Online Selection of Surrogate Models for Constrained Black-Box Optimization

Samineh Bagheri
Wolfgang Konen
Cologne University of Applied Sciences
(TH Köln) Compus Gummersbach
Email: {samineh.bagheri, wolfgang.konen}@th-koeln.de

Thomas Bäck
Department of Computer Science
Leiden University, LIACS,
2333 CA Leiden, The Netherlands
Email: T.H.W.Baeck@liacs.leidenuniv.nl

***Abstract*—Real-world optimization problems are often subject to many constraints which are expensive to be evaluated in terms of cost and time. Surrogate-assisted optimization techniques aim to reduce evaluation costs by substituting the objective and the constraint functions with cheap surrogate models e. g. radial basis functions. Selecting the correct basis function and its associated parameters is a challenging task and depends on many factors like initial design, population size and type of function to be modeled, which is usually unknown in black-box problems. We propose an online model selection strategy that sorts different models according to their approximation error and selects in each iteration the best model for each function. The model selection strategy is embedded in the SACOBRA optimizer [1]. We show that the proposed online model selection strategy boosts the overall performance of the algorithm when applied to 24 well-known constrained optimization problems.**

1. Introduction

A constrained optimization problem (COP) can be defined as the minimization of an objective function (fitness function) f subject to inequality constraint function(s) $g_1, \ldots, g_m$ and equality constraint function(s) $h_1, \ldots, h_r$:

$$\begin{array}{lll} \text{Minimize} & f(\vec{x}), & \vec{x} \in [\vec{a}, \vec{b}] \subset \mathbb{R}^d \\ \text{subject to} & g_i(\vec{x}) \leq 0, & i = 1, 2, \ldots, m \\ & h_j(\vec{x}) = 0, & j = 1, 2, \ldots, r \end{array} \quad (1)$$

where $\vec{a}$ and $\vec{b}$ define the lower and upper bounds of the search space (a hypercube). By negating the fitness function f a minimization problem can be transformed into a maximization problem without loss of generality. Real-world optimization problems are often multi-constrained and high-dimensional.

Surrogate models are getting more and more attention in recent years as a valuable tool for COPs where for example the objective and constraint function values are outputs of computationally expensive computer simulations. The basic idea inherent in many approaches of surrogate-assisted optimization [2], [3], [4], [5], [6] is to do most of the optimization work on the surrogate models and to use the true functions just for the selected infill points.

The question only briefly touched in most of the literature is: Which type of surrogate model should be chosen? Even if we restrict ourselves to radial basis functions (RBF) there is a wealth of possible RBF types: cubic, Gaussian with different widths, multiquadric with different widths, to name only a few. Rocha [7] has shown that different types of RBF may exhibit oppositional performance on modeling different classes of functions. However, most work on surrogate-assisted optimization makes only *one* a-priori choice of RBF type and fixes this type for all problems investigated.

The problem is even amplified in the case of COP where not only *one* objective function is to be modeled but a set of functions (objective plus constraints). Having the work of Rocha [7] in mind, it is highly probable that a good RBF type for one of these functions might be suboptimal for another function. Therefore, it would be desirable to allow for different RBF types when modeling different functions. No other surrogate-assisted optimization approach in the literature we are aware of does allow different RBF types for different COP functions.

In this work we try to answer the following research questions:

- **(H1)** Are there COPs which significantly benefit from using different types of RBF functions for objective and constraint functions?
- **(H2)** Is it possible to advise an online algorithm which automatically selects the right RBF type for each function? That is, does such an algorithm boost up the overall performance of a COP solver?

1.1. Related Work

Online model selection plays an important role for general surrogate modeling, not only in optimization.

Gorissen et al. [8] developed the EMS framework (evolutionary model type selection) for approximation tasks using a genetic algorithm to optimize the hyperparameters of each model and a ranking strategy for different classes of surrogates. This algorithm tries to find the best model which minimizes the cross validation error or Akaikes Information Criterion (AIC). Later, Mehmani et al. [9] introduced

a new online model selection algorithm which utilizes a multi-objective solver to select a surrogate by minimizing two error measures. The authors claim that their algorithm significantly outperforms other model selection algorithms which try to minimize CV error. Field [10] criticizes the fact that the so-far developed model selection algorithms do not take any criteria into account regarding the purpose of the surrogate use. He claims that the best model fit of a function is not necessarily the best model for optimizing it.

Couckuyt et al. [11] apply the EMS framework to **unconstrained** optimization processes. Their results indicate that using the online model selection algorithm EMS is better than using a fixed Kriging model for black-box optimization processes. Müller and Shoemaker [12] develop a surrogate-assisted optimizer which combines different classes of surrogates (Kriging, cubic RBF, polynomial regression and MARS) with different weights. The weights of each surrogate class are reassigned in each iteration proportional to the calculated LOOCV error for each single model. The algorithm has high computational costs and the optimization results show no significant improvement to use model ensembles. Friese et al. [13] apply different model selection algorithms to optimization tasks as well, but do not achieve significant improvements for model ensembles.

There is only very little work devoted to surrogate model selection in the area of **constrained** optimization problems (COPs). This is particularly astonishing since - as stated in the introduction - especially for the different COP functions one might expect great benefits from using different surrogate types. Villanueva et al. [5] address low-dimensional COPs with one constraint and perform online model selection. The models are selected from a set of candidates including Kriging, response surfaces with different kernels, and moving least squares. They use LOOCV error to assess the surrogates' performance. However, the main focus of their paper is to compare (for each COP function) the use of different surrogate models for different parts of the search space with one surrogate model in the whole search space. They do not compare with fixed model choices (one surrogate type for *all* COP functions). Furthermore, Bhattacharjee et al. [14] propose a surrogate-assisted optimizer for multi-objective problems which selects different models for objective and constraint functions but this study mainly concerns about comparing the performance of global and local modeling.

In this work we propose a novel online model selection algorithm for COPs. This algorithm uses an evaluation measure to assess the performance of surrogates which is cheaper than the classical measures like CV or LOOCV. We further claim that the use of LOOCV can be problematic in the case of optimization, because points far from the current optimum contribute to the selection decision. Instead we use only the most recent points from the optimization process. We investigate this approach on a large set of benchmark functions.

TABLE 1. COMMONLY USED RADIAL BASIS FUNCTIONS

Type of basis function	$\phi(r)$
Parameter-free RBF	
Cubic	r^3
Thin plate spline	$r^2 logr$
RBF with width parameter	
Gaussian	$e^{-(\frac{r}{w})^2}$
Multiquadric (MQ)	$\sqrt{1+(\frac{r}{w})^2}$

2. Methods

2.1. RBF Interpolation

The RBF approximation – originally developed by Hardy [15] – is now one of the most important multidimensional modeling algorithms. RBF interpolation approximates a function with a linear combination of basis functions $\phi(r)$ and a polynomial tail $p(\vec{x})$. Based on n design points $\vec{x}_i, i = 1, \ldots, n$ a surrogate model $s(\vec{x})$ is formed:

$$s(\vec{x}) = \sum_{i=1}^{n} \lambda_i \cdot \phi(r_i) + p(\vec{x}), \quad (2)$$

$$\text{with} \quad p(\vec{x}) = \sum_{j=1}^{d} (a_j x_j^2 + b_j x_j + c)$$

where $\phi(r_i)$ is a function of sample point's distance $r_i = ||\vec{x} - \vec{x}_i||$ to the ith design point.

The basis functions $\phi(r)$ can be categorized into piecewise smooth RBFs without any parameter and infinitely smooth RBFs with a hyperparameter often called shape parameter ϵ or width factor $w = \frac{1}{\epsilon}$ [16]. Table 1 shows a list of commonly used radial basis functions.

2.2. SACOBRA: Self-Adjusting Constrained Black-Box Optimization with RBF

SACOBRA [1], [17], [18] is a surrogate-assisted optimizer especially designed for high-dimensional constrained black-box optimization problems. SACOBRA is an extension of the algorithm COBRA [19]. Both algorithms use RBF interpolation to model objective and constraint functions separately and then use a constrained optimizer to solve the optimization problem on the surrogates. SACOBRA adds several extensions to COBRA which are described in more detail in the cited literature and help to increase the overall performance:

- Repair algorithm for infeasible solutions [18],
- Self-adjusting techniques to tune sensitive parameters automatically [1],
- Equality handling techniques [17].

2.3. Proposed Online Model Selection

The SACOBRA optimizer [1] works in the following way: Given a set of already visited points $P =$

$\left\{x^{(1)}, \cdots x^{(n)}\right\}$ in the search space, it builds surrogate models for each COP function $F \in \{f, g_i, h_j\}$ (objective function and constraint function(s) according to Eq. (1)). It performs constrained optimization on these surrogate models to deliver the next infill point $\vec{x}_{new}$.

In the original SACOBRA algorithm all surrogates were constructed from the same RBF model class or RBF type (e. g. all cubic or all Gaussian). It is the new element of this paper that we allow each COP function to have a surrogate model from a different model class in each iteration and advise an online procedure to select this model class.

Algorithm 1 shows this procedure. The set M of model classes or RBF types might include for example: {cubic, MQ with width 0.1, MQ with width 0.2, ...}. Note that Algorithm 1 is called independently for each COP function $F \in \{f, g_i, h_j\}$. The criterion for selecting model m_{k^*} is the minimization of the approximation error on the last L infill points. The **approximation error** is the deviation between surrogate model and function F at a point $\vec{x}_{new}$ (see Step 2 of Algorithm 1). $\vec{x}_{new}$ is a point not used during surrogate model build. The underlying assumption for such a criterion is that a model with a small approximation error on the last infill points is also good for constrained optimization, i. e. the search for the next infill point.

After Algorithm 1 has provided a surrogate model $s^{(n)}$ for each COP function F, the next iteration of the SACOBRA algorithm is started until the total budget is exhausted.

Algorithm 1 Online Model Selection. **Input:** Infill point $\vec{x}_{new}$, population $P = \left\{x^{(1)}, \cdots x^{(n)}\right\}$ excluding $\vec{x}_{new}$, pool of models: $M = \{m_0, m_1, \cdots m_K\}$, true COP function F, length L of sliding window. **Output:** Surrogate model $s^{(n)}$ built with selected model m_{k^*}.

1: Build surrogate models s_k for F using all points in P with all $m_k \in M$
2: Calculate approximation error $e_k^{(n)} = |s_k^{(n)}(\vec{x}_{new}) - F(\vec{x}_{new})|$
3: Calculate median error: $E_k = \text{median}(e_k^{(n)}, e_k^{(n-1)}, \cdots, e_k^{(n-L+1)})$
4: Select model m_{k^*} with $k^* = \arg\min_k E_k$
5: Build surrogate model $s^{(n)}$ for F based on the selected model m_{k^*} using the union of points $P \cup \vec{x}_{new}$
6: Return $s^{(n)}$

3. Experiments

The G-problem suite, described in [20], is a set of well-known and challenging constrained optimization problems. The fact that G-problems have very diverse characteristics makes them a suitable testbed for evaluating COP solvers. We apply the proposed algorithm on all 24 problems, among them 11 problems with equality constraints. The characteristics of the studied problems are listed in Table 2. The quantities FR and GR were estimated for each problem by Monte Carlo sampling with 10^6 random points in the search space.

TABLE 2. CHARACTERISTICS OF THE G-FUNCTIONS: d: DIMENSION, ρ^*: FEASIBILITY RATE (%), FR: RANGE OF THE FITNESS VALUES, GR: RATIO OF LARGEST TO SMALLEST CONSTRAINT RANGE, LI/NI: NUMBER OF LINEAR/NONLINEAR INEQUALITIES, LE/NE: NUMBER OF LINEAR/NONLINEAR EQUALITIES, a: NUMBER OF CONSTRAINTS ACTIVE AT THE OPTIMUM.

Fct.	d	ρ^*	FR	GR	LI / NI	LE / NE	a
G01	13	0.0003%	298.14	1.969	9 / 0	0 / 0	6
G02	20	99.997%	0.57	2.632	1 / 1	0 / 0	1
G03	20	0.0000%	$9.27 \cdot 10^{10}$	1.000	0 / 0	0 / 1	1
G04	5	26.9217%	9832.45	2.161	0 / 6	0 / 0	2
G05	4	0.0000%	8863.69	1788.74	2 / 0	0 / 3	3
G06	2	0.0072%	1246828.23	1.010	0 / 2	0 / 0	2
G07	10	0.0000%	5928.19	12.671	3 / 5	0 / 0	6
G08	2	0.8751%	1821.61	2.393	0 / 2	0 / 0	0
G09	7	0.5207%	10013016.18	25.05	0 / 4	0 / 0	2
G10	8	0.0008%	27610.89	3842702	3 / 3	0 / 0	6
G11	2	0.0000%	4.99	1.000	0 / 0	0 / 1	1
G12	3	0.04819%	0.72813	1	0 / 1	0 / 0	0
G13	5	0.0000%	$1.91 \cdot 10^{75}$	2.94	0 / 0	0 / 3	3
G14	10	0.0000%	1813.3	1.343	0 / 0	3 / 0	3
G15	3	0.0000%	586.0	1.034	0 / 0	1 / 1	2
G16	5	0.0000%	811263.1	75.73	4 / 34	0 / 0	4
G17	6	0.0000%	42278.85	4.09	0 / 0	0 / 4	4
G18	9	0.0000%	5584.5	4.9	0 / 13	0 / 0	6
G19	15	0.33592%	55659.1	1.95	9 / 5	0 / 0	0
G20	24	0.0000%	28.99	858.19	0 / 6	2 / 12	16
G21	7	0.0000%	1000	23516.64	0 / 1	0 / 5	6
G22	22	0.0000%	2000	$3.1 \cdot 10^9$	0 / 1	8 / 11	19
G23	9	0.0000%	13044.3	82.56	0 / 2	3 / 1	6
G24	2	0.44250%	6.97	1.82	0 / 2	0 / 0	2

3.1. Experimental Setup

We apply the proposed algorithm to all 24 G-problems. The initial population of size $3d$ is created by Latin hypercube sampling (LHS). In order to have statistically sound results, every single run is repeated with 50 different random seeds (different initial populations, different starting points). We define two different pools of models M described in Sec. 2.3. The first pool of models M_1 contains multiquadric basis functions with different width factors w abbreviated as MQ_w. The second model pool M_2 has all the elements of M_1 plus cubic RBF. The algorithms which use $M_1 = \{MQ_{0.01}, MQ_{0.2}, MQ_{0.5}, MQ_1, MQ_5\}$ are denoted as ***MQ*** ensemble and the algorithms using $M_2 = M_1 \cup \{cubic\}$ are called ***MQ-Cubic*** ensemble. In the early phases of this work we have included Gaussian basis functions as well. In this paper we exclude them, mainly because of the frequent crashes occurring with Gaussian RBF in our case.

It is also interesting to investigate the impact of varying the sliding window length L described in Sec. 2.3. Therefore, we present results for different values of $L \in \{1, 30, n\}$. When the sliding window has the minimum possible length $L = 1$, the model selection is done based on the most recent information and the past results do not have any direct impact on the model selection process. $L = n$ means that in each iteration the length of the sliding window is exactly equal to the number of evaluated

design points and the decision is made according to the *accumulated* information from the beginning of the process. In this article, the length of the sliding window is denoted in square brackets behind the type of ensemble. For example MQ-Cubic[1] represents MQ-Cubic ensemble with a sliding window of length $L = 1$.

Furthermore, we conduct some tests with fixed models (cubic and $MQ_{0.2}$) to compare the performance of SACOBRA with and without model selection which is the main purpose of this work. The cubic basis functions used to be the main basis functions in SACOBRA framework originally. Our initial experiments with fixed MQ_w basis function have shown that it is difficult to achieve consistently good optimization results when w is too large or too small. The best results achieved with MQ fixed modeling – for optimizing G10 – appeared to be with $MQ_{0.2}$.

All algorithms reported here use the repair-infeasible [18] and the equality handling [17] extensions to SACOBRA. Other parameters are automatically adjusted to the problem at hand by the self adjusting mechanism inside SACOBRA [1]. We considered a limited budget of not more than 500 true function evaluations for all G-problems. Inside SACOBRA, we use Powell's constrained optimizer COBYLA [21] for optimization on the surrogate models.

3.2. Data profile

In order to compare the overall performance of different optimization algorithms on a set of problems we use data profiles [22]:

$$d_s(\alpha) = \frac{1}{|\mathbb{P}|}|\{p \in \mathbb{P} : \frac{t_{p,s}}{d_p + 1} \leq \alpha\}|, \quad (3)$$

where $\mathbb{P}$ is a set of problems, $\mathbb{S}$ is a set of solvers and $t_{p,s}$ is the number of iterations that solver $s \in \mathbb{S}$ needs to *solve* problem $p \in \mathbb{P}$. d_p is the dimension of problem p.

A COP problem is said to be *solved* if a feasible solution $\vec{x}$ is found whose objective value $f(\vec{x})$ deviates from the best known objective value $f(\vec{x^*})$ less than a given tolerance, $|f(\vec{x^*}) - f(\vec{x})| < \tau$. First we consider a fixed tolerance $\tau = 0.01$ for all problems (see the black dashed horizontal lines in Figs. 1–4). But using the same tolerance for all problems may not always lead to a fair judgment because the G-problems have very diverse types and ranges for their objective functions. So, $\tau = 0.01$ might be a very harsh tolerance for one problem and a very easy tolerance for another. Therefore, we show as well the results when considering a problem-specific tolerance determined as

$$\tau_p = \max(10^{-5}, \bar{e}), \quad (4)$$

where $\bar{e}$ is the averaged optimization error achieved by all solvers (τ_p is shown as dashed red horizontal line in Figs. 1–4).

3.3. Results

Figs. 1–4 show our detailed results in the form of parallel box plots. The y-axes are always

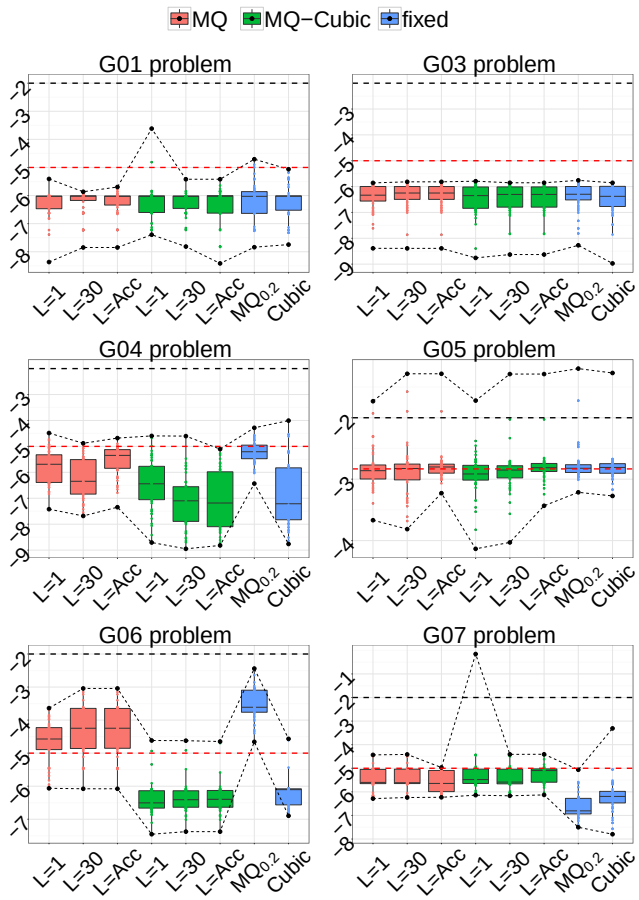

Figure 1. Comparison of final optimization results for G01,G03-G7 problems for 50 independent runs, achieved with and without ensemble and different sliding window sizes.

$\log_{10}$(final optimization error). The blue boxes show results for fixed models (cubic or $MQ_{0.2}$, **no** model selection), the red / green boxes show different model ensembles M_1 / M_2 **with** model selection according to Algorithm 1. The red and green boxes are further differentiated by their sliding window sizes: $L = 1$, $L = 30$, and *Acc* (accumulating, i. e. $L = n$). The last choice *Acc* is quite similar to LOOCV, since the approximation error from all n design points is averaged.

In many cases, $L = 1$ is better than *Acc* (e. g. G08, G09, G13, G21). In most cases the ensemble MQ-Cubic is equal or better than the ensemble MQ alone. For the fixed cases (blue boxes) sometimes cubic is better (G08), sometimes $MQ_{0.2}$ (G09). The ensemble MQ-Cubic, SW[1] is in most cases close to or better than the better choice.

This is further supported when we summarize our results from all 24 G-problems in the data profiles of Figs. 5 and 6: The best COP solvers are the ensembles with online model selection and sliding window length $L = 1$. The worst COP solvers are the ones with fixed models (MQ fixed or cubic fixed). Fig. 6 shows a substantial gap of nearly 15% between

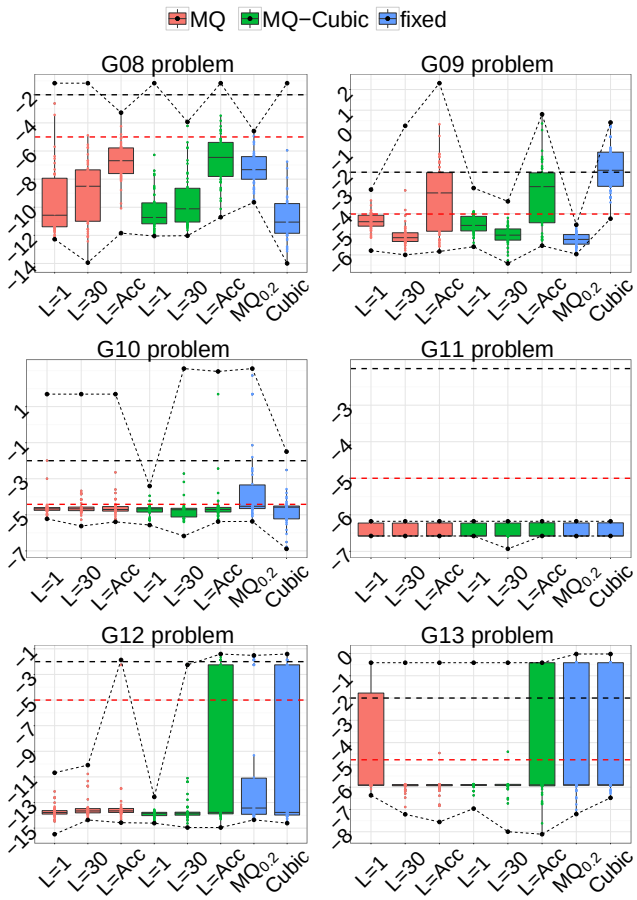

Figure 2. Same as Fig. 1 for problems G08-G13.

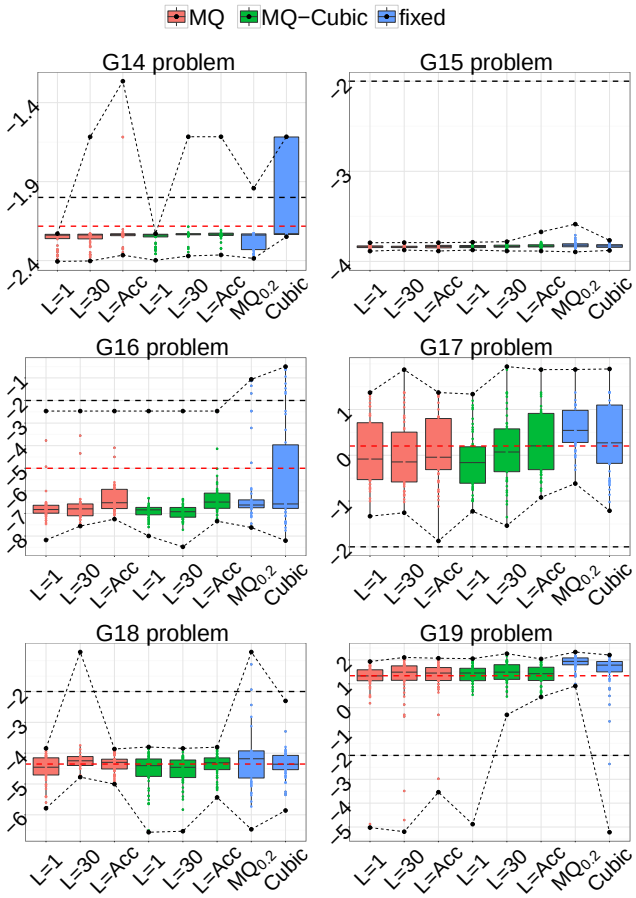

Figure 3. Same as Fig. 1 for problems G14-G19.

the worst and the best solver (the best solver can solve 15% more problems than the worst one).

The main driving force for this improvement is that in the best solver each COP function gets the best-suited surrogate model. Fig. 7 shows exemplarily the approximation errors for problem G13: While the constraint functions g2 and g3 are better approximated by MQ, g1 is better modeled by a cubic surrogate model, especially between iteration 100 and 250. This demonstrates that different surrogates for different COP functions have a clear beneficial effect.

Fig. 8 depicts the frequency of model selection with Algorithm 1 for the same problem G13. It shows nicely that the cubic RBF type is dominantly selected for g1, while MQ_5 is dominantly selected for g2 and g3 in the early iterations. The objective function has a similar approximation error for each surrogate model in Fig. 7. Consequently, there is no clear winner here. Nevertheless, the ability of the surrogate ensemble to switch between different RBF types might lead to better exploration of the search space (see Sec. 4).

Table 3 shows the results for all 24 G-problems as found by our algorithm and by several other authors who cover

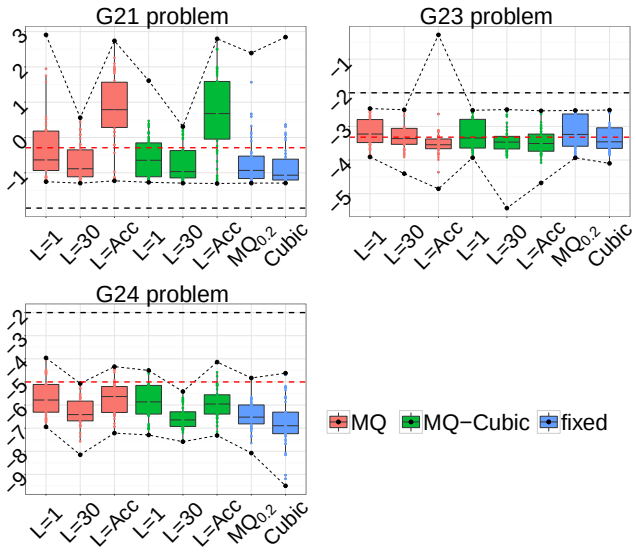

Figure 4. Same as Fig. 1 for problems G21, G23, G24.

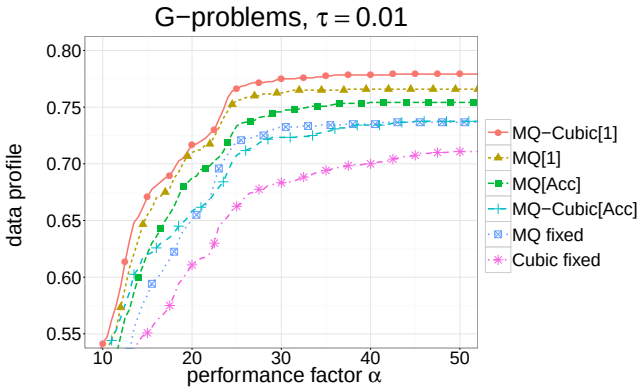

Figure 5. Data profile for problems G01-G24 with a constant tolerance value of 0.01 for all problems. Ensemble MQ-Cubic[1] with $L = 1$ is outperforming the former algorithm which used cubic as the fixed basis function.

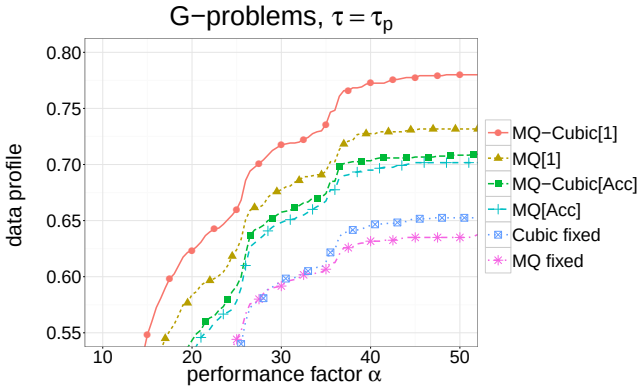

Figure 6. Data profile for problems G01-G24. This data profile uses the stricter problem-specific tolerance τ_p according to Eq. (4). Ensemble MQ-Cubic[1] with $L = 1$ is outperforming all other approaches.

the whole G-benchmark suite [23], [24], [25].[1] SACOBRA finds good solutions (small error in relation to the objective function range FR, see Table 2) for all G-problems except G02, G20 and G22. Problem G02 has a highly multimodal fitness function in 20-dimensional space, which is very likely not accessible for *any* surrogate-based optimization. The remaining problems G20 and G22 are known as very challenging COPs in the literature [23], [24], [25]. G20 and G22 are both high-dimensional problems ($d > 20$) with many equality constraints ($r > 10$). No feasible solution is known so-far for problem G20. Although the feasible optimum of G22 is known, most solvers – including SACOBRA – have difficulties to approach this optimum.

1. Some numbers in *red* in Table 3 are reportedly better than the true optimum. The reason for this is that some algorithms transform equality constraints into a small feasible tube with fixed margin $\epsilon = 0.0001$ around the true equality surface. Then a selected solution will usually deviate by a small amount from this equality surface in the direction of better objective values. If this happens at or near the true solution, the selected solution will have a better objective value than the optimum.

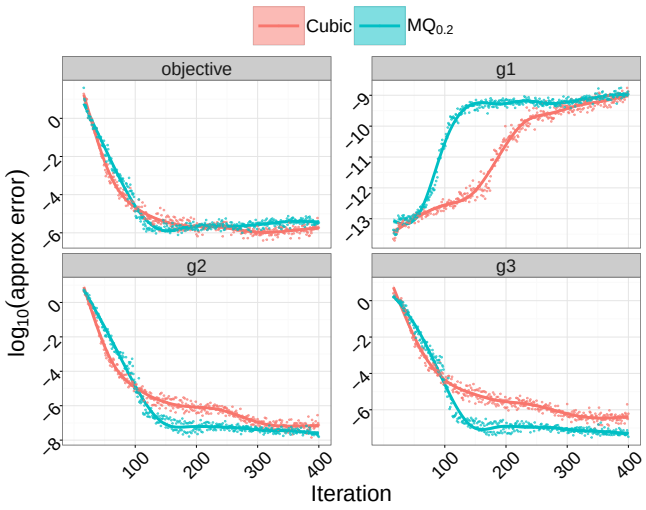

Figure 7. Approximation error as a function of iterations for problem G13. Each point is the median approximation error of 50 independent runs and each curve is a smoother fitted through these points.

4. Discussion

The results show that online model selection performs significantly better on the G-problem suite than any fixed choice of RBF type. We believe that the improvements are due to the ability of (a) having different RBF types for different COP functions, and (b) switching between different RBF types while iterating. The latter may facilitate exploration.

Sliding window: It is at first surprising, that a sliding window of size $L = 1$ turns out to be the best choice among all sliding windows. (We note in passing, that we tested several other values between $L = 1$ and $L = 30$ as well. We found steady improvement towards smaller L.) Why is this the case? We argue that two factors play a role:
(a) If the solver approaches the optimum, it is of utmost importance that the surrogate model is good in the vicinity of the last point(s) visited. The vicinity is emphasized by the (L=1)-approximation error, while a LOOCV procedure would measure the overall approximation accuracy. This may be the reason why earlier work on model selection for optimization with LOOCV [12] did not find significant improvement.
(b) Based on the frequency of model selection in Fig. 8, we made a deeper analysis of individual runs: In cases where several RBF types have roughly the same frequency, we found that the solver with L=1 switches in nearly every iteration between these several RBF types. The fact that such a switching behavior leads to better optimization results indicates that rapid switching probably leads to better exploration of the search space.

Computational costs: The extra computational costs for online model selection are quite low. The algorithm is embedded in the SACOBRA algorithm in such a way that no extra true function evaluations as compared to the

fixed model variants are necessary. (We assume that the true function evaluations are the really expensive part, e. g. as results of long-running computer simulations.) Secondly, the number of extra surrogate model builds is relatively modest: We have to build K models instead of 1 model, where K is the size of the model pool M in Algorithm 1. If we had to perform LOOCV, this number would be considerably larger, namely $K \cdot n$, where n is the number of design points which becomes as large as 500 in our experiments.

Large benchmark: Compared to our previous work [1] with results for G01–G11, we have now tested SACOBRA on all 24 problems G01–G24 of the G-function benchmark suite. This does now include both inequality and equality COPs. SACOBRA has proven to be successful on most of the problems, with the exception of G02, G20 and G22. A close look to Table 3 reveals that the results for G19 are somewhat mediocre as well, showing a median error of 28. But this error has to be set in context to the range FR of the objective function, which is above 50000 in the case of G19 (see Table 2).[2] All other problems are solved with good accuracy in only a small fraction of function evaluations as compared to other work [23], [24], [25].

5. Conclusion

We have proposed a new online model selection algorithm for surrogate models used in COP solving. The algorithm allows different COP functions to be modeled by different types of RBF surrogate models, including the possibility of online switching. We found clear evidence that online model selection is very beneficial on some problems (G12, G13). In these cases, different COP functions took advantage from selecting among different RBF types, which answers positively our research question **(H1)**.

Overall, online model selection proved to be more robust than fixed surrogate model choices, as our data profiles for the 24 G-problems show. Our positive answer to **(H2)** is: The best online model selection has a 13% higher performance than the best fixed-model variant (cubic). We can solve all but three (G02, G20, G22) of the 24 G-problems in less than 500 true function evaluations.

Possible future work could be (a) in the direction of extending the set of possible RBF model candidates and (b) a deeper investigation of the problems which are challenging for our algorithm so far (G02, G19, G20, G22).

Acknowledgments

This work has been supported by the Bundesministerium für Wirtschaft (BMWi) under the ZIM grant MONREP (AiF FKZ KF3145102, Zentrales Innovationsprogramm Mittelstand).

Supported by:
Federal Ministry for Economic Affairs and Energy
on the basis of a decision by the German Bundestag

2. We note in passing that COBYLA alone – without surrogate models – can solve G19 quite rapidly up to an accuracy of 10^{-4} in less than 9000 iterations.

TABLE 3. DIFFERENT OPTIMIZERS: MEDIAN (M) OF BEST FEASIBLE RESULTS AND (FE) AVERAGE NUMBER OF FUNCTION EVALUATIONS. RESULTS FROM 50 INDEPENDENT RUNS WITH DIFFERENT RANDOM NUMBER SEEDS. NUMBERS IN **BOLDFACE**: DISTANCE TO THE OPTIMUM ≤ 0.001. NUMBERS IN *italic (red)*: REPORTEDLY BETTER THAN THE TRUE OPTIMUM. NUMBERS IN (BRACKETS): SOLUTION VIOLATES SOME CONSTRAINTS.

Fct	Optimum		SACOBRA MQ-Cubic $L=1$	Takahama [24]	Jiao [25]	Wei [23]
G01	-15.0	m	**-15.0**	**-15.0**	**-15.0**	**-15.0**
		fe	100	59308	31710	34812
G02	-0.8036	m	-0.3466	**-0.8036**	**-0.8036**	**-0.8036**
		fe	500	149825	79278	500000
G03	-1.0	m	**-1.0**	**-1.0**	***-1.0005***	***-1.0005***
		fe	100	89407	19534	72377
G04	-30665.539	m	**-30665.539**	**-30665.539**	**-30665.539**	**-30665.539**
		fe	200	26216	5357	19390
G05	5126.497	m	**5126.497**	**5126.497**	**5126.497**	**5126.497**
		fe	200	97431	1558	1817
G06	-6961.814	m	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
		fe	100	7381	732	8798
G07	24.306	m	**24.306**	**24.306**	**24.306**	**24.306**
		fe	200	74303	137592	44859
G08	-0.0958	m	**-0.0958**	**-0.0958**	**-0.0958**	**-0.0958**
		fe	500	1139	541	1616
G09	680.630	m	**680.630**	**680.630**	**680.630**	**680.630**
		fe	500	23121	9357	19852
G10	7046.248	m	**7046.248**	**7046.248**	**7046.248**	**7046.248**
		fe	500	105234	85623	196974
G11	0.750	m	**0.750**	**0.750**	*0.7499*	*0.7499*
		fe	100	16420	135	280
G12	-1.0	m	**-1.0**	**-1.0**	**-1.0**	**-1.0**
		fe	400	4124	212	2934
G13	0.0539	m	**0.0539**	**0.0539**	**0.0539**	**0.0539**
		fe	400	34738	3103	734
G14	-47.765	m	-47.759	**-47.765**	**-47.765**	**-47.765**
		fe	400	113439	6093	893
G15	961.715	m	**961.715**	**961.715**	**961.715**	**961.715**
		fe	400	84216	757	641
G16	-1.905	m	**-1.905**	**-1.905**	**-1.905**	**-1.905**
		fe	500	12986	8982	11780
G17	8853.534	m	8854.230	**8853.534**	**8853.534**	**8853.534**
		fe	500	98861	3203	1448
G18	-0.8666	m	-0.8658	**-0.8666**	**-0.8666**	**-0.8660**
		fe	300	59153	3353	55715
G19	32.655	m	60.985	**32.655**	**32.655**	**32.655**
		fe	500	356350	56681	84624
G20	(0.0721)	m	(0.2794)	(0.0721)	–	(0.2050)
		fe	500	500000	–	500000
G21	193.724	m	193.950	**193.724**	**193.724**	**193.724**
		fe	500	135143	46722	2690
G22	236.431	m	(544.843)	248.763	–	(9819.251)
		fe	500	500000	–	500000
G23	-400.002	m	**-400.002**	**-400.002**	*-400.055*	*-400.055*
		fe	500	200765	9757	784
G24	-5.5080	m	**-5.5080**	**-5.5080**	**-5.5080**	**-5.5080**
		fe	500	2952	161	3850
	average	fe	370	111529.5	23201.8	86119.6

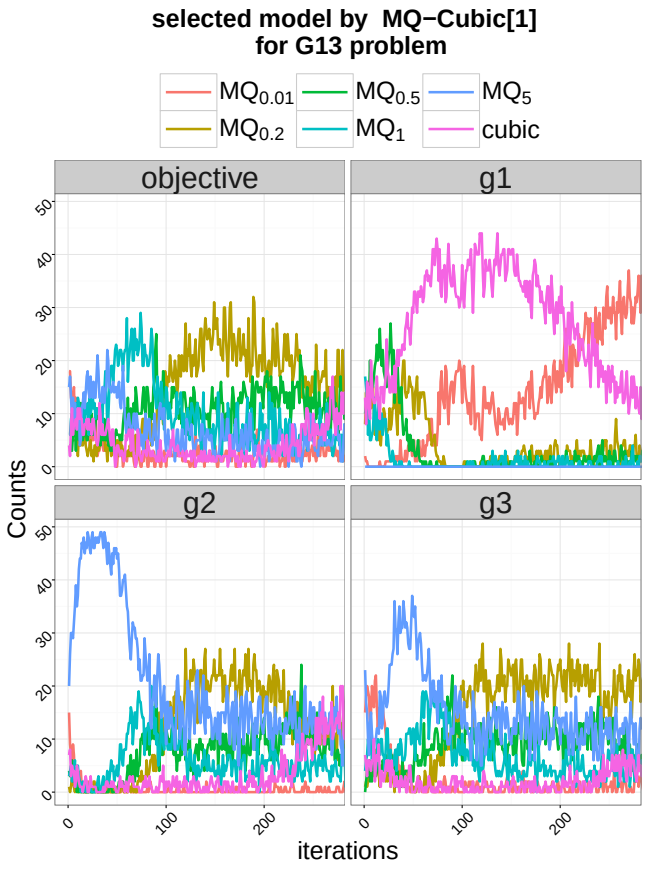

Figure 8. Frequency of model selection (out of 50 runs) in each iteration. It is clearly seen that objective function and three constraint functions call for different model selection patterns.

References

[1] S. Bagheri, W. Konen, M. Emmerich, and T. Bäck, "Solving the G-problems in less than 500 iterations: Improved efficient constrained optimization by surrogate modeling and adaptive parameter control," *arXiv preprint arXiv:1512.09251*, 2015.

[2] R. G. Regis and C. A. Shoemaker, "A quasi-multistart framework for global optimization of expensive functions using response surface models." *J. Global Optimization*, vol. 56, no. 4, pp. 1719–1753, 2013.

[3] P. Koch, S. Bagheri, C. Foussette, P. Krause, T. Bäck, and W. Konen, "Constrained optimization with a limited number of function evaluations," in *Proc. of the 24th Workshop on CI*, F. Hoffmann and E. Hüllermeier, Eds. Universitätsverlag Karlsruhe, 2014, pp. 119–134.

[4] M. Schonlau, W. J. Welch, and D. R. Jones, "Global versus local search in constrained optimization of computer models," in *New developments and applications in experimental design*, N. Flournoy, W. F. Rosenberger, and W. K. Wong, Eds. Hayward, CA: Institute of Mathematical Statistics, 1998, vol. 34, pp. 11–25.

[5] D. Villanueva, R. L. Riche, G. Picard, and R. T. Haftka, "Surrogate-based agents for constrained optimization," in *In 14th AIAA Non-Deterministic Approaches Conference*, 2012.

[6] C. C. Tutum, K. Deb, and I. Baran, "Constrained efficient global optimization for pultrusion process," *Materials and Manufacturing Processes*, vol. 30, no. 4, pp. 538–551, 2015. [Online]. Available: http://dx.doi.org/10.1080/10426914.2014.994752

[7] H. Rocha, "On the selection of the most adequate radial basis function," *Applied Mathematical Modelling*, vol. 33, no. 3, pp. 1573 – 1583, 2009.

[8] D. Gorissen, T. Dhaene, and F. D. Turck, "Evolutionary model type selection for global surrogate modeling," *J. Mach. Learn. Res.*, vol. 10, pp. 2039–2078, Dec. 2009.

[9] A. Mehmani, S. Chowdhury, and A. Messac, "A novel approach to simultaneous selection of surrogate models, constitutive kernels, and hyper-parameter values," *10th AIAA Multidisciplinary Design Optimization Conference*, 2014.

[10] R. V. Field, "A decision-theoretic method for surrogate model selection," *Journal of Sound and Vibration*, vol. 311, no. 35, pp. 1371 – 1390, 2008.

[11] I. Couckuyt, F. D. Turck, T. Dhaene, and D. Gorissen, "Automatic surrogate model type selection during the optimization of expensive black-box problems," in *Proceedings of the 2011 Winter Simulation Conference (WSC)*, Dec 2011, pp. 4269–4279.

[12] J. Müller and C. A. Shoemaker, "Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems," *Journal of Global Optimization*, vol. 60, no. 2, pp. 123–144, 2014.

[13] M. Friese, M. Zaefferer, T. Bartz-Beielstein, O. Flasch, P. Koch, W. Konen, and B. Naujoks, "Ensemble based optimization and tuning algorithms," *Schriftenreihe des Instituts für Angewandte Informatik, Automatisierungstechnik am Karlsruher Institut für Technologie*, p. 119, 2011.

[14] K. S. Bhattacharjee, H. K. Singh, and T. Ray, "Multi-objective optimization with multiple spatially distributed surrogates," *Journal of Mechanical Design*, vol. 138, no. 9, p. 091401, 2016.

[15] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," *Journal of Geophysical Research*, vol. 76, no. 8, pp. 1905–1915, 1971.

[16] B. Fornberg and N. Flyer, "Accuracy of radial basis function interpolation and derivative approximations on 1-d infinite grids," *Advances in Computational Mathematics*, vol. 23, no. 1, pp. 5–20, 2005.

[17] S. Bagheri, W. Konen, and T. Bäck, "Equality constraint handling for surrogate-assisted constrained optimization," in *WCCI'2016, Vancouver*, K. C. Tan, Ed. IEEE, 2016, p. 1. [Online]. Available: http://www.gm.fh-koeln.de/~konen/Publikationen/Bagh16-WCCI.pdf

[18] P. Koch, S. Bagheri, W. Konen, C. Foussette, P. Krause, and T. Bäck, "A new repair method for constrained optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference 2015 (GECCO)*, J. L. Jiménez-Laredo, Ed. ACM, 2015, pp. 273–280.

[19] R. G. Regis, "Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points," *Engineering Optimization*, vol. 46, no. 2, pp. 218–243, 2013.

[20] J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. C. Coello, and K. Deb, "Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization," *Journal of Applied Mechanics*, vol. 41, p. 8, 2006.

[21] M. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances In Optimization And Numerical Analysis*. Springer, 1994, pp. 51–67.

[22] J. J. Moré and S. M. Wild, "Benchmarking derivative-free optimization algorithms," *SIAM J. Optimization*, vol. 20, no. 1, pp. 172–191, 2009.

[23] W. Wei, J. Wang, and M. Tao, "Constrained differential evolution with multiobjective sorting mutation operators for constrained optimization," *Applied Soft Computing*, vol. 33, pp. 207 – 222, 2015.

[24] T. Takahama and S. Sakai, "Constrained optimization by the ϵ constrained differential evolution with gradient-based mutation and feasible elites," in *2006 IEEE International Conference on Evolutionary Computation*, July 2006, pp. 1–8.

[25] L. Jiao, L. Li, R. Shang, F. Liu, and R. Stolkin, "A novel selection evolutionary strategy for constrained optimization," *Information Sciences*, vol. 239, pp. 122 – 141, 2013.