Engineering With Computers manuscript No.
(will be inserted by the editor)

Multiobjective global surrogate modeling, dealing with the 5-percent problem

Dirk Gorissen · Ivo Couckuyt · Eric Laermans · Tom Dhaene

the date of receipt and acceptance should be inserted later

Abstract When dealing with computationally expensive simulation codes or process measurement data, surrogate modeling methods are firmly established as facilitators for design space exploration, sensitivity analysis, visualization, prototyping and optimization. Typically the model parameter (=hyperparameter) optimization problem as part of global surrogate modeling is formulated in a single objective way. Models are generated according to a single objective (accuracy). However, this requires an engineer to determine a single accuracy target and measure upfront, which is hard to do if the behavior of the response is unknown. Likewise, the different outputs of a multi-output system are typically modeled separately by independent models. Again, a multiobjective approach would benefit the domain expert by giving information about output correlation and enabling automatic model type selection for each output dynamically. With this paper the authors attempt to increase awareness of the subtleties involved and discuss a number of solutions and applications. In particular we present a multiobjective framework for automatic global surrogate model generation to help tackle both problems and that is applicable in both the static and sequential design (adaptive sampling) case.

Keywords surrogate modeling · metamodeling · error estimation · performance measures · model selection · multiobjective optimization · adaptive sampling · hyperparameter optimization

1 Introduction

Regardless of the rapid advances in High Performance Computing and multi-core architectures, it is rarely feasible to explore a design space using high fidelity computer simulations [94]. As a result, data based surrogate models (otherwise known as metamodels, emulators, or response surface models) have become a standard technique to reduce this computational burden and enable routine tasks such as visualization, design space exploration, prototyping, sensitivity analysis, and of course, optimization [99,88].

It is important to first comment on the difference between local and global surrogate models since motivation and philosophy are distinct. Local surrogate modeling involves building small, relatively low fidelity surrogates for use in optimization. Local surrogates are used as rough approximators of the (costly) optimization surface and guide the optimization algorithm towards good extrema while minimizing the number of simulations [6]. Once the optimum is found, the surrogate is discarded. Many advanced methods for constructing and managing these local surrogates have been designed, including trust region methods [100,2], various ensemble techniques [33], space mapping methods [19], and hierarchical surrogates [104]. In general the theory is referred to as Surrogate Based Optimization (SBO) or Metamodel Assisted Optimization (MAO). A good overview reference is given by [18], [79], and the work by Y. S. Ong [74].

In contrast, with global surrogate modeling the surrogate model itself is usually the goal. The objective is to construct a high fidelity approximation model that is as accurate as possible over the complete design space of interest using as few simulation points as possible. Once constructed, the global surrogate model (also referred to as a replacement metamodel) is reused in other stages of the computational science and engineering pipeline. For example as a cheap

D. Gorissen, I. Couckuyt, E. Laermans, T. Dhaene
Ghent University - IBBT, Department of Information Technology (INTEC), Gaston Crommenlaan 8, 9050 Gent, Belgium
E-mail: {dirk.gorissen, ivo.couckuyt, eric.laermans, tom.dhaene}@ugent.be

accurate and scalable replacement model in standard design software packages (e.g., [11]). Thus optimization is usually not the main goal (though it still can be), but rather a useful post-processing step.

Of course the dichotomy is not strict; ideas and approaches between the two types can, and should, be exchanged, allowing for different hybrids to emerge that borrow ideas from both types. A good example in this respect is the popular Efficient Global Optimization (EGO) approach first described by Jones et. al. in [50] and elaborated by many others (e.g., [84]).

The current paper attempts to increase the value and utility of global surrogate methods for practitioners by exploring a multiobjective approach to surrogate model generation. This gives an engineer or domain expert more flexibility in specifying a priori constraints on the surrogate model generation process (cfr *"The 5 percent problem"* in section 3). In addition, a multiobjective approach allows multi-output problems to be modeled directly, giving more information than modeling each output independently. At the same time we do not assume a fixed sample distribution but select and perform simulations iteratively (adaptive sampling) as would be the case in any real application.

2 Global surrogate modeling

We stress again that the context of this work is to efficiently generate accurate *global* surrogates (valid over the complete design space) using a minimal number of computationally expensive simulations. Optimization of the simulation output is not the main goal, rather we are concerned with optimization of the model parameters (hyperparameter optimization).

Global surrogate models are particularly useful for design space exploration, sensitivity analysis, prototyping, visualization, and *what-if* analysis. They are also widely used to build model libraries for use in controllers or engineering design software packages. In addition, they can cope with varying boundary conditions. This enables them to be chained together in a model cascade in order to approximate large scale systems [5]. A classic example is the full-wave simulation of an electronic circuit board. Electromagnetic modeling of the whole board in one run is almost intractable. Instead the board is modeled as a collection of small, compact, accurate surrogates that represent the different functional components (capacitors, transmission lines, resistors, etc.) on the board. Finally, if optimization is the goal, one could argue that a global model is less useful since significant time savings could be achieved if more effort were directed at finding the optimum rather than modeling regions of poor designs. However, this is the logic of purely local models, but they forgo any wider exploration of radical designs [29]. Some examples of global modeling approaches can be found in [72,31,7,89].

The mathematical formulation of the problem is as follows: approximate an unknown multivariate function $f : \Omega \mapsto \mathbb{C}^n$, defined on some domain $\Omega \subset \mathbb{R}^d$, whose function values $f(X) = \{f(x_1), ..., f(x_k)\} \subset \mathbb{C}^n$ are known at a fixed set of pairwise distinct sample points $X = \{x_1, ..., x_k\} \subset \Omega$. Constructing an approximation requires finding a suitable function s from an approximation space S such that $s : \mathbb{R}^d \mapsto \mathbb{C}^n \in S$ and s closely resembles f as measured by some criterion ξ. The task is then to find the best approximation $s^* \in S$ such that s^* satisfies $\min_{s \in S} \xi = s^*$. This minimization is an optimization problem over the model parameters, commonly referred to as the hyperparameter optimization problem. This may be solved manually, through trial and error, or using readily available optimization algorithms. Additional assumptions are that f is expensive to compute. Thus the number of function evaluations $f(X)$ needs to be minimized and data points must be selected iteratively, at points where the information gain will be the greatest [96]. Mathematically this means defining a sampling function

$$\phi(X_{i-1}) = X_i\,,\, i = 1, .., N \tag{1}$$

that constructs a data hierarchy

$$X_0 \subset X_1 \subset X_2 \subset ... \subset X_N \subset X \tag{2}$$

of nested subsets of $X = \{x_1, ..., x_k\}$, where N is the number of levels. X_0 is referred to as the *initial experimental design* and is constructed using one of the many algorithms available from the theory of Design and Analysis of Computer Experiments (see the work by Kleijnen et al. [55]). Once the initial design X_0 is available it can be used to seed the sampling function ϕ. An important requirement of ϕ is to minimize the number of sample points $|X_i| - |X_{i-1}|$ selected each iteration (f is expensive to compute), yet maximize the information gain of each successive data level. This process is referred to as adaptive sampling [15], but is also known as active learning [16], reflective exploration [11], Optimal Experimental Design [80] and sequential design [53]. The advantage of adaptive sampling is that the number of required data points need not be specified up front, avoiding potential over- or undersampling. At the same time, by intelligently choosing the location of each data point the accuracy of the surrogate may be maintained. An important consequence of the adaptive sampling procedure is that the task of finding the best approximation s^* becomes a dynamic problem instead of a static one. Since the optimal model parameters will change as the amount and distribution of data points changes.

3 *"The 5 percent problem"*

The basic algorithm for generating a global surrogate model through adaptive sampling is as follows: a small number of simulations are performed according to some Design of Experiment plan. Given these sample points the space of candidate models S is searched for the besting fitting model s^* according to ξ. If s^* is acceptable (i.e., the model meets the target requirement set out by the user) the algorithm terminates. Else the sampling function ϕ is used to generate a new set of sampling points (adaptive sampling) and the model search is resumed. This whole process continues until the user-defined accuracy has been reached or the computational budget is exhausted.

A crucial aspect of this algorithm is identifying a suitable criterion ξ, where ξ constitutes three parts:

$$\xi = (\Lambda, \varepsilon, \tau) \tag{3}$$

with Λ the generalization estimator, ε the error (or loss) function used, and τ the target value required by the user. This means that the global surrogate model generation problem (namely finding s^*) for a given set of data $D = (X_i, f(X_i))$ can be formally defined as

$$s^* = \arg\min_{t \in T} \arg\min_{\theta \in \Theta} \Lambda(\varepsilon, s_{t,\theta}, D) \tag{4}$$

such that

$$\Lambda(\varepsilon, s^*_{t,\theta}, D) \leqslant \tau \tag{5}$$

where $s_{t,\theta}$ is the parametrization θ (from a parameter space Θ) of s and $s_{t,\theta}$ is of model type t (from a set of model types T).

The outer minimization over $t \in T$ is the task of selecting a suitable approximation model type, i.e., a rational function, a neural network, a spline, etc. This is the model type selection problem. In practice, one typically considers only a single $t \in T$, though others may be included for comparison. Then given a particular approximation type t, the task is to find the hyperparameter assignment θ that minimizes the generalization estimator Λ (e.g., determine the optimal order of a polynomial model). This is the hyperparameter optimization problem, though generally both minimizations are simply referred to as the model selection problem.

Many implementations of Λ have been described: the hold-out, bootstrap, cross validation, jack-knife, Akaike Information Criterion (AIC), etc. In the simple case where Λ is just the in-sample error, the problem simplifies to

$$s^* = \min_{t \in T} \min_{\theta \in \Theta} \varepsilon(s_{t,\theta}(X_i), f(X_i)) \tag{6}$$

such that

$$\varepsilon(s^*_{t,\theta^*}(X_i), f(X_i)) \leqslant \tau \tag{7}$$

The crucial problem is identifying suitable implementations for Λ, ε and a target value for τ. The three are closely linked but only the Λ-selection problem has been extensively studied theoretically [52,101] and empirically [90, 68]. The selection of ε and τ is less appreciated and often overlooked, but equally important [60,25]. Selecting an error function ε and required target accuracy τ is difficult since it requires knowledge of the structure of the response and a full understanding of what the generalization estimator Λ actually measures. Failure to do so leads to misinterpretation, inappropriate application, ultimately resulting in a trial and error model generating procedure.

This brings us to, what we have termed, *“The 5 percent problem”*. The phrase stems from an application where an engineer needed a replacement metamodel. When asked what model accuracy was required the answer was simply 5 percent. While this may seem like a straightforward, objective requirement, enforcing it in practice turned out to be difficult. The reason is twofold and is detailed below.

3.1 Choice of error function

First, there is the choice of the error function ε. Roughly speaking there are two categories of error functions: absolute and relative.

3.1.1 Absolute errors

Absolute errors (e.g., Average Absolute Error (AAE), Mean Squared Error (MSE), etc.) are often undesirable in an application context since they are not unit-free and depend on the specific prediction value of the response. On the other hand they are very popular in machine learning settings but not always rightly used. A good example is the Root Mean Square Error (RMSE), by far the most popular error function:

$$RMSE(y, \tilde{y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2} \tag{8}$$

Where $y_i, \tilde{y}_i$ are the real and predicted response values respectively. The main advantage of the RMSE is that it is the best finite-sample approximation of the standard error $\sqrt{E[y-\tilde{y}]}$ and standard deviation (in the case of an unbiased model) [60]. However, its use is not recommended since it penalizes large errors too severely while virtually ignoring small errors. Such an error function is called pessimistic. Also it is unintuitive to interpret. The RMSE is often interpreted as the true arithmetic average euclidean distance between the prediction $\tilde{y}$ and the true value y. This is however not the case, the RMSE is really one $\sqrt{n}$-th of this value and thus has no simple geometrical interpretation whatsoever. A better solution would be to use the Average Euclidean Error (AEE) as proposed by [60]

$$AEE(y, \tilde{y}) = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(y_i - \tilde{y}_i)^2} \tag{9}$$

However, while the AEE enjoys many desirable properties, it still suffers from outliers (i.e., it is also pessimistic, though less than the RMSE). In cases where this is a problem alternative functions like the Geometric Average Error (GAE) and the Harmonic Average Error (HAE) [60] can be more useful.

$$GAE(y,\tilde{y}) = \left(\prod_{i=1}^{n} \sqrt{(y_i - \tilde{y}_i)^2}\right)^{\frac{1}{n}} \tag{10}$$

$$\begin{aligned} HAE(y,\tilde{y}) &= \left(\frac{1}{n}\sum_{i=1}^{n} \frac{1}{\sqrt{(y_i - \tilde{y}_i)^2}}\right)^{-1} \\ &= \frac{n}{\frac{1}{\sqrt{(y_1-\tilde{y_1})^2}} + \ldots + \frac{1}{\sqrt{(y_n-\tilde{y_n})^2}}} \end{aligned} \tag{11}$$

In contrast to the RMSE and AEE, the HAE is an optimistic error function since it is dominated by the small error terms. The HAE can be appropriate if the error fluctuates greatly over different runs. This property may be useful in the context of k-fold cross validation with relatively few samples. The GAE, on the other hand, is a balanced error function that suffers much less from extremes (large or small). The GAE, however, has as a disadvantage that if the error is zero in a single point, the overall error is also zero. This is of course may not be desirable.

Many more absolute error variants exist and we do not intend to give an exhaustive overview. Rather we wish to illustrate that each function brings its own tradeoffs and interpretation depending on how the absolute differences $|y_i - \tilde{y}_i|$ are aggregated. In general though, absolute error criteria are not ideally suited for performance estimation of an approximation model due to their context dependence (i.e., τ is hard to specify up front and depends on the units used).

3.1.2 Relative errors

Thus engineers typically prefer relative or percentage errors, e.g., 5%. A figure of 5% implies some kind of global averaged relative error, but there are different ways to calculate relative errors (depending on what reference and aggregation function is used): Average Relative Error (ARE), Maximum Relative Error (MRE), Relative Squared Error (RSE), Root Relative Square Error (RRSE), Relative Absolute Error I (RAEI), Relative Absolute Error II (RAEII), Root Mean Square Relative Error (RMSRE), etc. [31,60].

A natural solution is to take the most intuitive error function, the ARE:

$$ARE(y,\tilde{y}) = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i - \tilde{y}_i|}{|y_i|} \tag{12}$$

By taking the true value as a reference the ARE results in an intuitively understandable number. Multiplied by 100 it results in a natural percentage. However, taking the geometric or harmonic mean (resulting in the Geometric average Relative Error (GRE) and Harmonic average Relative Error (HRE) respectively) instead of the simple average can also be interpreted as a global percentage error. But since ARE, GRE, and HRE treat different types of errors differently (e.g., ARE is more sensitive to large errors than GRE) care should be taken when interpreting a figure like 5%. In addition, the "%" suffix is sometimes also used when using, for example, RRSE (see below). This, however, should be avoided since it is confusing.

The problem with the ARE is that care must be taken in its interpretation when the true function values y_i are small or, in the extreme but not unlikely case, zero. Since then the error tends to infinity, giving a biased result. What is sometimes done to combat this is add one $(+1)$ to the denominator to prevent division by zero. While this solves the numerical issue, the resulting error is an absolute-relative hybrid and becomes impossible to interpret. A different solution is to scale or translate the response to eliminate small absolute values (e.g., as proposed in [41]). However, the best scale factor is not always obvious and shifting the response can introduce its own problems. For example, figure 1 illustrates how a simple shifting of the response (+1000) can drastically improve the ARE value (3 orders of magnitude) for exactly the same model parameters (error measured in the samples).

Additionally, there is the well known issue of averaging the errors (relative or absolute). This means that a model with a low average error can still have areas where the prediction is very poor (i.e., the mean is not robust). Figure 2 shows an example using relative errors. The data in the figure are the result of a NIST study involving semiconductor electron mobility. The response variable is a measure of electron mobility, and the predictor variable is the natural log of the density. The fit is a rational function with a pole. Thus, since an engineer usually requires strict bounds on the maximum error it seems better to minimize the maximum error instead of the average (note that ARE $\leqslant$ MRE).

However, in the relative case, using a maximum aggregation function has its own counter-intuitive properties. For example, figure 3 illustrates how the zero function has a lower MRE than a model which overshoots the data, but else seems like a reasonable fit[1]. This property is particularly problematic if the model parameter space is searched automatically (hyperparameter optimization). In this case the optimization algorithm is easily deceived into generating flat models.

One would be tempted to resort to using the Maximum Absolute Error (MAE) instead. Since while it may be difficult to give a priori average error targets, giving maximum absolute error bounds is often easier since it can be related more directly to the application. However, the MAE is not a satisfactory solution either. First of all, like any absolute er-

[1] In this case the samples are noisy but the same phenomenon can occur with noise free data and a validation set.

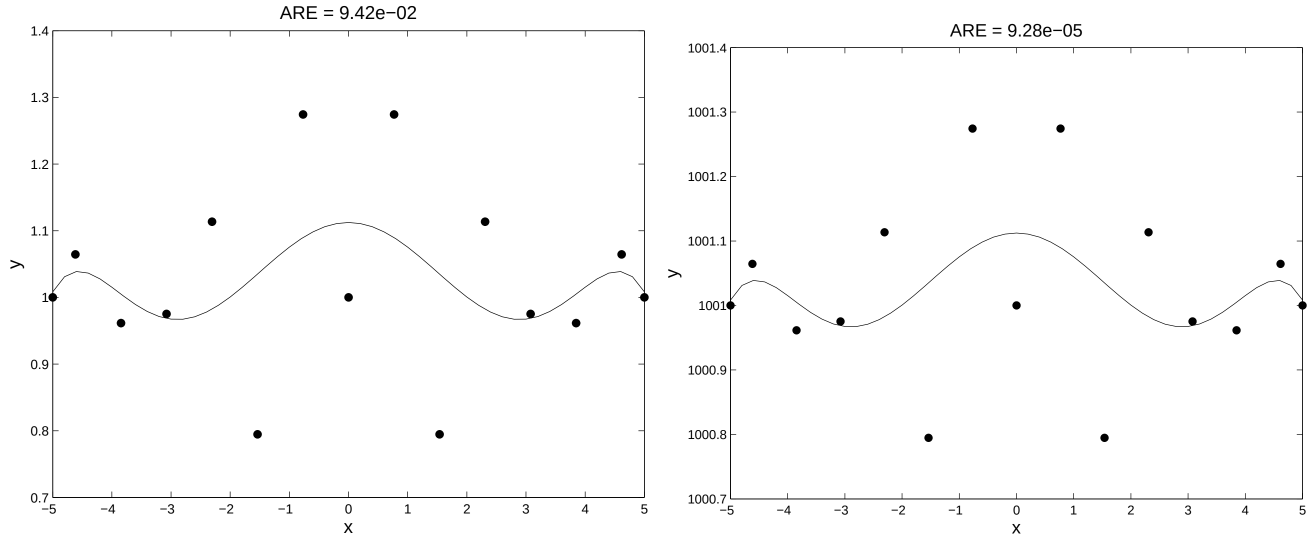

Fig. 1 Influence of shifting the response on the ARE

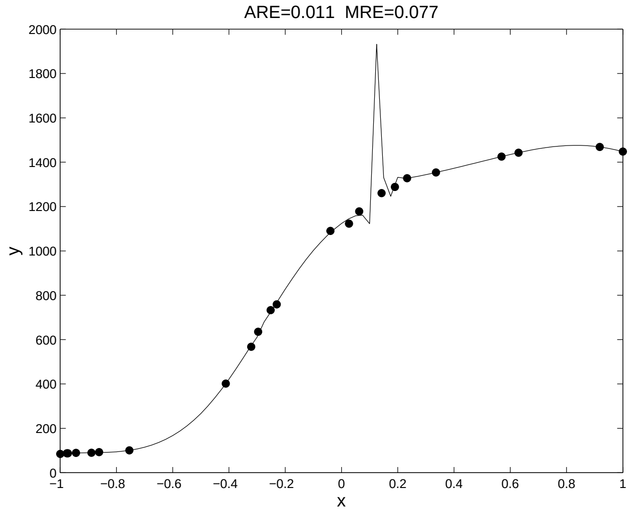

Fig. 2 MRE vs ARE

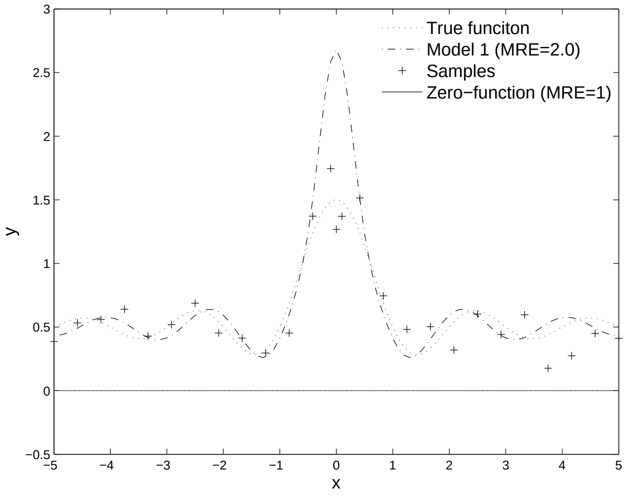

Fig. 3 Comparison of the MRE over different models

ror, it requires knowledge of the full range of the response. Also, it is not relative, meaning a deviation of 5 on a response value 1000 is considered worse than a deviation of 3 on a value of 0.5. Furthermore, enforcing a MAE is equivalent to restricting all fitted response values $\tilde{y}$ to lie inside the tube defined by $[y - MAE, y + MAE]$. This requirement can be too strict if the response contains regions that are very hard to fit (e.g., discontinuities), information that is not always available.

Another approach then, is to use the RRSE function, related to the popular R^2 criterion. In this case the deviation from the mean is used as the reference value.

$$RRSE(y, \tilde{y}) = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{13}$$

The RRSE is intuitively attractive since it measures how much better a fit is over the most simple model possible: the mean. Also it does not become problematic for small absolute values of y_i. Unfortunately, the problem with the RRSE is that it gives a pessimistic estimate of the error if the response that needs to be fitted is very smooth (i.e., the mean is already quite a good fit). Thus an understanding of the structure of the response is needed to properly interpret the RRSE value. The RRSE is also less intuitive for an engineer since it measures the improvement over the average model rather than the quality of fit directly (making a good choice of τ harder).

An improved function that is less sensitive to large errors and has some other attractive properties is given in [60], the Bayesian Estimation Error Quotient (BEEQ):

$$BEEQ(y,\tilde{y}) = \left(\prod_{i=1}^{n} \frac{\sum_{i=1}^{n} |y_i - \tilde{y}_i|}{\sum_{i=1}^{n} |y_i - \bar{y}|} \right)^{\frac{1}{n}} \tag{14}$$

However, like the GAE it will predict an error of zero overall if just a single point has an error of 0.

One could continue discussing different error functions (e.g., those based on the medain or mode) but it should be clear now that each error function has its own characteristics and that relative errors are not always as context free as one might assume at first. While the examples given here are quite simple, they are illustrative of the greater complexities that arise when combining an error function with a model selection metric. Also note that these subtleties are less a problem in classification (where most research on model selection is conducted). The concept of a good classifier is typically much more intuitive to grasp and define by a domain expert than in the case of regression.

Remark that the error function also influences choice of sampling strategy. For example if the error measure dictates that it is important that the optima of the model are captured accurately, one should make sure the sampling strategy employed will sample at those locations. Actually it turns out that in most cases a sampling algorithm can be formulated as a model selection critera and vica versa.

3.2 Choice of model selection metric

Assuming the choice of error function (and target value) can be decided upon there is still the problem of selecting a measure for estimating the generalization capabilities of a model (cross validation, bootstrap, validation set, jack-knife, etc.). This is the well known problem of model selection and has been discussed at length elsewhere [8, 52, 71, 90, 32]. A good high level introduction is given in [105]. The point this paper attempts to make is that it is far from obvious which method to select that, when minimized, produces a model that an engineer is satisfied with. Simply using the in-sample error is useless since it does not account for over-fitting and is meaningless when used with interpolating methods (e.g., Kriging). Measures like AIC and its variants (BIC, CIC, NIC, ...) and the methods from statistical learning theory (Vapnik-Chervonenkis (VC) Dimension, etc.) are more advanced in that they take the complexity of the model into account and have solid foundations in information theory. Unfortunately, an AIC value can only be interpreted relative to another value and has no meaning on its own. They also mean very little to a domain expert.

A validation (hold-out) set is a better solution but it means there is less data available for training. Also, the hold-out error can give extremely biased results (thus deceiving the hyperparameter optimization) if chosen poorly or if only a few points are available. For example, figure 4 gives a simple example were minimizing the validation error can lead to a sub-optimal model. This is of course an extreme example but similar problems are often encountered with real data.

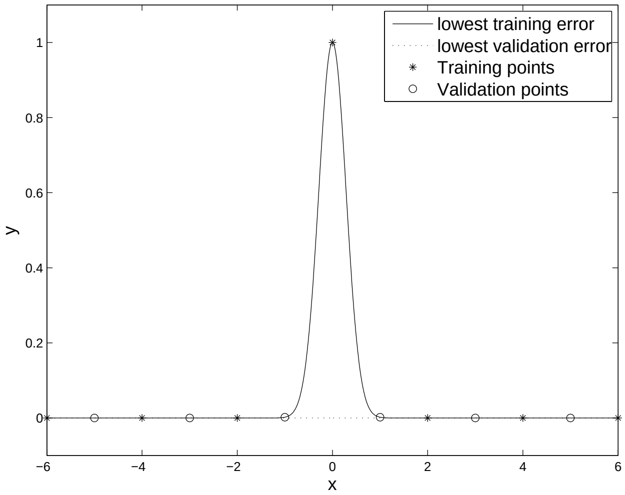

Fig. 4 A misleading validation set

The cross validation error (and its extreme version, the Leave-One-Out error), is a popular compromise, but it too depends on the data distribution [17, 71], can give misleading results [62], and is expensive to compute (the bootstrap even more so). Also there is the question on how to select the folds (randomly, evenly spread, etc.). Additionally one could argue the different cross validation variants should be interpreted as measuring sensitivity to loss of information rather than approximation accuracy. Finally there is the added complication of noise in the data and/or in the generalization estimator (e.g., k-fold cross validation). Since we only consider deterministic computer experiments noisy data is usually not an issue[2]. However, when dealing with measured data or stochastic simulations this adds an extra layer of complexity.

Yet a different approach is to employ Bayesian statistics (see the work by O'Hagan et. al. [73]). Through Bayesian inference one can exactly quantify the uncertainty or confidence one has in a particular model. This is usually very useful from an application standpoint but is only possible with specific model types.

The only true, unbiased test for model quality would be to assess the model on a very dense, independent test set or analytical solution. However, for any real problem this is not a feasible option since data is too expensive and an analytical solution is not available.

[2] In some cases discretization and convergence noise may be present, the magnitude depending on the application.

3.3 The need for handling multiple criteria

In sum, as it should be clear now, it is hard to agree upfront on a single requirement that the final replacement metamodel must respect. The fundamental reason is that an approximation task inherently involves multiple, conflicting, criteria. [60] summarizes this particularly succinctly:

> *It is an illusion that performance evaluation can be done completely fairly and impartially. This is partly because simple metrics cannot capture a complete picture of the performance of an estimation algorithm and those that are more complete [...] are more complex and subject to subjective interpretations. Also, use of any metric in performance evaluation implicitly favors the estimator that tries to optimize this same metric.*

Thus what usually happens in practice is the following: (1) a best effort is made to identify a suitable model selection metric, error function and targets; (2) simulations are performed, the model is generated and delivered to the engineer together with some statistical test results (e.g., different error metrics); and (3) the engineer visually inspects and explores the model and decides if it is satisfactory. If not the process must be repeated.

While the final evaluation stage by a domain expert should always be performed, it would be advantageous if the different desired criteria could be enforced from the start. This can be done in three main ways:

1. the different criteria (objectives) are combined into a single, global criterion which is then used to drive the model generation
2. the different objectives are enforced sequentially in a multi-level process
3. the different objectives are enforced simultaneously through a multiobjective approach

The first option is the easiest and allows existing algorithms to be re-used as is. An example of such scalarization is the geometric mean approach used by Goel et al. in [33]. However the problem remains of choosing an appropriate combination function (and its interpretation) and requiring an understanding of the ranges and nuances of the different member functions. Thus the problem is simply moved to a higher level.

The second option is a sequential or milestone approach. Multiple criteria are supported by specifying different hierarchical levels $L_1,...,L_l$ that must be reached in succession. For example, first the hyperparameter optimization process must produce a model that satisfies L_1 (e.g., a ARE of 5%). Once this target is reached, and only then, is the following level L_2 checked (e.g., a MRE of 10%). Thus, by sequentially working towards subsequent milestones, multiple criteria can be incorporated. The potential problem of this approach is that dependencies and tradeoffs between criteria can cause a deadlock (e.g., reaching one level means violating another). A different way to interpret this is as a constrained optimization problem in the hyperparameter space, each level adds a constraint. Care must be taken that there is at least one feasible region. Also the task for the optimizer (over the model parameters) becomes considerably more difficult since the optimization landscape may change suddenly and drastically when a change in level takes place.

The third solution is to tackle the problem directly as a dynamic multiobjective optimization problem in the hyperparameter space (recall that due to the incremental sampling the optimization surface is dynamic and not static). Each criterion becomes an objective and standard ranking methods are used to identify the Pareto-optimal set. The disadvantage here is that there is no longer the luxury of having a single, unambiguous best solution. However, since we noted above that such a linear ranking makes no sense this should come as no surprise. The advantage is that the problem can be tackled directly using standard algorithms. From the final Pareto set the user is then able to extract knowledge about the problem and make a better decision when choosing the final solution. In addition, the final Pareto front enables the generation of diverse ensembles, where the ensemble members consist of the (partial) Pareto-optimal set (see references in [33,82,49]). In this way all the information in the front can be used. An added advantage of using ensembles is that it allows the calculation of the prediction uncertainty which is very useful for an application.

Finally, one may imagine different hybrid combinations of the three methods mentioned above. For example, the multiobjective approach where the number of objectives varies dynamically. For example, when only little data is available it makes no sense to enforce application specific criteria, or force the model response into particular bounds. That makes more sense when sufficient data is available and the model uncertainty has been reduced. Other combinations are possible but this is a topic that has seen little research and that goes beyond the scope of this paper. Rather we shall concentrate on the multiobjective approach.

4 Modeling multiple outputs

The previous section described how a multiobjective approach to global surrogate modeling can help solve *the 5 percent problem*. A second use case is when dealing with multi-output systems. It is not uncommon that a simulation engine has multiple outputs that all need to be modeled [10]. For example, the combustion problem described in [44] has both a chemical and temperature source term that needs to be modeled. Also many Finite Element packages generate multiple performance values simultaneously.

The direct approach is to model each output independently with separate models (possibly sharing the same data). This, however, leaves no room for trade-offs nor gives any information about the correlation between different outputs. Instead of performing two modeling runs (doing a separate hyperparameter optimization for each output) both outputs can be modeled simultaneously if models with multiple outputs are used in conjunction with a multiobjective optimization routine. The resulting Pareto front then gives information about the accuracy trade-off between the outputs in hyperparameter space and allows the practitioner to choose the model most suited to the particular context. More arguments, of essentially the same discussion, are given in [67].

Again, multi-output Pareto based modeling enables the generation of diverse ensembles. This is a popular approach in rainfall runoff modeling and model calibration in hydrology [91,24]. Models are generated for different flow components and/or derivative measures and these are then combined into a weighted ensemble or fuzzy committee. A Pareto based approach to multi-output modeling also allows integration with the automatic surrogate model type selection algorithm described in [39]. This enables automatic selection of the best model type (Artificial Neural Network (ANN), Kriging, Support Vector Machine (SVM), ...) for each output without having to resort to multiple runs [36,37].

5 Related work

There is a vast body of research available on single objective hyperparameter optimization strategies and model selection criteria for different model types: [9,59,95,90,1,4,75] and the extensive work by Yao et. al. [102,103]. Many authors have noticed the problems with single objective hyperparameter optimization but it is only very recently that multiobjective versions of classical machine learning methods have been presented [70,92,30,47]. An extensive and excellent overview of the work in this area is given by Jin et. al. in [49] and the book (edited by Jin) [46]. By far the majority of the cited work uses multiobjective techniques to improve the training of learning methods. Typically an accuracy criterion (such as the validation error) is used together with some regularization parameter or model complexity measure (e.g., the number of support vectors in SVMs) in order to produce more parsimonious models [25]. Other criteria used include: sensitivity, specificity, interpretability, and number of input features [92,49].

It seems less work has been done on high level objectives (with error functions and generalization estimators in particular) that do not depend on a particular machine learning method. [26] optimize an accuracy metric (the RMSE) together with an application specific *Return* metric useful for stock market forecasting. An example of the use of multiple error measures (and incidentally one of the first formulations of multiobjective learning) is [64] who minimized the L_2-norm, the L_∞-norm and a complexity measure. Unfortunately, a single-objective GA was employed to perform the optimization, resulting in only a single solution. [27] also give an example with two error functions, the Euclidean and robust error which they use to fit a noisy sinusoid with an ANN.

Few references are available that explicitly deal with the trade-offs between different error functions for surrogate modeling. [25] agree that determining the error function is key but do not consider the problem any further. [3] give an extensive treatment of 15 popular error functions for time series extrapolation but is of little use for regression. A more relevant and extensive overview is given by Li and Zhao in [60] who discuss many practical metrics for performance estimation in general and propose a number of new ones. A more restricted and philosophical discussion is given in [43]. The previous references are mainly theoretical. Empirical results on performance estimation are harder to find. One example is [21] who compare four different error functions used for neural network classification training.

Another topic that has been the subject of extensive research is that of multiobjective surrogate based optimization (MOSBO). Surrogate methods are widely used for the optimization of expensive functions [78]. While initially their use has been constrained to the single objective case, an increasing number of results are being reported in the multiobjective case. An example is the work on statistical improvement by Keane et. al. [51] and ParEGO [56], the multiobjective version of the popular Efficient Global Optimization (EGO) approach [50]. Another example is the application to parameter optimization of earth system models in [77], for crashworthiness design optimization in [61], and for thin wall structure optimization in [87]. The well known NSGA-II algorithm [13] has also been extended to incorporate surrogate models [12,98]. In this context some work has also been done on comparing different performance measures for use in MOSBO [54,97]. [54] compare different performance criteria for improving metamodel based optimization. They also "*...recognize that in order to obtain desirable information or knowledge about a response surface, multiple performance measures taken in concert may be necessary.*" Unfortunately they stop there and do not discuss the issue any further. Though the research into MOSBO is still young, an excellent overview of current research is already available in [57].

The contribution of the current work is that it deals with global surrogate modeling with iterative sampling and hyperparameter optimization. The goal is to generate a high fidelity global approximation using as few simulations as possible (replacement metamodeling) and minimizing user interaction. The paper takes an application perspective, and multiobjective optimization is considered on a higher, behavioral level ("What criteria should a model satisfy") versus a more model specific level ("How to generate a parsimonious neural network").

More concretely, the authors stress the importance of a critical analysis of performance estimation criteria and the associated trade-offs when generating surrogates (optimizing the hyperparameters). In particular, a founded choice of error function and target is often overlooked and performance estimation is done in a more ad hoc manner [7, 14], constrained to a single objective [31,23,22], or done a posteriori (after the model parameters have been fixed) to compare different models [72,22]. While the implications and trade-offs of different performance criteria are well described in the statistics community (e.g., [3]), the resulting insights and possible solutions can use more visibility.

In addition we propose to model multi-output simulators simultaneously where this makes sense. Thus giving insight into the modeling trade-off between the outputs and avoiding multiple runs. An added benefit of this approach is the possibility of automatically selecting the best model type for each output. As [57] states *"Little is known about which types of model accord best with particular features of a landscape and, in any case, very little may be known to guide this choice."*. Thus an algorithm to automatically solve this problem is very useful [54]. This is also noticed by [98] who compare different surrogate models for approximating each objective during optimization. They note that in theory their approach allows the use of a different model type for each objective. However, such an approach will still require an a priori model type selection and does not allow for dynamic switching of the model type or the generation of hybrids. We know of no other related work that tackles this issue.

In sum this paper takes a domain expert's point of view. By building on advances and established research in machine learning [49] and statistics [3] we attempt to further improve the global surrogate modeling process and make it more useful and accessible for an engineer. At the same time we hope to increase awareness of the issues involved.

6 Applications

This section presents some concrete illustrations of the ideas and concepts discussed previously. All tests were run using the SUrrogate MOdeling (SUMO) Toolbox which we first briefly discuss below.

6.1 The SUMO Toolbox

The SUMO Toolbox [38,35] is an adaptive tool that integrates different modeling approaches and implements a fully automated, adaptive surrogate model construction algorithm. Given a simulation engine the toolbox produces a surrogate model within the time and accuracy constraints set by the user. Different plugins are supported: model types (rational functions, Kriging, splines, SVM, etc.), model parameter optimization algorithms (BFGS, EGO, simulated annealing, etc.), sample selection (random, error based, density based, etc.), and sample evaluation methods (local, on a cluster or grid). The behavior of each component is configurable through a central XML configuration file and components can easily be added, removed or replaced by custom implementations (see figure 5).

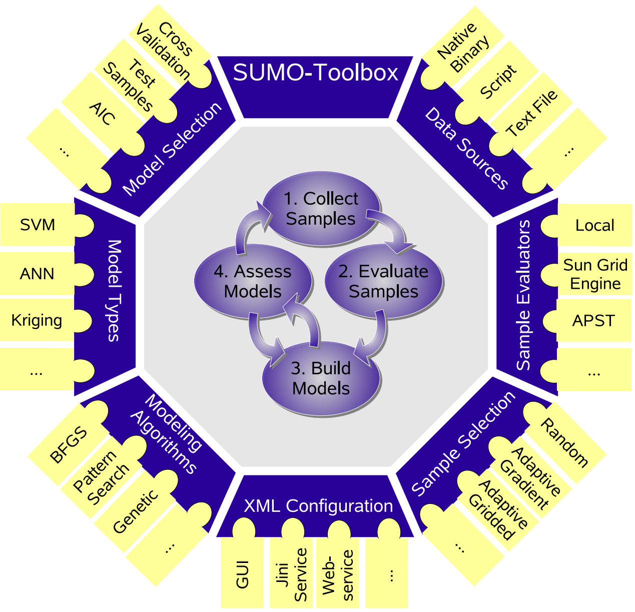

Fig. 5 SUMO Toolbox Plugins

The toolbox control flow is as follows: First, a small initial set of samples is chosen according to some experimental design (e.g., Latin hypercube, Box-Behnken, etc.). Based on this initial set, one or more surrogate models are constructed and their hyperparameters optimized according to a chosen hyperparameter optimization algorithm (e.g., BFGS, Particle Swarm Optimization (PSO), Genetic Algorithm (GA), EGO, DIRECT, NSGA-II, etc.). Models are assigned a score based on one or more measures (e.g., cross validation, AIC, etc.) and the optimization continues until no further improvement is possible. The models are then ranked according to their score and new samples are selected based on the best performing models and the behavior of the response (the exact criteria depend on the sampling algorithm used). The hyperparameter optimization process is continued or restarted intelligently and the whole process repeats itself until one of the following three conditions is satisfied: (1) the maximum number of samples has been reached, (2) the maximum allowed time has been exceeded, or (3) the user required accuracy has been met. Also, the sample evaluation component runs in parallel with the other components (non-blocking) and not sequentially. The toolbox and all algorithms described here is available for download from `http://www.sumo.intec.ugent.be`.

6.2 Low Noise Amplifier (LNA)

6.2.1 Background

We first consider a test case from electronics: a simple RF circuit, a narrow band Low Noise Amplifier (LNA) [58]. A LNA is the typical first stage of a receiver, having the main function of providing the gain needed to win the noise of subsequent stages, such as a mixer. In addition it has to give negligible distortion to the signal while adding as little noise as possible. We have extensively discussed the modeling of this system in [39,40,38]. For this paper we restrict ourselves to the 2D version and will use it to illustrate the use of multiple criteria in generating approximation models.

The input parameters are the (normalized) width of the MOSFET W_n and the normalized inductance L_{sn}. The output is the input noise current $\sqrt{i_{in}^2}$ which previous results have shown to be the most difficult to model [39].

6.2.2 Experimental setup

Previous experience with this function teaches us that this is a difficult function to model accurately with Kriging models (see [40]). Kriging models have difficulty reproducing the smooth surface of the input noise, they suffer from too many unwanted 'ripples' between the data points if a hold-out or cross validation measure is minimized. For this reason we consider two criteria. The first is the RRSE on a 20% min-max validation set, the second, a custom smoothness metric that penalizes a model if it produces ripples between data points.

The SUMO Toolbox (v6.1) was configured to use the Kriging [65] and NSGA-II [13] plugins. For the first run a fixed 7x7 factorial design was used, no additional sampling was performed. For the second run a density based sample selection algorithm was used that covers the design space evenly (previous tests showed it to give the best results with Kriging). Starting from a LHC design of 15 points together with the 4 corner points, the algorithm adds 15 points between each hyperparameter optimization iteration up to a maximum of 400.

For the first run NSGA-II was configured with a population size of 30 and run for a maximum of 250 generations. For the second run the maximum number of generations was set to 20, with the evolution continuing after each sampling iteration. Each individual in the population represents a Kriging model as a tuple (θ_1, θ_2) with θ_i the correlation parameter in log_{10} space ($\theta_i \in [-5, 3]$). The correlation function was set to Gaussian and a linear regression was used.

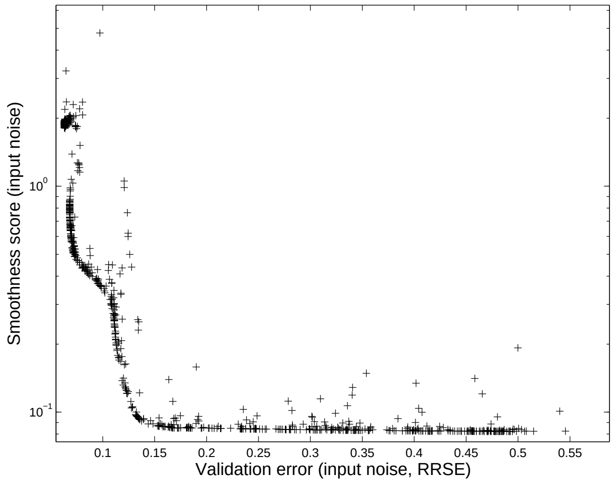

Fig. 6 Pareto search trace for the LNA problem (no sampling)

6.2.3 Results

A plot of the full Pareto search trace for the first run (no sampling) is shown in figure 6. As the figure shows there is a clear trade-off between the two objectives. This can also be seen from the plot of the model at each of the two extreme points (figure 7). Given these results a domain expert now has the flexibility to browse though the front and select the most suitable model.

When sample selection is enabled the optimal Pareto set changes as more data becomes available. The successive Pareto fronts at the start of each sampling iteration are shown in figure 8[3]. The figure clearly shows how the front advances and the model quality improves as more data becomes available. In addition the trade-off in the front seems to decrease as the number of points increase. This should be expected since as the amount of data increases there is less uncertainty about the correct hyperparameter values and the agreement between both measures increases.

6.3 Automotive problem

The second example is an application from the automotive industry (see [31] for details) and illustrates the modeling of a multi-output system.

[3] A movie showing the evolution is available at http://sumolab.blogspot.com/

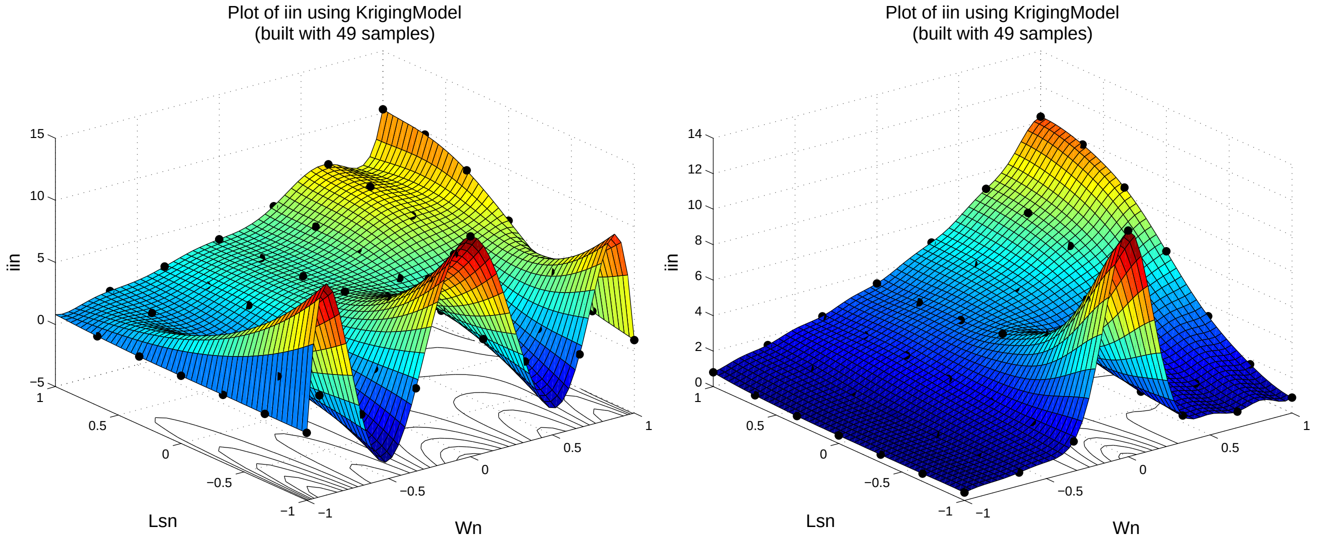

Fig. 7 Plot of the models at the extreme Pareto points for the LNA problem (no sampling, left: minimal validation error, right: minimal smoothness penalty)

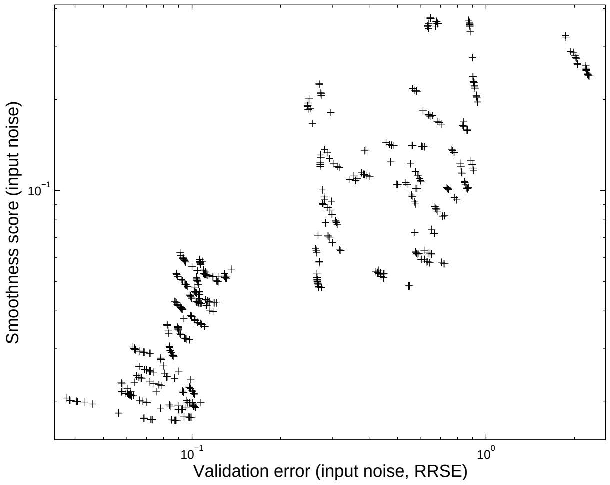

Fig. 8 Pareto search trace for the LNA problem (sampling enabled)

6.3.1 Background

Today the early concept phase of a car body development process is marked by the optimal coordination of design specifications with the requirements on the mechanical behavior of the structure as well as on the feasibility. This planning process is repetitive for the same body parts and the solution finding is carried out mostly by experience with an additional virtual tryout afterwards in order to improve the solution. The use of surrogate modeling can enable an early feasibility prediction of body parts.

The geometry of a B-pillar bottom of a side frame is shown in figure 9. There you have a recurring feasibility challenge in sheet metal forming that can be explained by radii, depths and angles as experience shows. Which of these geometry parameters and in which combination they have an effect on the feasibility is, however, intuitively hard to predict. For simulation purposes the door entry area can be separated from the side frame by simple boundary conditions without major restrictions for the validity of the analysis but computing times considerably go down.

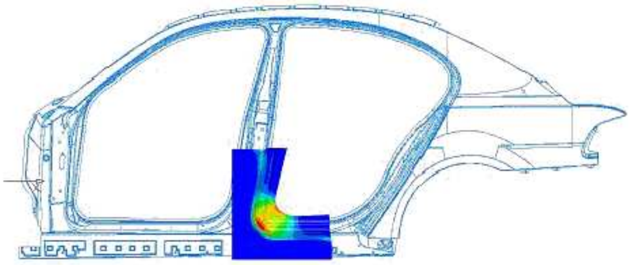

Fig. 9 B-pillar bottom of a side frame [31]

The entry angle α_1, opening angle α_2, frame depth h and entry radius r, have been chosen as geometry parameters (see [31] for details). In addition, for every geometry constellation the blank boundary was determined so that the forming result was optimal. Additional process parameters, like draw bead or blank holder forces, have not been used. So there were six parameters that have been taken into account. With these input quantities a sampling based on a LHC was created. Maximum scaled distances of the strain states to the forming limit curve and a maximum thinning limit respectively were chosen as output quantities indicating feasibility. The data sampling phase resulted in 1998 data points evaluated that were suitable for modeling. The overall target for this particular problem setting was to predict a given set of geometry parameters as feasible, i.e., to predict the existence of cracks (*cracking* output) or unacceptable thinning (*thinning* output).

6.3.2 Experimental setup

Both outputs shall be modeled together using the ANN and LS-SVM plugins of the SUMO Toolbox. The ANN models are based on the Matlab Neural Network Toolbox and are trained with Levenberg Marquard backpropagation with Bayesian regularization [66,28] (300 epochs). The topology and initial weights are determined by a GA. The LS-SVM models are based on the LS-SVMlab toolbox plugin [93] and the hyperparameters are searched in log_{10} space with $\sigma \in [-4,4]$, $c \in [-5,5]$ (an RBF kernel is used). The multiobjective algorithm used is the one implemented in the Matlab GADS toolbox which, in turn, is based on NSGA-II. The population size is set to 10. For comparison each output will be modeled separately as well (single objective).

In all cases the metric used to drive the hyperparameter optimization is the Average Relative Error (ARE) on 5-fold cross validation. For the single objective runs the timeout was 25 generations, for the multiobjective runs the timeout was 50 generations.

6.3.3 Results

Figure 10 shows the final error curves after the SUMO Toolbox has terminated. A point is plotted for each time the toolbox finds a model that improves on the previous model. As can be seen from the figure, the ANN models clearly outperform the SVM models, especially for the *cracking* output. One could argue the poor performance of the LS-SVM models is due to poor hyperparameter optimization. However, this is not the case. For reference a brute force search of the hyperparameter landscape was conducted on a 50 by 60 grid. This is shown in figure 11 (bounds in log_{10} scale, the white crosses show the area explored by the SUMO Toolbox). The minimum found through this search:

$$f_{cracking}(-0.1600,-1.4993) = 0.1348$$

$$f_{thinning}(0,-2.9996) = 0.0730$$

is comparable to the minimum found by the SUMO Toolbox:

$$f_{cracking}(-0.2173,0.2978) = 0.1280$$

$$f_{thinning}(0.0423,1.0948) = 0.0741$$

Thus the hyperparameter optimization is not to blame (remember that the cross validation procedure introduces some noise into the surface). A more extensive cross validation (15 folds) was also done on the final best model in each case (table 1). As can be seen, the accuracy remains unchanged.

	cracking	*thinning*
ANN	0.0414	0.0325
LS-SVM	0.1305	0.0741

Table 1 ARE on 15-fold cross validation of the final models (automotive example)

The poor performance of SVMs in this case is in line with the author's previous experience. We found SVM models to require too much data when a non-linear, noise-free response needs to be fitted smoothly and accurately. In those cases, SVM models are very good at fitting the non-linear regions but generate unwanted 'ripples' in the regions where the response needs to be smooth or data is sparse. ANN models on the other hand, are able to adapt much better to the heterogeneity of the response. The sigmoid transfer functions allow for high non-linearity, while proper training (e.g., through the use of regularization) ensures a smooth fit in the sparse regions.

Figure 12 shows the full trace of the multiobjective hyperparameter optimization. In this case the model generation is driven by a 2-objective (= the cross validation score on each output) optimization algorithm. In both cases it is immediately clear that there is no real Pareto front, a single best model can be identified in each case. Thus this teaches us that there is a very strong correlation between both outputs and that good performance on one output, implies good performance on the other. This is actually to be expected since *cracking* and *thinning* are closely related (as can also be seen from figure 11).

Of course this is not always the case (see for example [37]). It is not always clear how much the outputs are really correlated, or how much one quality metric influences another (in the case of multiple metrics). We argue that in those cases a direct multiobjective approach should be considered. It is guaranteed to give at least as much information as doing multiple single objective runs for about the same computational cost (which is still outweighed by the cost of the simulation). Also, it gives the engineer more flexibility and is a cleaner approach than manually combining the multiple objectives into a single formula.

6.4 Chemistry problem

6.4.1 Background

This example and its description is taken from [44], where the authors describe the generation of an optimal ANN using a pattern search algorithm. We use this example to briefly illustrate the automatic model type selection per output. For a more extensive example see [36].

The chemical process under consideration describes methane-air combustion. The GRI 2.11 chemical mechanism contain-

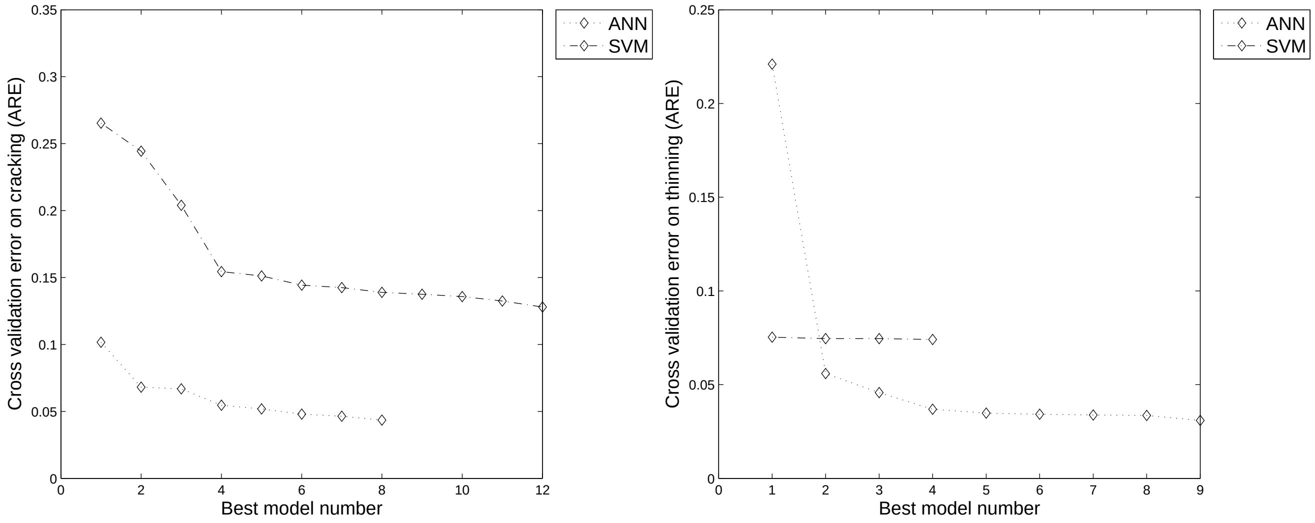

Fig. 10 Model accuracies in the single objective case (left: *cracking*, right: *thinning*)

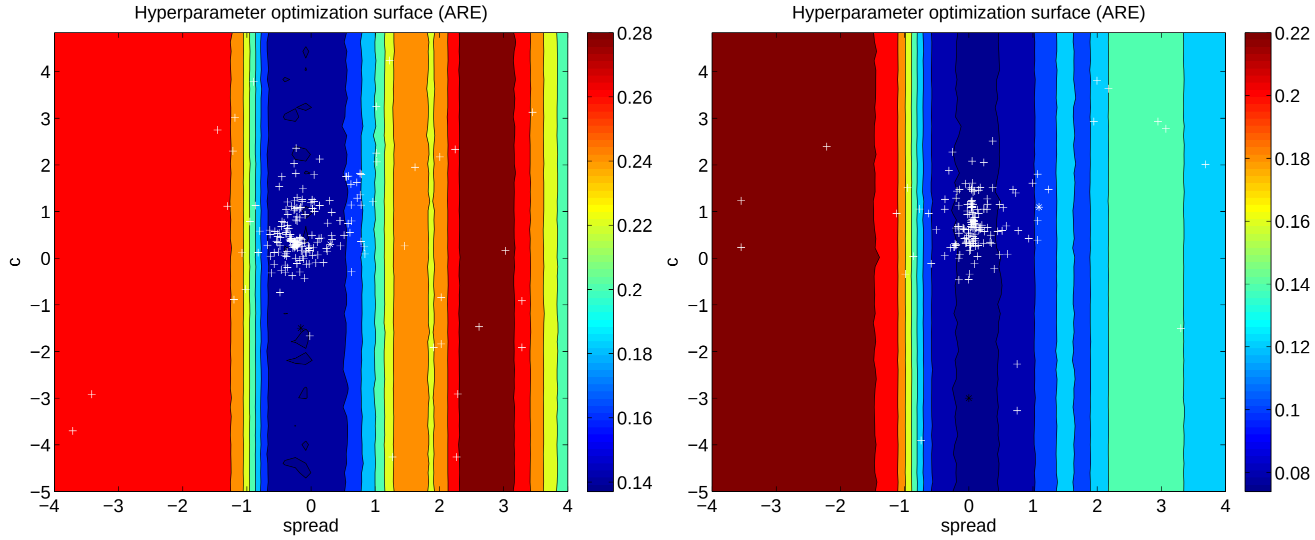

Fig. 11 SVM hyperparameter optimization surface (left: *cracking*, right: *thinning*)

ing 277 elementary chemical reactions among 49 species is used. The steady laminar flamelet equations [76] are often employed to describe the reaction-diffusion balance in non-premixed flames. The solutions to these equations provide temperature and mass fractions of all species in terms of two parameters. The mixture fraction z and the reaction progress variable c are used for this parametrization. The two responses are the temperature and the chemical source term of c, which can be viewed as a measure of heat release.

For the approximation 1000 data samples are available, half of which will be used for training, the other half to drive the hyperparameter optimization. Sample data were obtained by applying an acceptation-rejection method [81].

6.4.2 Experimental setup

The heterogeneous evolution plugin of the SUMO Toolbox is used and configured with the following model types: RBF ANNs, LS-SVMs, and Rational functions. Together with the ensemble models (which result from a heterogeneous crossover, e.g., a crossover between a neural network and a rational function), this makes that 4 model types will compete to fit the data. The GA used is the NSGA-II based algorithm as implemented in the Matlab GADS toolbox. The population size of each model type is set to 10 and the evolution was run for 290 generations. A full discussion of the automatic model type selection algorithm is out of scope for this paper. Such details can be found in [34, 39]. The difference with the work discussed in [39] is that now the algorithms have been extended to the multiobjective case.

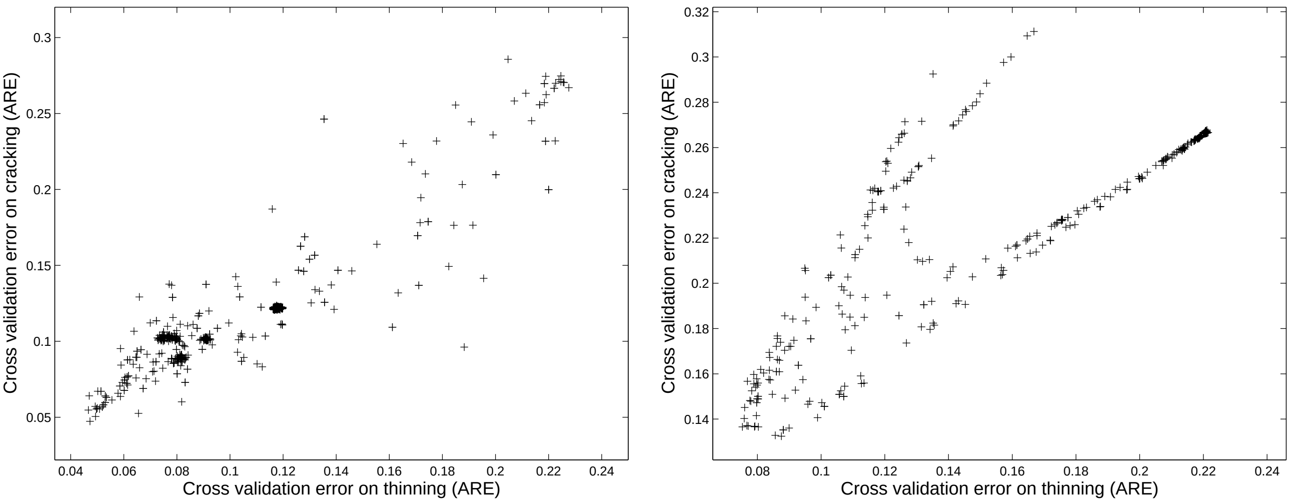

Fig. 12 Model accuracies in the multiobjective case (left: ANN, right: SVM)

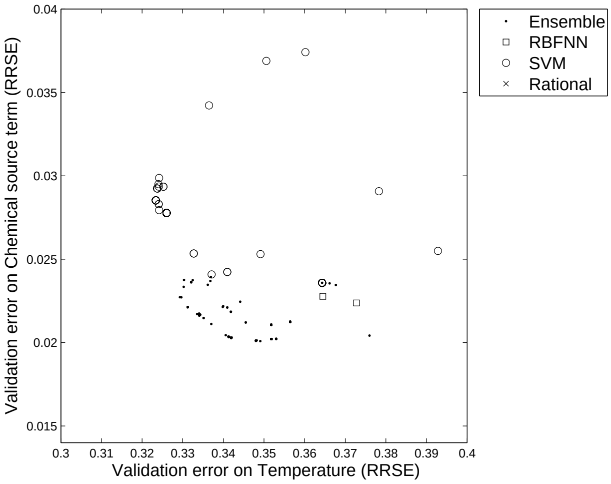

Fig. 13 Heterogeneous Pareto trace

6.4.3 Results

The full Pareto trace (enlarged for clarity) is shown in figure 13. The figure shows that the LS-SVM models are best at fitting the temperature output, while fitting the chemical source term works best with a combination of models (ensemble). The ensembles turn out to consist of a combination of LS-SVM and RBFNN models. The rational functions turn out to perform very poorly on this data and are thus not shown on the (enlarged) figure. This trace can now also be used to generate a global ensemble of models (e.g., for uncertainty estimation).

7 Summary and conclusion

The use of surrogate models to aid optimization, design exploration, sensitivity analysis, etc. has become standard practice among scientists and engineers alike. This work has concentrated on the construction of global surrogate models. A crucial problem of generating global surrogate models for a particular application (or any function approximation task for that matter), is agreeing upfront with the domain expert what criteria the final surrogate should satisfy. The problem is that each criterion (encompassing an error function, generalization estimator, and target value) involves a tradeoff between interpretability, accuracy, bias, and computational efficiency. Thus, for cases where this trade-off cannot be inferred from domain knowledge or application constraints the authors advocate a multiobjective approach to solving this problem should be considered. The advantage of a multiobjective approach is also that it allows multiple outputs to be modeled together, giving information about the tradeoff in the hyperparameter space. It further enables the generation of diverse ensembles and the application of an automatic model type selection algorithm. This enables each output to be automatically modeled with the most suitable model type. There is also some empirical evidence that the number of local optima can be reduced by converting multi-modal single-objective problems, into multiobjective ones [49]. If the same can be proven in machine learning it means the task of identifying good surrogate models can become easier through a multiobjective approach.

However, a disadvantage of the multiobjective approach is that as the number of dimensions (criteria/outputs) increases the solution space increases exponentially [85]. Thus the search for the Pareto optimal set becomes harder, requires more search iterations, and the final set is more cumbersome for the practitioner to explore and understand. For costly simulation codes the extra computational effort is negligible, and good GUI tools can help a domain expert understand the relationships present in the Pareto-optimal set. However, for cheaper codes a trade off between simulation cost and modeling cost will have to be made. The poor scalability of non-dominated sorting algorithms above 4 dimensions is also an issue [57]. Luckily, algorithmic advances (e.g., [83, 69]) and gains in computational efficiency (e.g., [45]) continue to be made.

A disadvantage of the multiobjective approach versus the milestone approach is that the direct multiobjective approach takes all criteria into account straight away. This is not necessarily a problem but is not always the most computationally efficient. For example, in the case of adaptive sampling it makes no sense to check or enforce an (expensive) application specific constraint if only a few data points are available. The model first needs to mature by incorporating more data before undergoing more stringent checks. In this case the number of objectives varies dynamically and thus a scalarized multiobjective approach with a dynamically varying weighting parameter (as discussed in [48]) can be useful. Alternatively a cooling approach as done in [96] could be used.

Thus, naturally much work remains. First of all, while support for multiple criteria is already very useful, more research is needed on intuitive criteria. Ideally criteria should be easily formulated in language that a domain expert is comfortable with and fully understands. Fuzzy theory can be helpful in this respect. Besides researching the feasibility of fuzzy criteria more work still needs to be done on classic model selection methods and explore the relationship with a constraint based approach. This to fully understand the relationship between error function and generalization estimator, and how they impact the final response. A way to vary the criteria dynamically with the sample selection loop would also be useful as is the study of transductive learning [86]. A possible integration with domain partitioning methods (e.g., as done in [41]) is also promising.

Another area requiring further investigation is understanding how the iterative sample selection process influences the hyperparameter optimization landscape. There is a mutual dependency between the model type, hyperparameter optimization strategy, and sampling strategy (e.g., see [40]). The exact nature of this dependency depends on the model type. Determining how they interact and may be optimally combined is a topic of ongoing research. For the tests in this paper the authors have simply let the optimization continue from the previous generation. However, some initial tests have shown that an intelligent restart strategy can improve results. Knowledge of how the number and distribution of data points affects the hyperparameter surface (determined by some metric) would allow for a better tracking of the optimum, reducing the computational cost. The influence of noise and discrete variables on the hyperparameter optimization (e.g., neural network topology selection) also remains an issue.

In general, while some progress towards dynamic multiobjective optimization has been made [63, 42], this is a topic that current research in multiobjective surrogate modeling is only just coming to terms with [57]. Or as English pithily puts it: *"Optimization is easy, learning is hard (in the typical function)."* [20]

Acknowledgments

The authors wish to thank Markus Ganser and Karen Grossenbacher from BMW motor company for making the automotive data available and the fruitful discussions. The authors also thank Jeroen Croon from the NXP-TSMC Research Center, Device Modeling Department, Eindhoven, The Netherlands for the LNA simulation code. Finally, the authors also thank Matthias Ihme from Standford University for providing the chemistry combustion data.

References

1. Abd El-Sallam, A., Kayhan, S., Zoubir, A.: Bootstrap and backward elimination based approaches for model selection. In: Image and Signal Processing and Analysis, 2003. ISPA 2003. Proceedings of the 3rd International Symposium on, vol. 1, pp. 152–157Vol.1 (2003). DOI 10.1109/ISPA.2003.1296885
2. Alexandrov, N.M.: On managing the use of surrogates in general nonlinear optimization and mdo. In: 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis (1998)
3. Armstrong, J.S., Collopy, F.: Error measures for generalizing about forecasting methods: Empirical comparisons. International Journal of Forecasting **8**(1), 69–80 (1992). URL `http://ideas.repec.org/a/eee/intfor/v8y1992i1p69-80.html`
4. Bartlett, P.L., Boucheron, S., Lugosi, G.: Model selection and error estimation. Machine Learning **48**(1-3), 85–113 (2002)
5. Barton, R.R.: Design of experiments for fitting subsystem metamodels. In: WSC '97: Proceedings of the 29th conference on Winter simulation, pp. 303–310. ACM Press, New York, NY, USA (1997). DOI http://doi.acm.org/10.1145/268437.268495
6. Booker, A.J., Dennis, J.E., Frank, P.D., Serafini, D.B., Torczon, V., Trosset, M.W.: A rigorous framework for optimization of expensive functions by surrogate. Structural and Multidisciplinary Optimization **17**(1), 1–13 (1999)
7. Busby, D., Farmer, C.L., Iske, A.: Hierarchical nonlinear approximation for experimental design and statistical data fitting. SIAM Journal on Scientific Computing **29**(1), 49–69 (2007). DOI 10.1137/050639983
8. Chen, H., Huang, S.: A comparative study on model selection and multiple model fusion. In: Information Fusion, 2005 8th International Conference on, vol. 1, p. 7pp. (2005). DOI 10.1109/ICIF.2005.1591938
9. Chen, P.W., Wang, J.Y., Lee, H.M.: Model selection of SVMs using GA approach. In: Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on, vol. 3, pp. 2035–2040 (2004)

10. Conti, S., O'Hagan, A.: Bayesian emulation of complex multi-output and dynamic computer models. research report no. 569/07, submitted to journal of statistical planning and inference. Tech. rep., Department of Probability and Statistics, University of Sheffield (2007)
11. De Geest, J., Dhaene, T., Faché, N., De Zutter, D.: Adaptive CAD-model building algorithm for general planar microwave structures. IEEE Transactions on Microwave Theory and Techniques **47**(9), 1801–1809 (1999)
12. Deb, K., Nain, P.K.S.: An evolutionary multi-objective adaptive meta-modeling procedure using artificial neural networks. In: S. Yang, Y.S. Ong, Y. Jin (eds.) Evolutionary Computation in Dynamic and Uncertain Environments, *Studies in Computational Intelligence*, vol. 51, pp. 297–322. Springer (2007). URL `http://dblp.uni-trier.de/db/series/sci/sci51.html\#DebN07`
13. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. Evolutionary Computation, IEEE Transactions on **6**(2), 182–197 (2002). DOI 10.1109/4235.996017
14. Devabhaktuni, V., Chattaraj, B., Yagoub, M., Zhang, Q.J.: Advanced microwave modeling framework exploiting automatic model generation, knowledge neural networks, and space mapping. IEEE Tran. on Microwave Theory and Techniques **51**(7), 1822–1833 (2003). DOI 10.1109/TMTT.2003.814318
15. Devabhaktuni, V.K., Zhang, Q.J.: Neural network training-driven adaptive sampling algorithm. In: Proceedings of 30th European Microwave Conference, Paris, France, vol. 3, pp. 222–225 (2000)
16. Ding, M., Vemur, R.: An active learning scheme using support vector machines for analog circuit feasibility classification. In: 18th International Conference on VLSI Design, pp. 528–534 (2005). DOI 10.1109/ICVD.2005.47
17. Efron, B.: The estimation of prediction error: Covariance penalties and cross-validation. Journal of the American Statistical Association **99**, 619–632 (2004). URL `http://ideas.repec.org/a/bes/jnlasa/v99y2004p619-632.html`
18. Eldred, M.S., Dunlavy, D.M.: Formulations for surrogate-based optimization wiht data fit, multifidelity, and reduced-order models. In: 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Protsmouth, Virginia (2006)
19. Eldred, T., Willcox, M., Robinson, K., Haimes, R.: Strategies for multifidelity optimization with variable dimensional hierarchical models. In: In Proceedings of the 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference (2nd AIAA Multidisciplinary Design Optimization Specialist Conference), Newport, Rhode Island (2006)
20. English, T.: Optimization is easy and learning is hard in the typical function. Proceedings of the 2000 Congress on Evolutionary Computation, 2000. **2**, 924–931 vol.2 (2000). DOI 10.1109/CEC.2000.870741
21. Falas, T., Stafylopatis, A.G.: The impact of the error function selection in neural network-based classifiers. Neural Networks, 1999. IJCNN '99. International Joint Conference on **3**, 1799–1804 vol.3 (1999). DOI 10.1109/IJCNN.1999.832651
22. Fang, H., Rais-Rohani, M., Liu, Z., Horstemeyer, M.: A comparative study of metamodeling methods for multiobjective crashworthiness optimization. Computers and Structures **83**, 2121–2136 (2005)
23. Farhang-Mehr, A., Azarm, S.: Bayesian meta-modelling of engineering design simulations: a sequential approach with adaptation to irregularities in the response behaviour. International Journal for Numerical Methods in Engineering **62**(15), 2104–2126 (2005). DOI 10.1002/nme.1261
24. Fenicia, F., Solomatine, D.P., Savenije, H.H.G., Matgen, P.: Soft combination of local models in a multi-objective framework. Hydrology and Earth System Sciences Discussions **4**(1), 91–123 (2007). URL `http://www.hydrol-earth-syst-sci-discuss.net/4/91/2007/`
25. Fieldsend, J.E.: Multi-objective supervised learning. In: J. Knowles, D. Corne, K. Deb (eds.) Multiobjective Problem Solving from Nature From Concepts to Applications, Natural Computing Series. Springer LNCS (2008)
26. Fieldsend, J.E., Singh, S.: Pareto multiobjective nonlinear regression modelling to aid capm analogous forecasting. In: Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on, vol. 1, pp. 388–393 (2002). DOI 10.1109/IJCNN.2002.1005503
27. Fieldsend, J.E., Singh, S.: Pareto evolutionary neural networks. Neural Networks, IEEE Transactions on **16**(2), 338–354 (2005). DOI 10.1109/TNN.2004.841794
28. Foresee, F., Hagan, M.: Gauss-newton approximation to bayesian regularization. In: Proceedings of the 1997 International Joint Conference on NeuralNetworks, pp. 1930–1935 (1997)
29. Forrester, A.I.J., Bressloff, N.W., Keane, A.J.: Optimization using surrogate models and partially converged computational fluid dynamics simulations. Proceedings of the Royal Society **462**, 2177–2204 (2006)
30. Furukawa, T., Lee, C.J.K., Michopoulos, J.: Regularization for parameter identification using multi-objective optimization. In: Multi-Objective Machine Learning, pp. 125–149 (2006)
31. Ganser, M., Grossenbacher, K., Schutz, M., Willmes, L., Back, T.: Simulation meta-models in the early phases of the product development process. In: Proceedings of Efficient Methods for Robust Design and Optimization (EUROMECH 07) (2007)
32. Goel, T., Haftka, R., Shyy, W.: Comparing error estimation measures for polynomial and kriging approximation of noise-free functions. Journal of Structural and Multidisciplinary Optimization **28**(5), 429–442 (2009)
33. Goel, T., Haftka, R., Shyy, W., Queipo, N.: Ensemble of surrogates. Structural and Multidisciplinary Optimization **33**, 199–216 (2007). DOI doi:10.1007/s00158-006-0051-9
34. Gorissen, D.: Heterogeneous evolution of surrogate models. Master's thesis, Master of AI, Katholieke Universiteit Leuven (KUL) (2007)
35. Gorissen, D.: Grid-enabled adaptive metamodeling and active learning for computer based design. In: Proceedings of The 22nd Canadian Conference on Artificial Intelligence (AI 2009), Kelowna, pp. 266–269. Springer - Lecture Notes in Artificial Intelligence, Vol. LNCS 5549 (2009)
36. Gorissen, D., Couckuyt, I., Crombecq, K., Dhaene, T.: Pareto-based multi-output model type selection. In: Proceedings of the 4th International Conference on Hybrid Artificial Intelligence (HAIS 2009), Salamanca, Spain, pp. 442–449. Springer - Lecture Notes in Artificial Intelligence, Vol. LNCS 5572 (2009)
37. Gorissen, D., Couckuyt, I., Laermans, E., Dhaene, T.: Pareto-based multi-output metamodeling with active learning. In: Proceedings of the 11th International Conference on Engineering Applications of Neural Networks (EANN 2009), London, England (2009)
38. Gorissen, D., De Tommasi, L., Crombecq, K., Dhaene, T.: Sequential modeling of a low noise amplifier with neural networks and active learning. Neural Computing and Applications **18**(5), 485–494 (2009)
39. Gorissen, D., De Tommasi, L., Croon, J., Dhaene, T.: Automatic model type selection with heterogeneous evolution: An application to rf circuit block modeling. In: Proceedings of the IEEE Congress on Evolutionary Computation, WCCI 2008, Hong Kong (2008)
40. Gorissen, D., De Tommasi, L., Hendrickx, W., Croon, J., Dhaene, T.: Rf circuit block modeling via kriging surrogates. In: Proceedings of the 17th International Conference on Microwaves, Radar and Wireless Communications (MIKON 2008) (2008)

41. Hamad, H., Al-Smadi, A.: Space partitioning in engineering design via metamodel acceptance score distribution. Eng. Comput. (Lond.) **23**(3), 175–185 (2007)
42. Hatzakis, I., Wallace, D.: Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach. In: GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation, pp. 1201–1208. ACM, New York, NY, USA (2006). DOI http://doi.acm.org/10.1145/1143997.1144187
43. Hennig, C., Kutlukaya, M.: Some thoughts about the design of loss functions. REVSTAT - Statistical Journal **5**, 19–39 (2007)
44. Ihme, M., Marsden, A.L., Pitsch, H.: Generation of optimal artificial neural networks using a pattern search algorithm: Application to approximation of chemical systems. Neural Computation **20**, 573–601 (2008)
45. Jensen, M.: Reducing the run-time complexity of multiobjective eas: The nsga-ii and other algorithms. Evolutionary Computation, IEEE Transactions on **7**(5), 503–515 (2003). DOI 10.1109/TEVC.2003.817234
46. Jin, Y. (ed.): Multi-Objective Machine Learning, *Studies in Computational Intelligence*, vol. 16. Springer (2006)
47. Jin, Y.: Pareto-based multi-objective machine learning. In: Hybrid Intelligent Systems, 2007. HIS 2007. 7th International Conference on, pp. 2–2 (2007). DOI 10.1109/HIS.2007.73
48. Jin, Y., Okabe, T., Sendhoff, B.: Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does It Work and How? In: L. Spector, E.D. Goodman, A. Wu, W. Langdon, H.M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M.H. Garzon, E. Burke (eds.) Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001), pp. 1042–1049. Morgan Kaufmann Publishers, San Francisco, California (2001)
49. Jin, Y., Sendhoff, B.: Pareto-based multiobjective machine learning: An overview and case studies. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on **38**(3), 397–415 (2008). DOI 10.1109/TSMCC.2008.919172
50. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. of Global Optimization **13**(4), 455–492 (1998). DOI http://dx.doi.org/10.1023/A:1008306431147
51. Keane, A.J.: Statistical improvement criteria for use in multiobjective design optimization. AIAA Journal **44**(4), 879–891 (2006)
52. Kenneth P. Burnham, D.A.: Model Selection and Multi-Model Inference. Springer (2003)
53. Keys, A.C., Rees, L.P.: A sequential-design metamodeling strategy for simulation optimization. Comput. Oper. Res. **31**(11), 1911–1932 (2004). DOI http://dx.doi.org/10.1016/S0305-0548(03)00146-1
54. Keys, A.C., Rees, L.P., Greenwood, A.G.: Performance measures for selection of metamodels to be used in simulation optimization. Decision Sciences **33**, 31 – 58 (2007)
55. Kleijnen, J.P., Sanchez, S.M., Lucas, T.W., Cioppa, T.M.: State-of-the-art review: A user's guide to the brave new world of designing simulation experiments. INFORMS Journal on Computing **17**(3), 263–289 (2005)
56. Knowles, J.: Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. IEEE Transactions on Evolutionary Computation **10**(1), 50–66 (2006)
57. Knowles, J., Nakayama, H.: Meta-modeling in multiobjective optimization. In: Multiobjective Optimization - Interactive and Evolutionary Approaches. Springer LNCS, In press (2008)
58. Lee, T.H.: The Design of CMOS Radio-Frequency Integrated Circuits, 2 edn. Cambridge University Press (2004)
59. Lessmann, S., Stahlbock, R., Crone, S.: Genetic algorithms for support vector machine model selection. In: Neural Networks, 2006. IJCNN '06. International Joint Conference on, pp. 3063–3069 (2006)
60. Li, X.R., Zhao, Z.: Evaluation of estimation algorithms part I: incomprehensive measures of performance. IEEE Transactions on Aerospace and Electronic Systems **42**(4), 1340–1358 (2006). DOI 10.1109/TAES.2006.314576
61. Liao, Xingtao, Li, Qing, Yang, Xujing, Zhang, Weigang, Li, Wei: Multiobjective optimization for crash safety design of vehicles using stepwise regression model. Structural and Multidisciplinary Optimization **35**(6), 561–569 (2008). DOI 10.1007/s00158-007-0163-x. URL `http://dx.doi.org/10.1007/s00158-007-0163-x`
62. Lin, Y.: An efficient robust concept exploration method and sequential exploratory experimental design. Ph.D. thesis, Georgia Institute of Technology (2004)
63. Liu, C., Wang, Y.: Dynamic multi-objective optimization evolutionary algorithm. In: Natural Computation, 2007. ICNC 2007. Third International Conference on, vol. 4, pp. 456–459 (2007). DOI 10.1109/ICNC.2007.340
64. Liu, G., Kadirkamanathan, V.: Learning with multi-objective criteria. Fourth International Conference on Artificial Neural Networks pp. 53–58 (1995)
65. Lophaven, S.N., Nielsen, H.B., Søndergaard, J.: Aspects of the matlab toolbox DACE. Tech. rep., Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby (2002)
66. MacKay, D.J.C.: Bayesian model comparison and backprop nets. In: J.E. Moody, S.J. Hanson, R.P. Lippmann (eds.) Advances in Neural Information Processing Systems 4, pp. 839–846. Morgan Kaufmann, San Mateo, California (1992)
67. Matsuyama, Y.: Harmonic competition: a self-organizing multiple criteria optimization. Neural Networks, IEEE Transactions on **7**(3), 652–668 (1996). DOI 10.1109/72.501723
68. Meckesheimer, M., Booker, A.J., Barton, R., Simpson, T.: Computationally inexpensive metamodel assessment strategies. AIAA Journal **40**(10), 2053–2060 (2002)
69. Mehnen, J., Wagner, T., Rudolph, G.: Evolutionary optimization of dynamic multi-objective test functions. In: S. Cagnoni, L. Vanneschi (eds.) Digital Proceedings of the 3° Workshop Italiano di Vita Artificiale e della 2a Giornata di Studio Italiana sul Calcolo Evoluzionistico. LabSEC – Laboratorio Simulazioni di fenomeni Socio Economici Complessi (2006)
70. Mierswa, I.: Controlling overfitting with multi-objective support vector machines. In: GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation, pp. 1830–1837. ACM, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1276958.1277323
71. Molinaro, A.M., Simon, R., Pfeiffer, R.M.: Prediction error estimation: a comparison of resampling methods. Bioinformatics **21**(15), 3301–3307 (2005). DOI http://dx.doi.org/10.1093/bioinformatics/bti499
72. Mullur, A., Messac, A.: Metamodeling using extended radial basis functions: a comparative approach. Eng. with Comput. **21**(3), 203–217 (2006). DOI http://dx.doi.org/10.1007/s00366-005-0005-7
73. O'Hagan, A.: Bayesian analysis of computer code outputs: a tutorial. Reliability Engineering and System Safety **91**, 1290–1300 (2006)
74. Ong, Y.S., Nair, P., Lum, K.: Max-min surrogate-assisted evolutionary algorithm for robust design. Evolutionary Computation, IEEE Transactions on **10**(4), 392–404 (2006). DOI 10.1109/TEVC.2005.859464
75. Ou, Y.Y., Chen, C.Y., Hwang, S.C., Oyang, Y.J.: Expediting model selection for support vector machines based on data reduction. In: Systems, Man and Cybernetics, 2003. IEEE International Conference on, vol. 1, pp. 786–791vol.1 (2003). DOI 10.1109/ICSMC.2003.1243910

76. Peters, N.: Laminar diffusion flamelet models in non-premixed turbulent combustion. Progress in Energy and Combustion Science **10**(3), 319–339 (1984). DOI http://dx.doi.org/10.1016/0360-1285(84)90114-X
77. Price, A., Voutchkov, I., Pound, G., Edwards, N., Lenton, T., Cox, S.: Multiobjective tuning of grid-enabled earth system models using a non-dominated sorting genetic algorithm (nsga-ii). In: e-Science and Grid Computing, 2006. e-Science '06. Second IEEE International Conference on, pp. 117–117 (2006). DOI 10.1109/E-SCIENCE.2006.261050
78. Queipo, N.V., Arévalo, C.J., Pintos, S.A.: The integration of design of experiments, surrogate modeling and optimization for thermoscience research. Eng. Comput. (Lond.) **20**(4), 309–315 (2005)
79. Queipo, N.V., Haftka, R.T., Shyy, W., Goel, T., Vaidyanathan, R., Tucker, P.K.: Surrogate-based analysis and optimization. Progress in Aerospace Sciences **41**, 1–28 (2005)
80. Robertazzi, T.G., Schwartz, S.C.: An accelerated sequential algorithm for producing d-optimal designs. Siam Journal on scientific Computing **10**, 341–358 (1989)
81. Rubinstein, R.Y.: Simulation and the Monte Carlo Method. John Wiley & Sons, Inc., New York, NY, USA (1981)
82. Sanchez, E., Pintos, S., Queipo, N.: Toward an optimal ensemble of kernel-based approximations with engineering applications. In: In Proceedings of the International Joint Conference on Neural Networks, 2006. IJCNN '06., pp. 2152–2158 (2006). DOI 10.1109/IJCNN.2006.246987
83. Sangkawelert, N., Chaiyaratana, N.: Diversity control in a multi-objective genetic algorithm. In: Evolutionary Computation, 2003. CEC '03. The 2003 Congress on, vol. 4, pp. 2704–2711Vol.4 (2003). DOI 10.1109/CEC.2003.1299430
84. Sasena, M.J., Papalambros, P.Y., Goovaerts, P.: Metamodeling sampling criteria in a global optimization framework. In: 8th AIAA/ USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Long Beach, CA, AIAA Paper 2000-4921. (2000)
85. Sastry, K., Goldberg, D., Pelikan, M.: Limits of scalability of multiobjective estimation of distribution algorithms. In: Evolutionary Computation, 2005. The 2005 IEEE Congress on, vol. 3, pp. 2217–2224Vol.3 (2005). DOI 10.1109/CEC.2005.1554970
86. Schwaighofer, A., Tresp, V.: Transductive and inductive methods for approximate gaussian process regression. In: NIPS, pp. 953–960 (2002)
87. Sheriffa, N.M., Guptab, N., Velmuruganc, R., Shanmugapriyand, N.: Optimization of thin conical frusta for impact energy absorption. Thin-Walled Structures **46**(6), 653–666 (2008)
88. Simpson, T.W., Poplinski, J.D., Koch, P.N., Allen, J.K.: Metamodels for computer-based engineering design: Survey and recommendations. Eng. Comput. (Lond.) **17**(2), 129–150 (2001)
89. Simpson, T.W., Toropov, V., Balabanov, V., Viana, F.A.C.: Design and analysis of computer experiments in multidisciplinary design optimization: a review of how far we have come or not. In: Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2008 MAO, Victoria, Canada (2008)
90. Smets, K., Verdonk, B., Jordaan, E.M.: Evaluation of performance measures for svr hyperparameter selection. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN2007) (2007)
91. Solomatine, D.P., Ostfeld, A.: Data-driven modelling : some past experiences and new approaches. Journal of hydroinformatics **10**(1), 3–22 (2008)
92. Suttorp, T., Igel, C.: Multi-objective optimization of support vector machines. In: Multi-Objective Machine Learning, pp. 199–220 (2006)
93. Suykens, J., Gestel, T.V., Brabanter, J.D., Moor, B.D., Vandewalle, J.: Least Squares Support Vector Machines. World Scientific Publishing Co., Pte, Ltd., Singapore (2002)
94. Toal, D.J., Bressloff, N.W., Keane, A.J.: Kriging hyperparameter tuning strategies. AIAA Journal **46**(5), 1240–1252 (2008)
95. Tomioka, S., Nisiyama, S., Enoto, T.: Nonlinear least square regression by adaptive domain method with multiple genetic algorithms. IEEE Transactions on Evolutionary Computation **11**(1), 1–16 (2007)
96. Turner, C.J., Crawford, R.H., Campbell, M.I.: Multidimensional sequential sampling for nurbs-based metamodel development. Eng. with Comput. **23**(3), 155–174 (2007). DOI http://dx.doi.org/10.1007/s00366-006-0051-9
97. Vasanth Kumar, K., Porkodi, K., Rocha, F.: Comparison of various error functions in predicting the optimum isotherm by linear and non-linear regression analysis for the sorption of basic red 9 by activated carbon. Journal of hazardous materials **150**, 158–165 (2008)
98. Voutchkov, I., Keane, A.: Multiobjective Optimization using Surrogates. In: I. Parmee (ed.) Adaptive Computing in Design and Manufacture 2006. Proceedings of the Seventh International Conference, pp. 167–175. Bristol, UK (2006)
99. Wang, G.G., Shan, S.: Review of metamodeling techniques in support of engineering design optimization. Journal of Mechanical Design **129**(4), 370–380 (2007). DOI 10.1115/1.2429697
100. Yang, C., Meza, J.C., Wang, L.W.: A trust region direct constrained minimization algorithm for the kohn-sham equation. SIAM Journal on Scientific Computing **29**(5), 1854–1875 (2007). DOI http://dx.doi.org/10.1137/060661442
101. Yang, Y., Barron, A.: An asymptotic property of model selection criteria. Information Theory, IEEE Transactions on **44**(1), 95–116 (1998). DOI 10.1109/18.650993
102. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE **87**(9), 1423–1447 (1999). DOI 10.1109/5.784219
103. Yao, X., Xu, Y.: Recent advances in evolutionary computation. J. Comput. Sci. Technol. **21**(1), 1–18 (2006)
104. Zhou, Z., Ong, Y.S., Nair, P.: Hierarchical surrogate-assisted evolutionary optimization framework. In: Congress on Evolutionary Computation (CEC2004), vol. 2, pp. 1586–1593Vol.2 (2004). DOI 10.1109/CEC.2004.1331085
105. Zucchini, W.: An introduction to model selection. Journal of Mathematical Psychology **44**, 41–61 (2000)