Weighted Ensembles in Model-based Global Optimization

Martina Friese[1,a)], Thomas Bartz-Beielstein[1], Thomas Bäck[2], Boris Naujoks[1] and Michael Emmerich[2]

[1]*SPOTSeven Lab, TH Köln, Steinmüllerallee 1, 51643 Gummersbach, Germany*
[2]*LIACS, Leiden University, Niels Bohrweg 1, 2333CA Leiden , Netherlands*

[a)]Corresponding author: martina.friese@th-koeln.de

Abstract. It is a common technique in global optimization with expensive black-box functions, to learn a regression model (or surrogate-model) of the response function from past evaluations and to use this model to decide on the location of future evaluations. In surrogate model assisted optimization it can be difficult to select the right modeling technique. Without preliminary knowledge about the function it might be beneficial if the algorithm trains as many different surrogate models as possible and selects the model with the smallest training error. This is known as model selection. Recently a generalization of this approach was proposed: instead of selecting a single model we propose to use optimal convex combinations of model predictions. This approach, called model mixtures, is adopted and evaluated in the context of sequential parameter optimization. Besides discussing the general strategy, the optimal frequency of learning the convex weights is investigated. The feasibility of this approach is examined and its benefits are compared to simpler methods.

INTRODUCTION

Sequential parameter optimization (SPO) is a well known approach for optimization tasks with expensive function evaluations [1, 2, 3]. It combines a sequential design approach and tools for reducing costly function evaluations on the original model by replacing them partly with fast approximate evaluations on surrogate models. As a *surrogate model*, we understand a function $\hat{y} : \mathbb{R}^d \rightarrow \mathbb{R}$ that is an approximation to the objective function $y : \mathbb{R}^d \rightarrow \mathbb{R}$, learned from a finite set of evaluations of the objective function. SPO packages, such as the SPO Toolbox (SPOT), come with a large variety of surrogate models (base models) from which the user can choose. The choice of the surrogate model can have a large influence on the solution quality and performance of the optimizer. To decide which surrogate model to select for a given problem, often expert knowledge is needed. But if there is no preliminary knowledge about the objective function, it would be beneficial if the algorithm could learn all by itself which surrogate model type suits the problem best. For instance, this can be done by evaluating different models on training data a priori and using a statistical model selection approach to select the most promising surrogate model.

Recent results show that instead of selecting just one surrogate model, it can be beneficial to choose several ones and linearly combine their predictions [4, 5]. Further constraints on the coefficients lead to convex combinations, which can be defined as follows. Given s predictions of different surrogate models $\hat{y}_i : \mathbb{R}^d \rightarrow \mathbb{R}, i = 1, \ldots, s$ and d the dimension of the search space, by a *convex combination of models* (CCM) we understand a surrogate model given by $\sum_{i=1}^{s} \alpha_i \hat{y}_i$ with $\sum \alpha_i = 1$ and $\alpha_i \geq 0, i = 1, \ldots, s$. As demonstrated in [4], outliers or deviations in opposing directions can be compensated this way resulting in an overall more robust performance. Compared to more complex surrogate models, CCM have the advantage of increased transparency and decreased over-fitting risk. Moreover, they can be used for ensembles of heterogeneous surrogate models, because they only combine the predictions of each surrogate model.

Previous studies showed that CCM can indeed improve the surrogate model quality as compared to mere surrogate model selection [4]. Based on these results, it seems promising to integrate optimized CCM into SPO. This paper can be seen as a first result into this direction. We address the following two questions:

1. Can optimized (and dynamically adapted) CCM compete with fixed, base models in SPO?
2. How does the CCM approach compete with approaches that dynamically update the surrogate model selection?

ALGORITHM

The main steps of our proposed CCM method are presented in Algorithm 1 which, up to details, are self-explanatory.

```
Algorithm 1: CCM Ensemble-building using weighted 10-fold cross-validation
   Data: Previously evaluated $n$ data-points $(x_1, y(x_1)), \ldots, (x_n, y(x_n))$;
   Result: CCM function $\hat{y} : \mathbb{R}^d \to \mathbb{R}$
1  begin
2      For all base models, compute weighted cross-validation on previously evaluated $n$ data-points;
3      Search for best weights using box-constrained optimization and data generated in step 2;
4      Generate ensemble-predictor function $\hat{y}$ using best weights;
```

During the progress of a SPO the data set is continuously increased. Therefore, it might be beneficial to update the weight combination over time. The proposed method updates the ensemble combination in every τ-th step of the optimization. The values $\tau = 5$ and $\tau = \infty$ (only initial computation of weights) were tested.
To evaluate the fit of the base models, 10-fold cross-validations are carried out on the previously evaluated n data-points (cf. Alg.1 line 2). It is expected that during the progress of a SPO evaluated points will cluster at local optima. Without taking this into account during cross-validation we risk to put too much emphasis on prediction errors close to those local optima, which leads to over-fitting in these areas. To overcome this we use density weighted cross-validation, where the Root Mean Square Error (RMSE), which is used as quality indicator, is calculated as follows: $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\beta_i(y_i - \hat{y}_i))^2}$. The weights, $\beta_i \in [0,1]$, $i = 1,\ldots,n$, depend on the density of the point's direct neighborhood. For the calculation of this density the k closest neighbors are used, where k=20 was used in the tests. The density of a point $j, j = 1,\ldots,n$ is then calculated as the median distance to these neighbors: $dens_j = Median\left\{\sqrt{\sum_{i=1}^{d}(x_j^i - x_k^i)^2} \mid k = 1,\ldots,20\right\}$. Density values exceeding the overall mean density are truncated to the mean value. This is done to ensure that all points with a neighborhood of mean density, or sparser, get full weights in the cross-validation. To normalize weights to the $[0,1]$ range, we compute $\beta_j = dens_j / max\{dens_0,\ldots,dens_n\}$.

To find the best ensemble weights, in Alg.1 line 3, we use a (1+1)-evolution strategy (ES) with 1/5th success probability rule for step size adaptation [6]. This choice is motivated by the good results obtained in [4], but other choices might also be suitable. The search favors a combination of models over a single model, only if its overall fit in terms of RMSE is strictly better. Due to the incremental data update, the position of the optimal weight combination in the current iteration is likely to not deviate much from the one obtained in the previous optimization. Therefore, the previously obtained solution is used as a first offspring solution. The first parent solution is the best base model obtained in step 2. We assume that the benefit of a model with a contribution to the ensemble of less than two percent is negligible. Thus the mutation step of the ES is adopted such that single weights are at least two percent or zero.

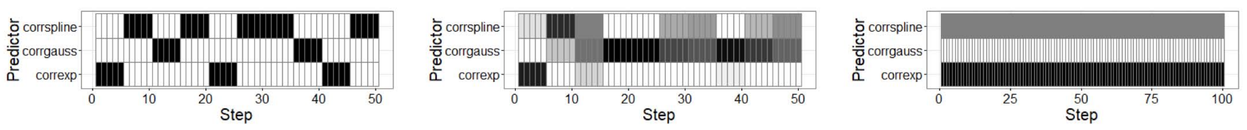
EXPERIMENTAL SETUP

To gain interpretable results a straightforward SPO framework is implemented for the experiments. This framework allows the use of a predefined initial design of experiments to ensure that all experiments use the same initial configuration. In every subsequent step, two points are chosen to satisfy the demand for exploration and exploitation. Both points are selected after pre-evaluation of a large Latin Hypercube Design (LHD) on the model. For exploitation a local search is run on the model, starting from the 20 points of the LHD with the best function values. Parameters violating the constraints are set to the allowed bounds. The best point found during this procedure is chosen to be evaluated on the objective function. For exploration, from the 20 points of the LHD with the farthest nearest neighbor, the point with the best function value is chosen to be evaluated.

The methods were tested on two Gaussian Landscape Generator functions (GLG4D, GLG8D) and two real simulators (piston function, robot arm function). The GLG 4D and GLG 8D functions [7] can be described as follows: A Max-Set of Gaussian Landscape Generators that computes the upper envelope of m weighted Gaussian process realizations and can be used to generate continuous, box-constrained optimization problems. Experiments start from 20 predefined designs. Each design contains 60 points for the 4D function and 100 points for the 8D function. During optimization on the 4D problem 100 additional points are to be evaluated, 220 points on the 8D function. The Piston

TABLE 1. Mean values (standard deviations are shown in brackets) and the corresponding ranks (Mn RankSums). Md RankSums refers to the sums of the ranked median optimization results.

FUN	Choose	CCM (τ = 5)	Initial (τ = inf)	BestBaseModel	BaseModelName
GLG4D	22.2187 (±11.7400)	21.9572 (±13.8244)	24.0334 (±11.8531)	**21.1013** (±13.1714)	corrgauss
GLG8D	39.3837 (±13.3109)	29.9714 (±9.4306)	35.4665 (±15.1034)	**28.7021** (±11.3945)	corrgauss
piston	0.1740 (±0.0070)	**0.1703** (±0.0034)	0.1730 (±0.0050)	0.1708 (±0.0017)	correxp
robot	0.0344 (±0.0206)	**0.0221** (±0.0139)	0.0269 (±0.0190)	0.0309 (±0.0153)	corrgauss
Mn RankSums	15	**6**	12	7	
Md RankSums	12	**5**	13	10	



(a) Choose (b) Ensemble($\tau = 5$) (c) Initial ($\tau = \inf$)

FIGURE 1. Distribution of the weights used during an example run on GLG4D.

function is a seven dimensional problem. It models the circular motion of a piston within a cylinder and involves a chain of nonlinear functions [8]. The Robot Arm function [8] is a nonlinear function that has 8 input dimensions. The response is the distance from the end of the robot arm to a reference point. The experiments on real simulators start from 10 predefined experimental designs, each containing 110 data points. During optimization further 50 data points are to be evaluated.

The test setup is realized using three Kriging models with correlation functions of different types, as before in [4]. These are Kriging with exponential correlation function ('correxp'), gaussian correlation function ('corrgauss'), and spline correlation function ('corrspline') as described in [9].
To investigate the performance of the proposed dynamically adapted CCM method (τ = 5), which will be referred to as 'WeightedEnsemble', its performance is compared to the performance of the base models as well as to two ensemble-like approaches. These approaches will hereafter be referred to as 'Initial' and 'Choose'. 'Initial' builds the CCM only in the first iteration of SPO ($\tau = \infty$). Using this method we want to get some insights into the benefits of adjusting the ensemble during the optimization process. 'Choose' selects a single base model in every fifth iteration by 10-fold cross-validation. Thereby we assess the benefits of using mixtures of models instead of selecting and updating only single base models.

RESULTS

Table 1 shows optimization results with mean standard deviation per model and function. The ranking of these values is summed up in the 'Md RankSums' row. The 'Mn RankSums' row depicts the sums of the ranked mean optimization results. The column 'BestBaseModel' displays the result of the best model only, the name of this best performing base model is given in the last column.

The best ranked method on this benchmark is the proposed dynamically adapted CCM, followed closely by 'BestBaseModel'. The good performance of the best base model corrgauss on GLG4D and GLG8D is to be expected since Gaussian processes are used for the generation of this test function. However, in a black-box setting the knowledge of the best base model is not available. Moreover, despite this accordance the dynamically adapted CCM shows a nearly as good performance as the winning base model while 'Choose' and 'Initial' clearly fall behind.

The functionality of the different ensemble approaches is shown in Fig. 1. Each of the three plots displays the weights used in every step of the optimization. Initial (1c) is restricted to specifying its weights in the initial step only. Choose (1a) and Ensemble (1b) update their weights in every fifth step of the optimization process.

DISCUSSION AND RELATED WORK

Three new strategies for an automated model handling in the SPO framework were proposed: ('Choose') periodically selecting the best model, based on the cross-validation error, ('Initial') selecting the best convex linear combination of models based on an initial design, and ('dynamically adapted CCM') periodically selecting the best convex linear combination of models. First experimental results on a small set of nonlinear problems of moderate dimensionality are promising. The dynamically adapted CCM model indeed uses more than only one model to achieve an increased performance (see Fig. 1). Even compared to the a priori unknown best base model a moderate improvement is achieved.

Future work will have to extend the comparison with related ensemble-based methods. So far only few work is done on adaptations of the model or the choice of the model during the sequential step (Bauer et al. [10], Hess [11], [12]). Other common approaches of ensemble-based learning are to choose an averaging approach or a weighting based on model variances, given these are available, for combining models. The convex model combinations presented in this paper can be viewed as an elegant stacking approach and as such they are similar to 'ensembles of surrogates', which however use a fixed rule for determining weights. Dynamical model selection can also be achieved by reinforcement learning [13], which would be an alternative for single model selection by optimization ('Choose'). Furthermore, it will be interesting to extend the set of test problems and include more types of base models. Moreover, the optimal setting of control parameters should be studied, such as the optimal frequency of the CCM updates.

ACKNOWLEDGMENTS

This work is partially funded by the European Commission's H2020 programme, through the UTOPIAE Marie Curie Innovative Training Network, H2020-MSCA-ITN-2016, Grant Agreement No 722734 as well as through the SYNERGY twinning project, research and innovation program under Grant Agreement No 692286.

REFERENCES

[1] A. Žilinskas and J. Mockus. On one bayesian method of search of the minimum. *Avtomatika i Vychislitelnaya Teknika*, 4:42–44, 1972.

[2] T. Bartz-Beielstein, C. Lasarczyk, and M. Preuss. Sequential Parameter Optimization. In B McKay et al., editors, *Proceedings 2005 Congress on Evolutionary Computation (CEC'05), Edinburgh, Scotland*, pages 773–780, IEEE Press, Piscataway NJ, 2005.

[3] F. Hutter, H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization*, pages 507–523. Springer, 2011.

[4] M. Friese, T. Bartz-Beielstein, and M. Emmerich. Building ensembles of surrogates by optimal convex combination. In Gregor Papa and Marjan Mernik, editors, *Bioinspired Optimization Methods and their Applications*, pages 131–143, 2016.

[5] T. Bartz-Beielstein and M. Zaefferer. Model-based methods for continuous and discrete global optimization. *Applied Soft Computing*, 55:154 – 167, 2017.

[6] H.-P. Schwefel. *Evolution and optimum seeking*. Wiley, New York, 1995.

[7] M. Gallagher and B. Yuan. A general-purpose tunable landscape generator. *IEEE Trans. Evolutionary Computation*, 10(5):590–603, 2006.

[8] S. Surjanoviv and D. Bingham. Virtual library of simulation experiments. [ONLINE], 2016. Available at: http://www.sfu.ca/ ssurjano/. [Accessed 30 November 2016].

[9] J. Søndergaard S. N. Lophaven, H. B. Nielsen. Dace - a matlab kriging toolbox. Technical report, Technical University of Denmark, 2002.

[10] N. Bauer, J. Schiffner, and C. Weihs. Comparison of classical and sequential design of experiments in note onset detection. Technical report, SFB 823, 2011.

[11] S. Hess. Sequentielle modellbasierte Optimierung mit Ensembles durch einen Reinforcement-Learning-Ansatz. Master's thesis, Technische Universität Dortmund, 2012.

[12] T. Goel, R. T. Haftka, W. Shyy, and N. V. Queipo. Ensemble of surrogates. *Structural and Multidisciplinary Optimization*, 33(3):199–216, 2007.

[13] R. Allmendinger, M. Emmerich, J. Hakanen, Y. Jin, and E. Rigoni. Surrogate-assisted multicriteria optimization: Complexities, prospective solutions, and business case. *Journal of Multi-Criteria Decision Analysis*, 24(1-2):5–24, 2017.