First Investigations on Noisy Model-Based Multi-Objective Optimization

Daniel Horn[1], Melanie Dagge[2], Xudong Sun[3], and Bernd Bischl[3]

[1] TU Dortmund University, Computational Statistics, 44227 Dortmund, daniel.horn@tu-dortmund.de
[2] TU Dortmund University, Statistics with Applications in the Field of Engineering Sciences, 44227 Dortmund, melanie.dagge@tu-dortmund.de
[3] LMU Munich, Computational Statistics, 80539 Munich, {xudong.sun,bernd.bischl}@stat.uni-muenchen.de

Abstract. In many real-world applications concerning multi-objective optimization, the true objective functions are not observable. Instead, only noisy observations are available. In recent years, the interest in the effect of such noise in evolutionary multi-objective optimization (EMO) has increased and many specialized algorithms have been proposed. However, evolutionary algorithms are not suitable if the evaluation of the objectives is expensive and only a small budget is available. One popular solution is to use model-based multi-objective optimization (MBMO) techniques. In this paper, we present a first investigation on noisy MBMO. For this purpose we collect several noise handling strategies from the field of EMO and adapt them for MBMO algorithms. We compare the performance of those strategies in two benchmark situations: Firstly, we perform a purely artificial benchmark using homogeneous Gaussian noise. Secondly, we choose a setting from the field of machine learning, where the structure of the underlying noise is unknown.

Keywords: Noisy Optimization, Machine Learning, Bayesian Optimization, Model-Based Optimization, Multi-Objective Optimization

1 Introduction

In many practical optimization scenarios it is necessary to consider multiple contradicting objectives. In such multi-objective optimization (MOO) problems several solutions with optimal trade-offs across the objectives are available. Most MOO algorithms assume that repeated evaluations of the same solution will yield the same objective values. However, in many applications this assumption is violated due to the presence of noise.

The field of machine learning offers a lot of different settings in which multiple performance measures can be considered: runtime vs. predictive power, model sparsity vs. predictive power or multiple measures from ROC analysis [8]. ROC measures are, in the case of binary classification, derived from the so-called confusion matrix, which tabulates the classes of the prediction versus the classes of

the true outcome. Examples are the true-positive and false-positive rate, whose trade-off is usually interesting to consider in imbalanced problems or problems with unknown error cost structure. Machine learning models are usually evaluated by resampling techniques [2]. These split the dataset repeatedly into a training and a corresponding test set – the training sets are used to train the models, their performance is evaluated on the test sets. Naturally, the resulting performance depends on the splits used, which leads to noise in the optimization.

In recent years, many strategies have been proposed to solve noisy MOOs. Most of them are based on EMO algorithms [1,6,9,14,23]. However, evolutionary algorithms (EAs) suffer from some major disadvantages. One drawback is the large number of function evaluations required to achieve a good approximation of the optimal set, however, in many practical settings the budget is limited. For example, the training of a support vector machine (SVM) on a dataset containing about a million samples can take several hours [12]. Hence, it is often not practical to tune the hyperparameters of a SVM on such datasets using EMO techniques. We formerly proposed to use model-based multi-objective optimization (MBMO) to tackle such problems [11]. However, very little work has been published on MBMO algorithms solving noisy MOOs.

Knowles et al. [18] used ParEGO [17], a special MBMO variant to solve noisy MOOs. They were able to show that ParEGO is quite robust to the presence of noise and still achieves reasonable results. However, in settings with noisy observations ParEGO was outperformed by two specialized algorithms [10, 28]. One approach of handling noise in MBMO algorithms is to use the mean value of replicated evaluations for each parameter setting [6]. In [19] this approach was compared with a re-interpolating approach.

The main contribution of this paper is a first look into noise handling using MBMO algorithms and the presentation of two naive and two advanced noise handling strategies. All of them can be added to any MBMO algorithm. In two benchmark studies we investigate their effectiveness, however, we are not yet able to give a precise rule which strategy should be used in which situation.

2 Model-Based Multi-Objective Optimization

Multi-objective optimization refers to an optimization setting with a vector-valued objective function $\boldsymbol{f} = (f_1, f_2, \ldots, f_m) : \mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_d \to \mathbb{R}^m$ and with the usual convention that each f_i is to be minimized. In general, multiple objectives are contradicting, and we are interested in finding the set of all optimal trade-offs. A solution $\boldsymbol{x} \in \mathcal{X}$ is said to dominate a solution $\boldsymbol{x}' \in \mathcal{X}$ if $\boldsymbol{x}$ is at least as good as $\boldsymbol{x}'$ in all objectives and strictly better in at least one objective. This dominance relation is sufficiently strong for a definition of optimality: a solution $\boldsymbol{x} \in \mathcal{X}$ is called Pareto optimal if and only if it is not dominated by any other solution $\boldsymbol{x}' \in \mathcal{X}$. The set of all non-dominated solutions is called the Pareto set, the set $\left\{ \boldsymbol{f}(\boldsymbol{x}) \,\middle|\, \boldsymbol{x} \in \mathcal{X} \text{ is Pareto optimal} \right\}$ is called the Pareto front.

Many different MBMO algorithms have been proposed within the last years. Most of them are based on ideas of the Efficient Global Optimization (EGO)

Algorithm 1 General sequential model-based optimization procedure

Require: expensive blackbox function $\boldsymbol{f}(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}^m$, budget n
1: generate initial design $\mathcal{D} = \{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(n_{init})}\} \subset \mathcal{X}$
2: evaluate $\mathcal{Y} = \boldsymbol{f}(\mathcal{D}) = \{\boldsymbol{f}(\boldsymbol{x}^{(1)}), ..., \boldsymbol{f}(\boldsymbol{x}^{(n_{init})})\}$
3: set $i := n_{init} + 1$
4: **while** i**++** $\leq n$ **do**
5: fit surrogate model(s) $\hat{\boldsymbol{f}}$ on $\mathcal{Y}$ and $\mathcal{D}$
6: $\boldsymbol{x}^{(i)} = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \text{Inf}(\boldsymbol{x})$
7: evaluate $\boldsymbol{y}^{(i)} = \boldsymbol{f}(\boldsymbol{x}^{(i)})$
8: update $\mathcal{D} \leftarrow \mathcal{D} \cup \{\boldsymbol{x}^{(i)}\}$ and $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\boldsymbol{y}^{(i)}\}$
9: **end while**
10: **return** Pareto set and front of $\mathcal{D}$ and $\mathcal{Y}$

procedure [15]. At first EGO samples and evaluates an initial design of size n_{init}, typically using Latin Hypercube Sampling.

Then a surrogate regression model (typically a Kriging model) is fitted to all design points obtained so far and optimized with regard to a so-called infill criterion (Inf). A new candidate point is proposed and evaluated with the objective function. The procedure iterates until a stopping criterion is reached, which is usually a budget on the number of function evaluations or global runtime.

The general procedure of MBMO algorithms is described in Algorithm 1. Although most steps of the MBMO procedure are close to the EGO algorithm, there are some important differences. Mainly, the model fitting procedure has to be adapted to the multi-objective context. Most MBMO algorithms are adjusted by fitting distinct surrogate models to each objective function. As in EGO, new points are proposed by optimizing an infill criterion. However, this step cannot be adopted directly from EGO, since there might be multiple surrogates.

There is a variety of MBMO algorithms, which are all based on this general procedure, but they differ in the number of fitted models, the infill criterion, the optimization strategy and many more details. An overview of many published MBMO algorithms including their taxonomy is presented in [13].

Although our work on noisy MBMO is compatible with any MBMO algorithm, we decided to focus on one algorithm for simplicity. The SMS-EGO [20] algorithm with its good performance in previous benchmark studies (see [11,13]) appears to be a good choice for this. It works by fitting individual Kriging models to each objective and choosing new design points by maximizing their hypervolume contribution to the current Pareto front approximation.

3 Noisy Multi-Objective Optimization

In noisy MOO the true objective functions $\boldsymbol{f}$ cannot be observed. Instead, we have observations $\boldsymbol{y}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{\epsilon}(\boldsymbol{x})$ that are affected by observational noise $\boldsymbol{\epsilon}(\boldsymbol{x}) = \{\epsilon_1(\boldsymbol{x}), ..., \epsilon_m(\boldsymbol{x})\}$. In the general blackbox case we cannot make any assumptions about the characteristics of $\boldsymbol{\epsilon}$. Since this case is hard to handle, most

noise handling algorithms do make further assumptions on $\boldsymbol{\epsilon}$. Typical assumptions are: $\boldsymbol{\epsilon}$ is stochastically independent of $\boldsymbol{x}$, $\boldsymbol{\epsilon}$ is stochastically independent of the point in time of the evaluation, ϵ_j is stochastically independent and identically distributed for $j = 1, ..., m$ or that the expected value of $\boldsymbol{\epsilon}$ is $\mathbf{0}$ [16].

Noise introduces two main challenges that should be addressed by the specialized optimizer:

(I) The noisy observations provide misleading information. This can induce insufficient results that are not close enough to the true optimal Pareto set.
(II) Noise may lead to overoptimistic assessment of obtained objectives. Thus, the returned Pareto front is very likely to overrate the true Pareto front.

The main idea for handling noise is to perform k (additional) evaluations for each $\boldsymbol{x}$-setting, also called re-evaluations. This reduces the noise's standard deviation by a factor of $\sqrt{k+1}$ [14]. Assuming $E(\boldsymbol{\epsilon}) = \mathbf{0}$, the mean value of all re-evaluations of $\boldsymbol{x}$ is an unbiased estimator for $\boldsymbol{f}(\boldsymbol{x})$. On the one hand its quality can be improved by increasing k, on the other hand the optimizer has to spend a part of its budget for each re-evaluation. In particular, in settings with restricted budgets, the number of re-evaluations should be kept as small as possible.

In recent years, many different EMO strategies have been proposed to solve noisy MOOs. Most of them describe how to choose $\boldsymbol{x}$-settings for re-evaluations. In this chapter we will give a brief introduction to some naive and advanced approaches. For an overview please refer to Zhou et al [27].

3.1 Naive Variants

The following two naive strategies for choosing the re-evaluations serve as a basis for the advanced methods. Due to their simplicity both strategies can be added to most optimizer, in particular to EAs, MBMO and even random search.

The Enlarged Strategy The simplest idea is to ignore the fact that $\boldsymbol{y}$ is affected by noise and perform optimization as if $\boldsymbol{y}$ was deterministic (i.e. assume that $\boldsymbol{y} = \boldsymbol{f}$). Hence, every setting $\boldsymbol{x}$ is evaluated once, after it was proposed by the algorithm ($k = 0$). Since no budget needs to be spend on re-evaluations, a higher number of different $\boldsymbol{x}$ settings can be evaluated (the space of explored, distinct points is *enlarged*). Since there is no special strategy for handling noise, enlarged algorithms neither address (I) nor (II). (I) may be solved by the brute force of the larger budget, but the returned Pareto front of enlarged algorithms is very likely to be too optimistic.

The Repeated Strategy The second naive strategy is to perform a constant number of k re-evaluations for every $\boldsymbol{x}$-setting directly (the evaluations are *repeated*). Then the mean value of all $k + 1$ evaluations is used as the new target value for optimization and optimization is performed as if deterministic. This intuitive strategy reduces the strength of noise for each $\boldsymbol{x}$ by a factor of $\sqrt{k+1}$ and has been applied in some practical scenarios (see, e.g., [6]). It is also highly related to resampling strategies from machine learning applications. Performing a k-fold cross-validation is similar to performing k evaluations of the same

$\boldsymbol{x}$-setting. By applying a repeated strategy, the effects of (I) and (II) can be reduced.

3.2 Advanced Strategies

It seems not that meaningful to evaluate a weak noisy, inferior point as often as a very noisy, promising setting. There are several approaches to adapt the number of re-evaluations during optimization. One idea is to increase the number of re-evaluations while optimization proceeds [1]. Other options are to choose the number of re-evaluations according to the observed standard deviation [1] or to use a statistical testing procedure for comparing two solutions and to perform re-evaluations until their difference becomes significant to a given level [23].

In this paper we focus on two different advanced strategies. Firstly, the Rolling Tide EA (RTEA) proposed by Fieldsend [9], which gives a heuristic on how to choose promising settings for re-evaluations. Secondly, we propose a new strategy that combines the ideas of the two described naive approaches.

The Rolling Tide Strategy The main idea of the RTEA is described in Algorithm 2. It handles noise by re-evaluating only promising settings, while inferior settings are evaluated only once. In each iteration, after a new point has been proposed using evolutionary operators, k already evaluated points are chosen for re-evaluation. The selection of these k points is based on the dominance relation and the number of prior re-evaluations: Sequentially, those Pareto optimal points with the least number of re-evaluations are chosen. New points are only proposed in the first part of the optimization, the second part is used to solely refine the archive. The actual RTEA contains more details on how to perform the evolutionary operations and how to efficiently update the Pareto set. We omit these details since they are not related to noise handling.

The Reinforced Strategy This strategy combines the ideas of the enlarged and the repeated algorithm aiming to eliminate the disadvantages of these naive

Algorithm 2 The Rolling Tide Evolutionary Algorithm (RTEA)

Require: noisy blackbox function $\boldsymbol{y}(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}^m$, number of re-evaluations k, proportion of evaluations z to solely refine the archive, budget n

Generate and evaluate initial design of size n_{init}
Set $i := n_{init}$
while i**++** $\leq n$ **do**
  if $i < (1 - z) \cdot n$ **then**
    Propose a new $\boldsymbol{x}^{(i)}$ using evolutionary operators
    Calculate $\boldsymbol{y}(\boldsymbol{x}^{(i)})$ and update archive and Pareto set A
  end if
  for k iterations **do**
    choose $\boldsymbol{x}^{(j)} \in A$ with the lowest amount of re-evaluations
    Re-evaluate $\boldsymbol{y}(\boldsymbol{x}^{(j)})$ and update both the archive and A
  end for
end while
return Pareto set A and corresponding Pareto front

strategies. Our first results showed that the main drawback of the enlarged strategy are overly optimistic Pareto fronts, while for the repeated strategy it is the high number of unnecessary re-evaluations. Therefore, we propose to start the optimization using the enlarged strategy and only one evaluation per $\boldsymbol{x}$. In the end k re-evaluations are done for each Pareto optimal point (the final Pareto front is *reinforced*). Thus, the reinforced strategy uses a large budget just like the enlarged strategy, but also returns a reliable estimation of the Pareto front.

3.3 Noise Handling in MBMO Algorithms

To the best of our knowledge, very little work has been published on noise handling in MBMO algorithms up to today. The general capability of MBMO algorithms to solve noisy MOOs was shown in [18]. There are also investigations on solving noisy applications by using the repeated strategy [6, 19] and some specialized algorithms have been introduced [10, 28]. However, no work on adapting advanced strategies for choosing re-evaluations has been published yet. Since all described strategies solving noisy MOOs can be used within the MBMO framework, we will employ these approaches as a basis for further investigations.

As representative for advanced noise handling strategies we chose RTEA and adapted it toward MBMO by exchanging the evolutionary operations from the RTEA by their MBMO equivalents. This is implemented by inserting lines 8 to 11 of Algorithm 2 into Algorithm 1 after line 9. Moreover, in the RTEA each point is evaluated directly after it was chosen for re-evaluation. We suggest to collect the new proposed point and all k points for re-evaluation in a batch in order to evaluate them in parallel. This procedure comes with the disadvantage that the Pareto front cannot be updated in between the re-evaluations.

Almost all MBMO algorithms use Kriging models as surrogates. But in terms of noisy optimization the interpolating nature of Kriging is problematic: For each $\boldsymbol{x}^{(i)} \in \mathcal{D}$, the model will return the corresponding $\boldsymbol{y}^{(i)} \in \mathcal{Y}$. Consequently, Kriging models cannot be fitted to repeated stochastic evaluations.

We are aware of two solutions for this issue: Firstly, the model can be fitted on the mean values of the repeated evaluations. Thus, information about the strength of the noise is not used. Secondly, by adding a so-called nugget effect to the Kriging model the model loses its interpolating nature. It becomes an approximating model and can be fitted to repeated evaluations. This is achieved by introducing parameter $\lambda \in [0, \infty)$, which controls the strength of smoothing, where $\lambda = 0$ corresponds to use no smoothing, i.e., use interpolating Kriging. Here we focus on the first solution planning to investigate the second one in subsequent studies.

The final Kriging models can also be used to improve the optimization result. Instead of returning the Pareto front of $\mathcal{Y}$ (the *tune front*) the front of the models (the *model front*) can be used. For this the Pareto set of $\mathcal{D}$ is calculated and their outcomes are predicted by using the final models. Naturally, the tune front and the model front are identical, if interpolating Kriging is used. However, if a small nugget effect is added, the model may smooth the affected tune front. Especially for the enlarged strategy we expect the model front to be more realistic.

4 Artificial Experiment

To test and compare the proposed naive and advanced noise handling strategies, we conduct two benchmarks. In this section, we use artificial test functions contaminated with homogeneous Gaussian noise.

4.1 Experimental Setup

For comparison of our four proposed approaches, the noise handling strategies are fused with the SMS-EGO algorithm. Additionally, these algorithms are compared against a baseline comprised of the original RTEA and the two naive strategies combined with a simple random search strategy (rs). This results in a total of seven different algorithms. They are abbreviated using the pattern class_noisehandling, e.g, mbmo_rt refers to the SMS-EGO algorithm using the rolling tide (rt) noise handling strategy. As test functions we use UF1-UF10 from the CEC 09 challenge [26]. For noise we add normal distributed random vectors with expected value $\mathbf{0}$ and covariance matrix $\sigma \cdot \mathbf{I}_m$, $\sigma \in \{0, 0.01, 0.1, 1\}$. This setting mostly matches with an experimental setup proposed by Fieldsend [9].

To simulate an expensive setting, we limit the budget to $40d$ iterations, of which $8d$ are reserved for the initial design and $32d$ for the sequential optimization. The noise handling strategy is allowed to perform k additional evaluations in each sequential iteration, resulting in a total budget of $40d+32d\,k$ evaluations. Table 1 shows how the different strategies utilize their budget. Since SMS-EGO is an expensive optimization algorithm (m Kriging models have to be fitted in each iteration), we decide to investigate a scenario with $d = 5$ and $k = 4$.

Additional parameters of the SMS-EGO are set as in previous benchmarks [11]. As surrogate we use a Kriging model and add a small nugget effect ($\lambda = 10^{-8}$) for numerical stability. For the RTEA we apply simulated binary crossover ($n = 10, p = 0.7$) and polynomial mutation ($n = 25, p = 0.3$).

The analysis is based on the so-called *test front*, which is generated by evaluating the final Pareto set on the true, deterministic function. Performance of the different approaches is evaluated and compared by two different measures: Firstly, goodness of optimization (see (I)) is measured by the dominated hypervolume based on the test fronts. For statistical analysis we perform an

Table 1. Partition of the budget for the individual noise handling strategies. The total budget is $8d + 32d(k + 1)$ for all strategies. The number of iterations for the reinforced variant changes with the size of the final Pareto front.

Algorithm	token	n_{init}	Evaluations per iteration	Number of iterations
enlarged	enl	$8d$	1	$32d(k+1)$
repeated	rep	$8d(k+1)$	$k+1$	$32d - \lfloor \frac{k}{k+1} 8d \rfloor$
rolling tide	rt	$8d$	$k+1$	$32d$
reinforced	reinf	$8d$	1	-

ANOVA and post-hoc pairwise one-sided paired t-tests on the normalized hypervolume values [7]. Secondly, to judge the approximation quality of the tune front (see (II)), we calculate the so-called Average Hausdorff distance [22] (parameter $p = 1$) between the tune front and the test front.

The experiments are implemented using our own software packages written in R, including mlrMBO [5], mlr [3,21] and BatchExperiments [4]. Our experiments are executed on the LiDOngrL cluster of the TU Dortmund university.

4.2 Results

The resulting hypervolume values for all algorithms, test functions and different noise strengths are displayed in Fig. 1. First of all, we can state that all SMS-EGO variants beat the three baselines in nearly all cases.

In the deterministic case ($\sigma = 0$) the results support our expectation. Since re-evaluating points does not give additional information, the enlarged and reinforced strategies perform best. Surprisingly, this result does also hold for the moderate noise cases $\sigma \in \{0.01, 0.1\}$. The hypervolume values of the individual algorithms are only slightly smaller than the values in the deterministic case, while the ranking of the algorithms stays the same. It seems that ignoring the noise is the best way to handle it.

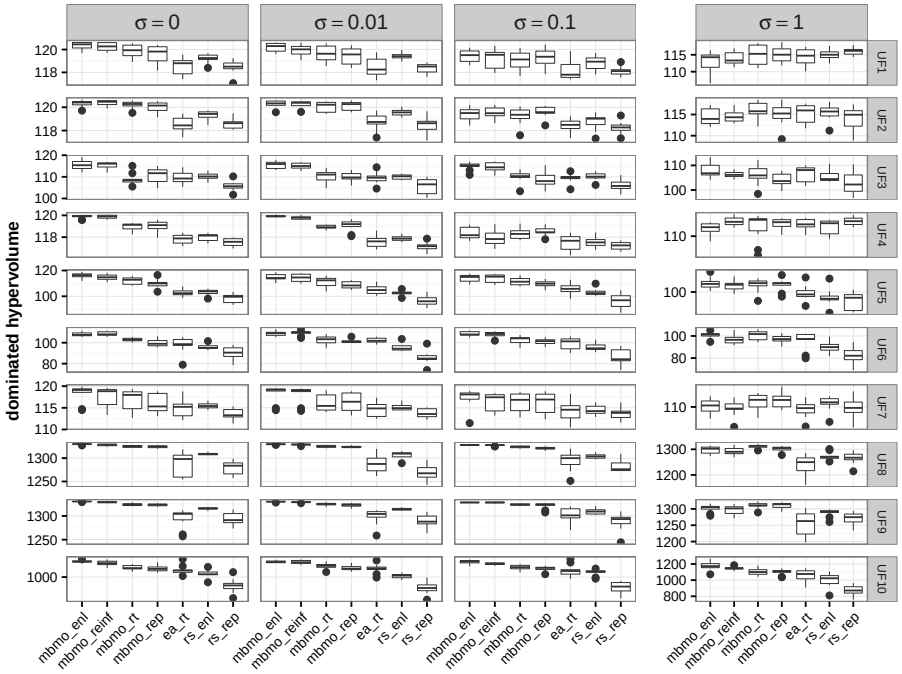

Fig. 1. Dominated hypervolume of the seven algorithms on the ten test functions. Reference point is (11, 11) for UF1 - UF7 and (11, 11, 11) for UF8 - UF10.

This general picture changes in the presence of strong noise ($\sigma = 1$). Performance of all algorithms deteriorates and there are no clear differences in hypervolume between the algorithms on most of the test functions. On some test functions the performance of SMS-EGO is even equivalent to the random search baselines. We speculate that the combination of the large noise and the low budget is too complicated to be solved much better than using simple random search. In fact, all Pareto fronts of the UF test functions range from $(0, 1)$ to $(1, 0)$, but the noise can range within $[-3, 3]$ [4]. Hence, the noise is even stronger than the actual trade-offs between the objectives.

The results of the ANOVA are presented in Table 2. All parameters have a highly significant influence, though the low p-values are intensified by the rather large number of 2 800 observations. Table 3 shows the results of the post-hoc t-tests, which mainly confirm the impression we got from the figures. The tests give a clear ranking of the algorithms and some groups of strategies can be identified. The best group contains the enlarged and the reinforced SMS-EGO variants. Although the enlarged variant is significantly better than the reinforced one, the difference between them is much smaller than the differences towards the remaining algorithms. The second best group includes the repeated and the Rolling Tide variant of SMS-EGO, followed by the third group with the original RTEA and the enlarged random search. The poor performance of the RTEA can be explained by the expensive setting of our experiment. Naturally, EAs need a lot more function evaluations to reach good results. The repeated random search is worse than every other variant.

Table 2. Results of the ANOVA for the artificial experiment.

variable	sum Sq	p-value
algorithm	1.61	< 2e-16
function	10.96	< 2e-16
σ	0.71	< 2e-16
replication	0.03	0.0081

Table 4 displays the average runtimes of the seven different algorithms across all test functions. It shows that fitting the Kriging model to a larger number of observations variants results in a notable increase in modeling overhead. However, this can mostly be neglected in the expensive setting.

[4] As per the 3σ rule, 99.7% of normal distributed random numbers are within $\pm 3\sigma$.

Table 3. Post-hoc pairwise one-sided paired t-tests for the alternative that the method in the row reached a lower hypervolume than the one in the column.

	mbmo_enl	mbmo_reinf	mbmo_rt	mbmo_rep	ea_rt	rs_enl
mbmo_reinf	5.1e-05	-	-	-	-	-
mbmo_rt	< 2e-16	1.4e-14	-	-	-	-
mbmo_rep	< 2e-16	< 2e-16	2.8e-05	-	-	-
ea_rt	< 2e-16	< 2e-16	< 2e-16	< 2e-16	-	-
rs_enl	< 2e-16	< 2e-16	< 2e-16	< 2e-16	0.0086	-
rs_rep	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16	< 2e-16

Table 4. Average runtimes of the different algorithms.

algorithm	rs_rep	rs_enl	ea_rt	mbmo_rep	mbmo_rt	mbmo_reinf	mbmo_enl
runtime [sec]	1	3	7	699	8 721	37 327	53 965

4.3 Using the Nugget Effect

Although the enlarged strategy reaches the best hypervolume values, it cannot overcome its main disadvantage. Fig. 2 shows the Average Hausdorff distances between the tune and the test front for the enlarged and the reinforced variant. Since in the case of $\sigma = 0$ both the tune and the test evaluations return the same values, the distances are 0. For all higher σ–values we observe that the enlarged strategy has high distances and returns overly optimistic Pareto fronts. On the contrary, the reinforced strategy reaches essentially lower distances.

Besides the tune fronts, we can also look at the performance of model fronts. Therefore, we enable the fixed nugget effect in the Kriging model. Subsequently, we rerun the experiments adding different nugget effects $\lambda \in \{0.01, 0.1, 1\}$ to

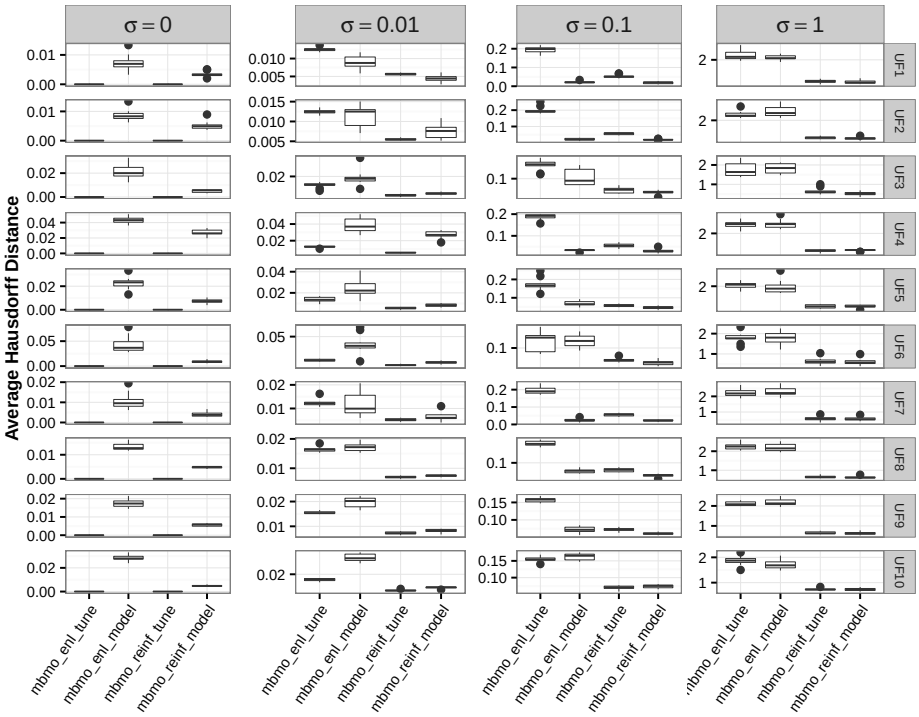

Fig. 2. Average Hausdorff distances for the enlarged and the reinforced SMS-EGO methods. Distances are calculated for the tune fronts with $\lambda = 10^{-8}$ and for the model fronts with $\lambda = 0.01$.

Table 5. Results of the ANOVA for the artificial experiment with added nugget effects (left table). Post-hoc pairwise one-sided paired t-tests for the alternative that the λ in the row reached a higher hypervolume than the one in the column. (right table).

variable	sum Sq	p-value
algorithm	0.39	< 2e-16
λ	0.09	< 2e-16
function	11.33	< 2e-16
σ	1.30	< 2e-16
replication	0.07	< 2e-16

λ	1e-08	0.01	0.1
0.01	0.21	-	-
0.1	5.6e-07	7.4e-06	-
1	< 2e-16	< 2e-16	< 2e-16

the Kriging models. The results of the corresponding ANOVA are reported in Table 5. The nugget effect has a significant influence on the hypervolume values. The post-hoc tests for the nugget effect show that performance deteriorates while using higher nugget effects. Only the smallest effect ($\lambda = 0.01$) reaches hypervolume values equal to those of the interpolating Kriging model. Thus, the smallest nugget effect can be used during optimization and for calculation of the final model front without deteriorating the hypervolume value.

The boxplots in Fig. 2 additionally show the distances for the model fronts. Naturally, the model fronts perform worse than the tune fronts in the deterministic case ($\sigma = 0$). This relationship flips with increasing noise, until in the case $\sigma = 0.1$ the model front is clearly superior for most test functions. For the reinforced strategy the usage of the model front does not improve the final Pareto front. Furthermore, in most cases the tune fronts of the reinforced strategy are better than the model fronts of the enlarged strategies.

5 Machine Learning Experiment

The second benchmark is a practical setting from machine learning. We consider the bi-objective minimization of the false-negative and the false-positive rate (FNR and FPR) in binary classification using SVMs with radial kernels.

5.1 Experimental Setup

In this setting each function evaluation corresponds to estimating the performance of a given hyperparameter setting for the SVM using a resampling strategy. While classical parameter tuning approaches would use a k-fold cross-validation for each setting, here a single 10% holdout is performed to reflect the noisy optimization. If needed, the noise handling strategy can perform multiple re-evaluations of a parameter setting, i.e., it can perform multiple holdout iterations. This is not equivalent to the usual cross-validation, but especially for the repeated strategies it is highly related.

As a general strategy for balancing FNR and FPR, class weighting can be used. Without loss of generality it is sufficient to adapt the weight ω for

the positive class. We optimize the cost parameter C and the inverse kernel each within a region-of-interest of $2^{[-15,15]}$, as well as the class weighting parameter ω within the interval $2^{[-7,7]}$, resulting in an input dimension $d = 3$. All parameters are optimized on a logarithmic scale. We run the experiments on several data sets (see Table 6), all of them are available on the machine learning platform OpenML [24]. For an unbiased comparison of the final Pareto fronts we perform tuning on 50% of the data points and leave the remaining 50% for calculation of the final test front. The same algorithms and further settings as described in section 4 are used.

Table 6. Description of the used data sets. #obs is the number of observations and #feats is the number of features in the respective data sets.

name	#obs	#feats
ada_agnostic	4 562	48
eeg-eye-state	14 980	14
kdd_JapaneseVowels	9 961	14
pendigits	10 992	16
phoneme	5 404	5
spambase	4 601	57
wind	6 574	14
waveform	5 000	40

5.2 Results

In Fig. 3 the hypervolume values of the resulting test fronts are displayed. In contrast to the artificial experiment, here Rolling Tide and repeated variant of SMS-EGO perform best. Both, the enlarged and reinforced variant reach lower values and perform even worse than the RTEA and the repeated random search. The same effect can be seen comparing the baselines:

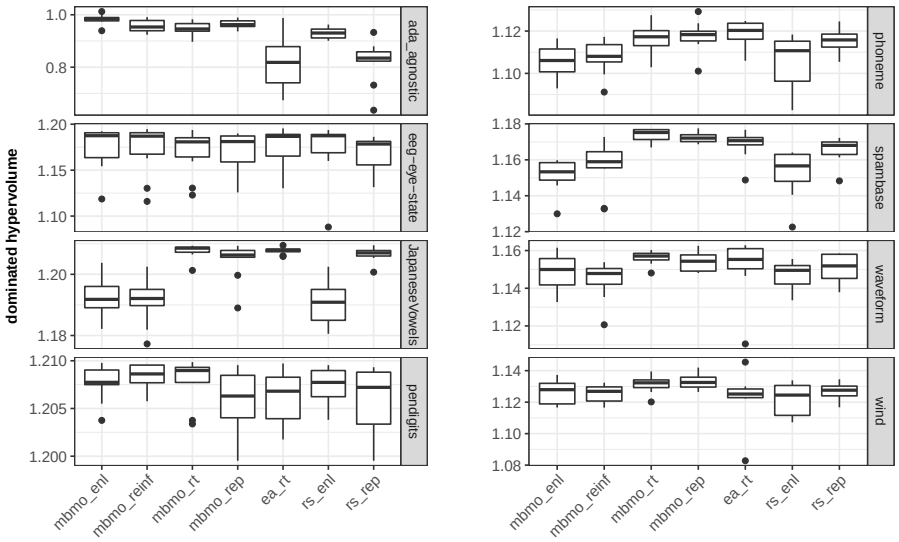

Fig. 3. Dominated hypervolume for the second experiment (reference point (1.1, 1.1)).

The enlarged random search is worse than the repeated one. Both the algorithm and the data set have a significant influence on the performance (Table 7). As in the artificial experiment, we can give a rather clear ranking of the algorithms (Table 8), although the differences are not as explicit as before. However, the ranking is rather different: the best algorithms of the artificial experiment are now on the second and third last ranks.

Table 7. Results of the ANOVA for the artificial experiment.

variable	sum Sq	p-value
algorithm	2.703e-4	$<$ 2e-16
data set	4.088e-4	$<$ 2e-16
replication	0.074e-4	0.266

6 Conclusion

In this paper, we took a first look at noise handling within MBMO algorithms. We proposed several variants and evaluated their performance with two experiments. In the first one, we used an artificial test set with homogeneous Gaussian noise. In the second one, from a machine learning context, the noise is unknown and likely does not follow any simple distribution, but is heterogeneous in $\mathcal{X}$.

The results of our two experiments are contradictory in terms of the best variant. In the first experiment, the enlarged and reinforced variant of SMS-EGO performed best. Thus, performing more than one evaluation per $\boldsymbol{x}$-setting is not necessary to guide the optimization. For a reliable estimation of the Pareto front one should either add a small nugget effect to the underlying Kriging model and use its estimated front, or invest some final evaluations to reinforce the final front. In the second experiment, all strategies using only a single evaluation for each setting performed poorly and even worse than a repeated random search. In this setting, the proposed Rolling Tide MBMO variant reached the best performance.

An intuitive explanation for the different behavior of our methods in the two experiments can be given by looking at the noise effect in the underlying Kriging models. The noise in the artificial experiment is homogeneous. In the Kriging model, the correlation between observations is modeled in the covariance matrix. Thus, the noise effect on individual points can be reduced by evaluating multiple blurry points together due to the smoothing effect of the model. Hence, exploring more points using the enlarged variant gives more information about the global

Table 8. Post-hoc pairwise one-sided paired t-tests for the alternative that the method in the row reached a lower hypervolume than the one in the column.

	mbmo_rt	mbmo_rep	ea_rt	rs_rep	mbmo_reinf	mbmo_enl
mbmo_rep	0.00211	-	-	-	-	-
ea_rt	0.00053	0.15213	-	-	-	-
rs_rep	1.5e-07	0.01358	0.07858	-	-	-
mbmo_reinf	1.3e-15	3.2e-14	3.8e-11	1.2e-10	-	-
mbmo_enl	$<$ 2e-16	6.5e-16	9.6e-14	3.0e-13	0.01164	-
rs_enl	9.2e-16	6.9e-15	1.7e-13	3.3e-12	0.00087	0.28380

structure of the response. On the contrary, the noise in the machine learning experiment is heterogeneous. Simply exploring more points in the $\mathcal{X}$ space cannot reduce the noise efficiently and fails to gain more information compared to a re-evaluation of the same point for a more reliable estimation of the observation.

In our future, work we will continue to investigate this topic with the aim to understand how the different strategies for choosing the re-evaluations are affected by different types of noise. Furthermore, we think that the smoothing effect of the surrogate model has a great potential in guiding a noisy optimizer. One possibility to improve this effect could be the integration of repeated evaluations into the model, instead of using mean values. But then the interpolating Kriging model cannot be used any longer. A reasonable alternative could be the so-called stochastic Kriging approach [25]. Instead of using a fixed λ, it could also be promising to tune λ during optimization. However, the results of this paper showed that low λ values always give the best results, independent of σ.

References

1. Aizawa, A.N., Wah, B.W.: Scheduling of genetic algorithms in a noisy environment. Evolutionary Computation 2(2), 97–122 (1994)
2. Bischl, B., Mersmann, O., Trautmann, H., Weihs, C.: Resampling methods for meta-model validation with recommendations for evolutionary computation. Evolutionary Computation 20(2), 249–275 (2012)
3. Bischl, B., Lang, M., Kotthoff, L., Schiffner, J., Richter, J., Studerus, E., Casalicchio, G., Jones, Z.M.: mlr: Machine learning in R. Journal of Machine Learning Research 17(170), 1–5 (2016)
4. Bischl, B., Lang, M., Mersmann, O., Rahnenführer, J., Weihs, C.: BatchJobs and BatchExperiments: Abstraction mechanisms for using R in batch environments. Journal of Statistical Software 64(11), 1–25 (2015)
5. Bischl, B., Richter, J., Bossek, J., Horn, D., Lang, M.: mlrMBO: A Toolbox for Model-Based Optimization of Expensive Black-Box Functions, `https://github.com/berndbischl/mlrMBO`
6. Breiderhoff, B., Bartz-Beielstein, T., Naujoks, B., Zaefferer, M., Fischbach, A., Flasch, O., Friese, M., Mersmann, O., Stork J.: Simulation and optimization of cyclone dust separators. In: Proc. 23. Workshop Computational Intelligence. pp. 177–195 (2013)
7. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
8. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. 27(8), 861–874 (2006)
9. Fieldsend, J.E., Everson, R.M.: The rolling tide evolutionary algorithm: A multi-objective optimizer for noisy optimization problems. IEEE Transactions on Evolutionary Computation 19(1), 103–117 (2015)
10. Hernández-Lobato, D., Hernández-Lobato, J.M., Shah, A., Adams, R.P.: Predictive entropy search for multi-objective bayesian optimization. arXiv preprint arXiv:1511.05467 (2015)
11. Horn, D., Bischl, B.: Multi-objective parameter configuration of machine learning algorithms using model-based optimization. IEEE Symposium on Computational Intelligence in Multicriteria Decision-Making (2016), (Accepted)

12. Horn, D., Demircioğlu, A., Bischl, B., Glasmachers, T., Weihs, C.: A comparative study on large scale kernelized support vector machines. Advances in Data Analysis and Classification pp. 1–17 (2016)
13. Horn, D., Wagner, T., Biermann, D., Weihs, C., Bischl, B.: Model-based multi-objective optimization: Taxonomy, multi-point proposal, toolbox and benchmark. In: International Conference on Evolutionary Multi-Criterion Optimization. pp. 64–78 (2015)
14. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments - a survey. IEEE Transactions on Evolutionary Computation 9(3), 303–317 (2005)
15. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. Journal of Global Optimization 13(4), 455–492 (1998)
16. Kleijnen, J.P.C.: White noise assumptions revisited: Regression metamodels and experimental designs in practice. In: Proceedings of the 38th Conference on Winter Simulation. pp. 107–117 (2006)
17. 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)
18. Knowles, J., Corne, D., Reynolds, A.: Noisy multiobjective optimization on a budget of 250 evaluations. In: International Conference on Evolutionary Multi-Criterion Optimization. pp. 36–50 (2009)
19. Koch, P., Wagner, T., Emmerich, M.T., Bck, T., Konen, W.: Efficient multi-criteria optimization on noisy machine learning problems. Applied Soft Computing 29, 357 – 370 (2015)
20. Ponweiser, W., Wagner, T., Biermann, D., Vincze, M.: Multiobjective optimization on a limited amount of evaluations using model-assisted $\mathcal{S}$-metric selection. In: Proc. 10th Int'l Conf. Parallel Problem Solving from Nature (PPSN). pp. 784–794 (2008)
21. Schiffner, J., Bischl, B., Lang, M., Richter, J., Jones, Z.M., Probst, P., Pfisterer, F., Gallo, M., Kirchhoff, D., Kühn, T., Thomas, J., Kotthoff, L.: mlr tutorial. CoRR abs/1609.06146 (2016)
22. Schutze, O., Esquivel, X., Lara, A., Coello, C.A.C.: Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. IEEE Transactions on Evolutionary Computation 16(4), 504–522 (2012)
23. Syberfeldt, A., Ng, A., John, R.I., Moore, P.: Evolutionary optimisation of noisy multi-objective problems using confidence-based dynamic resampling. European Journal of Operational Research 204(3), 533 – 544 (2010)
24. Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: Networked science in machine learning. SIGKDD Explor. Newsl. 15(2), 49–60 (2014)
25. Zhan, J., Ma, Y., Zhu, L.: Multiobjective simulation optimization using stochastic kriging. In: Proceedings of the 22nd International Conference on Industrial Engineering and Engineering Management 2015. pp. 81–91 (2016)
26. Zhang, Q., Zhou, A., Zhaoy, S., Suganthany, P.N., Liu, W., Tiwari, S.: Multiobjective optimization test instances for the cec 2009 special session and competition. Tech. rep., University of Essex and Nanyang Technological University (2008)
27. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation 1, 32–49 (2011)
28. Zuluaga, M., Krause, A., Püschel, M.: e-pal: An active learning approach to the multi-objective optimization problem. Journal of Machine Learning Research 17(104), 1–32 (2016)