95)	-	∞ (0.0)
	5	7		**1095** ±81 (1.0)	1767 ±74 (1.0)	-	∞ (0.0)
	10	10		**3029** ±106 (1.0)	3637 ±110 (1.0)	-	∞ (0.0)
	20	10		**8150** ±196 (1.0)	6155 ±409 (1.0)	-	∞ (0.0)
$f_{\text{Rastrigin}}(\boldsymbol{x})$	2	50		**1360** ±264 (.85)	1982 ±325 (.85)	-	∞ (0.0)
	5	140		**7320** ±1205 (.85)	8486 ±1160 (.85)	-	∞ (0.0)
	10	500		**29250** ±2769 (1.0)	40152 ±5409 (.95)	-	∞ (0.0)

pendent runs, randomly initialized in the given intervals. For the underlying CMA-ES we use the standard parameter settings given in [9] except for the population size λ: for the multimodal functions we choose the optimal λ from [9, Fig. 2]. In Table 3 the results are compared to the standard CMA-ES without meta-model support, the Gaussian Process Optimization Procedure (GPOP)[2] [2], and MATLAB's `fminunc`[3]. `fminunc` implements the BFGS Quasi-Newton method with a mixed quadratic and cubic line search procedure. In the present context of black-box optimization, gradients are estimated via finite difference approximation.

On the convex quadratic f_{Schwefel}, lmm-CMA improves CMA-ES by a factor of 5-8, and on $f_{\text{Rosenbrock}}$ the speedup is 2-3. Compared to `fminunc`, lmm-CMA is at most a factor of two slower (on $f_{\text{Rosenbrock}}$), but on f_{Schwefel} it performs even better for $n \geq 8$. The results of GPOP on f_{Schwefel} and $f_{\text{Rosenbrock}}$ are competitive for $n \leq 4$. However, for larger n the Gaussian Process Regression model gets less reliable and the performance deteriorates. Note that for small n

[2] In GPOP the optimal size of the training data set depends on the problem and the problem dimension. For the comparison we take the best data presented in [2].

[3] We set 'LargeScale'='off', 'TolFun'=1e-10, 'TolX'=1e-15, and 'MaxFunEvals'=1e6.

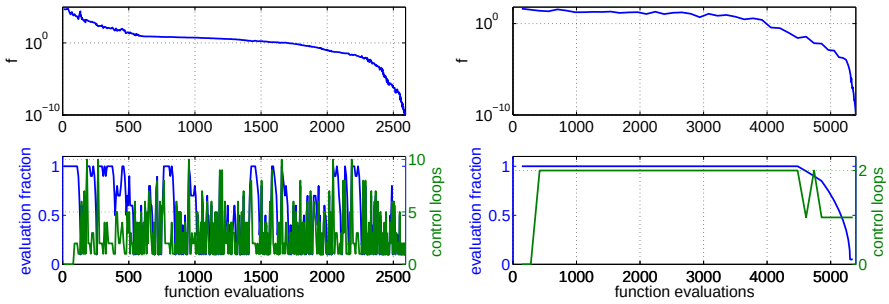


Fig. 5. Evolution of the evaluation fraction and the iteration loop count in the course of typical runs of lmm-CMA plotted aside the convergence of the fitness f on $f_{\text{Rosenbrock}}$, $n = 10$, (a), and $f_{\text{Rastrigin}}$, $n = 5$, (b).

the performance gain of lmm-CMA is limited by the n_b evaluations performed in every generation ($n_b = 1$ for $\lambda \leq 10$) and it generally scales well in n.

On $f_{\text{NoisySphere}}$ with fitness proportional Gaussian noise `fminunc` fails to converge due to the finite difference gradient estimation. In contrast, lmm-CMA and CMA-ES are able to cope with the noise levels as given in Table 3. The lmm-CMA shows a small advantage that vanishes with increasing dimension.

On the multimodal functions f_{Ackley} and $f_{\text{Rastrigin}}$ lmm-CMA is advantageous in small dimensions ($n \leq 10$), but the improvement compared to pure CMA-ES decays with increasing n. This might be an effect of suboptimal bandwidth selection of the local model for multimodal problems. Nevertheless, the results on $f_{\text{Rastrigin}}$ indicate that the adapted approximate ranking procedure works robustly with large populations.

Figure 5 exemplifies the evolution of the evaluation fraction and the iteration loop count in the course of typical runs of lmm-CMA. On $f_{\text{Rosenbrock}}$ the evaluation fraction varies throughout the whole search in the process of building local approximations of the non-quadratic function. The average evaluation fraction of $\sim \frac{1}{3}$ matches the predictions of Sect. 4 well. An interesting behavior can be observed on $f_{\text{Rastrigin}}$: In the initial (global) search phase, the local meta-models are not able to predict reliable rankings and thus are not used. However, as soon as the strategy finds the attraction region of the global optimum, the meta-model predictions get reliable and the local convergence is considerably accelerated.

7 Summary and Conclusion

The objective of this work was to enhance the performance of CMA-ES in the optimization of expensive fitness functions by incorporating local meta-models. We investigated the necessary model complexity using training data obtained from the CMA-ES. As a result, only *full quadratic* local models seem to have the potential to achieve a significant speed up. We demonstrate that locally

weighted polynomial regression can preserve the ranking of the objective function enhancing significantly the performance of an already highly competitive ES on a number of benchmark problems.

The resulting lmm-CMA outperforms the standard CMA-ES on unimodal test functions by a factor between 2 and 8 scaling well with increasing dimension n. On noisy and multimodal functions, the speedup does not exceed a factor of 3 and vanishes with increasing dimension. Nevertheless, the meta-model does not jeopardize the performance even when the function cannot be modeled effectively. Therefore its main drawback remains the computational complexity of n^6 for building the meta-model and we hope to reduce the computational cost to n^4 in future implementations. Finally, we expect to be able to improve the performance in particular on multimodal and noisy functions by implementing a more sophisticated choice of the model bandwidth.

References

1. Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft Computing **9**(1) (2005) 3–12
2. Büche, D., Schraudolph, N.N., Koumoutsakos, P.: Accelerating evolutionary algorithms with Gaussian process fitness function models. IEEE Trans. Sys., Man Cyber. **35**(2) (2005) 183–194
3. Huang, D., Allen, T.T., Notz, W.I., Zeng, N.: Global optimization of stochastic black-box systems via sequential kriging meta-models. J. Glob. Opt. **34**(3) (2006) 441–466
4. Jin, Y., Olhofer, M., Sendhoff, B.: A framework fo evolutionary optimization with approximate fitness functions. IEEE Trans. Evol. Comput. **6**(5) (2002) 481–494
5. Branke, J., Schmidt, C., Schmeck, H.: Efficient fitness estimation in noisy environments. In Spector, J., ed.: Genetic and Evolutionary Computation, Morgan Kaufmann (2001) 243–250
6. Regis, R.G., Shoemaker, C.A.: Local function approximation in evolutionary algorithms for the optimization of costly functions. IEEE Trans. Evol. Comput. **8**(5) (2004) 490–504
7. Runarsson, T.P.: Constrained evolutionary optimization by approximate ranking and surrogate models. In Yao, X., et al., eds.: Parallel Problem Solving from Nature - PPSN VIII, LNCS 3242, Springer (2004) 401–410
8. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evol. Comput. **9**(2) (2001) 159–195
9. Hansen, N., Kern, S.: Evaluating the CMA evolution strategy on multimodal test functions. In Yao, X., et al., eds.: Parallel Problem Solving from Nature - PPSN VIII, LNCS 3242, Springer (2004) 282–291
10. Auger, A., Hansen, N.: A restart CMA evolution strategy with increasing population size. In: Proceedings of the IEEE Congress on Evolutionary Computation. (2005)
11. Jin, Y., Hüsken, M., Sendhoff, B.: Quality measures for approximate models in evolutionary computation. In Barry, A.M., ed.: GECCO 2003: Proceedings of the Bird of a Feather Workshop, Genetic and Evolutionary Computation Conference, AAAI (2003) 170–173
12. Atkeson, C.G., Moore, A.W., Schaal, S.: Locally weighted learning. Artificial Intelligence Review **11**(1-5) (1997) 11–73