A Ranking and Selection Strategy for Preference-based Evolutionary Multi-objective Optimization of Variable-Noise Problems

COIN Report Number 2016002

Florian Siegmund, Amos H.C. Ng
School of Engineering Science
University of Skövde
Högskolevägen, 54128 Skövde, Sweden
{florian.siegmund, amos.ng}@his.se

Kalyanmoy Deb, IEEE Fellow
Department of Electrical and Computer Engineering
Michigan State University
428 S. Shaw Lane, East Lansing, MI 48824, USA
kdeb@egr.msu.edu

***Abstract*—In simulation-based Evolutionary Multi-objective Optimization the number of simulation runs is very limited, since the complex simulation models require long execution times. With the help of preference information, the optimization result can be improved by guiding the optimization towards relevant areas in the objective space, for example with the R-NSGA-II algorithm [9], which uses a reference point specified by the decision maker. When stochastic systems are simulated, the uncertainty of the objective values might degrade the optimization performance. By sampling the solutions multiple times this uncertainty can be reduced. However, resampling methods reduce the overall number of evaluated solutions which potentially worsens the optimization result.**

In this article, a Dynamic Resampling strategy is proposed which identifies the solutions closest to the reference point which guides the population of the Evolutionary Algorithm. We apply a single-objective Ranking and Selection resampling algorithm in the selection step of R-NSGA-II, which considers the stochastic reference point distance and its variance to identify the best solutions. We propose and evaluate different ways to integrate the sampling allocation method into the Evolutionary Algorithm. On the one hand, the Dynamic Resampling algorithm is made adaptive to support the EA selection step, and it is customized to be used in the time-constrained optimization scenario. Furthermore, it is controlled by other resampling criteria, in the same way as other hybrid DR algorithms. On the other hand, R-NSGA-II is modified to rely more on the scalar reference point distance as fitness function. The results are evaluated on a benchmark problem with variable noise landscape.

I. Introduction

The goal of Preference-based Evolutionary Multi-objective Optimization is to provide the decision maker with a set of alternative solutions within a limited area of the objective space. If the optimization problem is a simulation of a practical, real-world optimization problem, the number of solution evaluations is usually limited [18]. Preference-based optimization allows to find a better result set of solutions in the preferred area within the available optimization time. In this way, a Preference-based EMO algorithm can find a non-dominated front which is closer to the true Pareto-front within the preferred area than the result front that can be attained by an algorithm which explores the whole Pareto-front. As a second goal, in order to provide the decision maker with a solution set which are genuine alternatives, Preference-based EMO needs to preserve a certain amount of diversity in the result population. Multi-objective evolutionary algorithms that can perform guided search with preference information are, for example, the R-NSGA-II algorithm [9], Visual Steering [30], and progressively approximated value functions [8].

Besides limited optimization time, Simulation-based Optimization entails the challenge of handling noisy objective functions [1], [2], [17], [32], [15]. To obtain an as realistic as possible simulation of the system behavior, stochastic system characteristics are often built into the simulation models. When running the stochastic simulation, this expresses itself in deviating result values. Therefore, if the simulation is run multiple times for a certain system configuration, the result value is slightly different for each simulation run.

If an evolutionary optimization algorithm is run without noise handling on a stochastic simulation optimization problem, the performance will degrade in comparison with the case if the true mean objective values would be known. The algorithm will have wrong knowledge about the solutions' quality. Two cases of misjudgment will occur. The algorithm will perceive bad solutions as good and select them into the next generation (Type II error). Good solutions might be assessed as inferior and might be discarded (Type I error). The performance can therefore be improved by increasing the knowledge of the algorithm about the solution quality.

Resampling is a way to reduce the uncertainty about objective values of solutions. Resampling algorithms evaluate solutions several times to obtain an approximation of the expected objective values. This allows EMO algorithms to make better selection decisions, but it comes with a cost. As the modeled systems are usually complex, they require long simulation times, which limits the number of available solution evaluations. The additional solution evaluations needed to reduce uncertainty of objective values are therefore not available for exploration of the objective space. This exploration vs. exploitation trade-off can be optimized, since the required knowledge about objective values varies between solutions. For example, in a dense, converged population which often occurs in Preference-based EMO, it is important to know the objective values well, whereas an algorithm which is about to explore the

objective space is not harmed much by noisy objective values. Therefore, a resampling strategy which samples the solutions carefully according to their resampling need, can help an EMO algorithm to achieve better results than a static resampling allocation. Such a strategy is called Dynamic Resampling, DR. In this article we propose new Dynamic Resampling strategies for Preference-based EMO. They are evaluated together with the Preference-based EMO algorithm Reference point-guided Non-domination Sorting Genetic Algorithm-II optimization algorithm (R-NSGA-II) [9], which uses reference points specified by the decision maker to guide the population towards the preferred area in the objective space.

The resampling need varies between solutions and can be calculated in many different ways [27], [28]. One approach is to assign more samples to solutions close to the Pareto-front. Since in real-world problems, the Pareto-front is not known, this can be achieved approximatively by assigning more samples to solutions as the optimization time progresses. Another approximation strategy is to assign more samples to solutions that dominate more other solutions, or are dominated by fewer other solutions, respectively. This is done by, for example, Confidence-based Dynamic Resampling and the MOPSA-EA algorithm [31], or Rank-based Dynamic Resampling [25], [26], [27], [28]. In our case, when preference information is available, increased sampling budget can be assigned to solutions close to a preferred area (i.e. in this article close to a R-NSGA-II reference point). This valuable information allows for DR algorithm with better performance to be designed. DR algorithms which use distance as a resampling criterion, i.e. the DDR algorithm [25], [26], [27], [28], and Progress-based DR algorithms [26], [27], [28] perform better than DR algorithm which do not have the distance information available. The second important approach is to assign samples in order to compare the objective values of solutions. EMO algorithms have a selection operator which determines if solutions will be part of the next population or be discarded. This approach is used by the MOPSA-EA algorithm [31] which compares pairs of solutions using Confidence-based DR (CDR) or the MOCBA and EA approach [5] (Multi-objective Optimal Computing Budget Allocation), which compares sets of solutions. Both approaches use a method called Ranking and Selection [12], and both algorithms show promising results.

In this article, we propose a Dynamic Resampling algorithm which uses a Ranking and Selection method to allocate samples for a single-objective fitness function. The R-NSGA-II algorithm uses the scalar reference point distance as secondary fitness criterion. The single-objective version OCBA of the MOCBA algorithm was proposed in [4] identifies the alternative design with the best performance value out of a set of candidate designs or configurations of a system. In [3], an algorithm variant was proposed which identifies the best m alternatives, called OCBA-m. This algorithm can be used to identify a set of solutions with the best reference point distances in R-NSGA-II. We show how this algorithm can be integrated with the R-NSGA-II algorithm to support it in the best possible way. We also propose a modification of the R-NSGA-II algorithm, in order to give a higher emphasis to the reference point distance as a fitness criterion, without losing control over the population diversity of the Evolutionary Algorithm. Similar to previously published hybrid Dynamic Resampling algorithms [25], [26], [27], [28], we show how the distance-based OCBA-m Dynamic Resampling algorithm can be controlled by other resampling criteria.

The paper is structured as follows. In Section II, different resampling techniques for EMO and for Preference-based EMO are reviewed. Section III explains the R-NSGA-II algorithm and an extension with scalar fitness function is proposed. On this basis, Section IV presents a new Ranking and Selection-based resampling algorithm and hybrid algorithm variants. Section VI shortly explains benchmark problems with variable noise landscape, which are used for evaluating the proposed resampling algorithms. In Section VII, numerical experiments on benchmark functions are performed and evaluated with performance metrics for preference-based multi-objective optimization. In Section VIII, conclusions are drawn and possible future work is pointed out.

II. Resampling Algorithms

In this section, Dynamic Resampling algorithms are reviewed as a reference for the Ranking and Selection Strategy proposed in this paper.

A. Generally applicable resampling algorithms

In this section resampling algorithms from the literature are described which do not rely on an external preference-information of the decision maker. However, they can be applied both on MOO algorithms which explore the Pareto-front and algorithms which explore a preferred part of the Pareto-front.

1) Static Resampling: Static Resampling assigns the same fixed number of samples to all solutions involved in an optimization run. It is popular among optimization practitioners since it requires a relatively small implementation effort. The disadvantage is that accurate knowledge of the objective vectors is not needed for all solutions: Not all solutions have objective values with high variability, and in the beginning of an optimization run the benefit of accurate values usually does not justify their cost. The degenerated form of Static Resampling with only one sample per solution achieves competitive results [26], since the optimization has more solutions available for exploring the search space. But with the exception of this special case, Static Resampling is inferior to many Dynamic Resampling algorithms.

2) Time-based Dynamic Resampling: Time-based DR [25] is a generally applicable dynamic resampling algorithm which can be used on any optimization algorithm. It assumes that the need for accurate knowledge of objective values increases towards the end of the optimization run. Time-based DR makes only one sampling decision for each solution (one-shot allocation) according to the elapsed time. However, if a solution survives into the next generation of an evolutionary algorithm the decision is made again.

3) Rank-based Dynamic Resampling: Rank-based DR [25] is a dynamic resampling algorithm which can be used on any multi-objective optimization algorithm. It measures the level of non-dominance of a solution and assigns more samples to solutions with lower Pareto-rank. It can have a parameter allowing only the solutions with a Pareto-rank of n or less to be allocated additional samples (RankMaxN-based DR) [26]. Rank-based DR can be used in a hybrid version as Rank-Time-based DR [26]. Rank-based Dynamic Resampling shows good results on the (R-)NSGA-II algorithm which is expected since the dominance relation is its dominant fitness measure. Since after each added sample, the dominance relations in the

population of an MOO algorithm changes Rank-based DR is performed sequentially.

4) Standard Error Dynamic Resampling: The SEDR resampling algorithm assigns the number of samples to a solution based on its objective variance, for each solution individually. It was proposed in [10], as a single-objective DR algorithm. Here, a multi-objective version is described. SEDR continuously adds samples to a solution until the standard error of the mean falls below a user-defined threshold $\mathrm{se}_n(s) < \mathrm{se}_{th}$.

In order to formulate the termination criterion for multiple objectives we calculate an aggregation of the standard error of the mean of all objectives [25] and push it below the threshold se_{th}. We propose to use the maximum standard error of all objective values.

$$\overline{\mathrm{se}}_n(s) := \max_i \mathrm{se}_i^n(s) < \mathrm{se}_{th}. \quad (1)$$

By using the maximum of all $\mathrm{se}_i^n(s),\ i = 1 \ldots H$, we guarantee that the termination criterion is satisfied for all objectives.

B. Preference-based resampling algorithms

The DR algorithms in this section use preference information specified by a decision maker to define their sampling allocation. In this paper, the preference information is specified as reference points in the objective space for R-NSGA-II. These resampling algorithms are described here, since we propose hybrid extensions of SEDR which are based on and build in the same way as the algorithms below.

1) Progress-based Dynamic Resampling: Progress-based DR is described in [26]. Its premise is that if the R-NSGA-II population progress towards a reference point slows down the population members are concentrated in a small area. In this situation the algorithm will benefit from accurate knowledge about the objective vectors. The disadvantage of this strategy becomes apparent in a situation of premature convergence. If a population based optimization algorithm gets stuck in a local optimum, it has been shown that more objective uncertainty, and not less, can be helpful to escape the local optimum.

2) Distance-based Dynamic Resampling, DDR: DDR was proposed in [25] and it assigns samples to a solution based on its distance to a reference point. Since a reference point in the objective space can be unattainable by the optimization, the hybrid DDR algorithm combines the distance allocation with a progress measure. Such unattainable reference points are common, since the decision maker usually will pick a reference point out of an optimistic area in the objective space. Therefore, if the optimization convergence slows down, the progress measure will indicate that the best attainable solutions with the closest distance to the reference point have been found and the sampling allocation is increased to b_{max} for those solutions. In addition, DDR also considers the elapsed optimization time, in order to handle situations of premature convergence in a local Pareto-front where the progress measure causes unwanted high number of samples [25], or if a satisfying convergence never is achieved during the optimization runtime.

3) Distance-Rank-based Dynamic Resampling, DR2: DR2 was proposed in [26] and it combines Distance-based DR and Rank-based DR [25]. Since DDR is involved, DR2 is a preference-based DR algorithm. Due to the promising results of Rank-Time-based DR in [26] we combined the rank criterion with the distance resampling criterion and were able to show its superiority for complex optimization problems.

C. Hybrid SEDR Strategies

Similar to the hybrid DR algorithms proposed in [26], MO-SEDR can be combined with other resampling criteria [28]. We proposed hybrid SEDR algorithms which allocate samples both based on the noise level of a solution s, and on another criterion $p \in [0, 1]$, or on multiple other criteria. Different Hybrid Multi-objective SEDR variants can be defined, as described in the following sections.

1) Standard Error Time-based DR: The SEDR threshold $\mathrm{se}_{th}(p)$, or the min./max. number of samples $b_{max}(p)$ and $b_{min}(p)$, or both, are controlled by the elapsed optimization time.

2) Standard Error Rank-Time-based DR: The SEDR parameters $\mathrm{se}_{th}(s, p)$, or $b_{max}(s, p)$, $b_{min}(s, p)$ are controlled both by the Pareto-rank of a solution s and the elapsed optimization time, according to the sampling allocation of the RankMaxN-Time-based DR algorithm [26].

3) Standard Error Distance-based DR: In Standard Error Distance-based Dynamic Resampling, or SE-DDR, $\mathrm{se}_{th}(s, p)$, or $b_{max}(s, p)$, $b_{min}(s, p)$ are adapted according to a solution's distance to a reference point, as in the Distance-based Dynamic Resampling (DDR) algorithm [26]. As mentioned above, DDR also considers population progress and elapsed optimization time as sampling criteria.

4) Standard Error Distance-Rank-based DR: Standard Error Distance-Rank-based Dynamic Resampling, which we call SEDR2, uses the allocation of the Distance-Rank-based Dynamic Resampling algorithm (DR2) [26] to control $\mathrm{se}_{th}(s, p)$ or $b_{max}(s, p)$, $b_{min}(s, p)$, thereby combining five resampling criteria: Standard error and Pareto-rank of a solution, distance and progress towards a R-NSGA-II reference point, and the elapsed optimization time.

III. The R-NSGA-II algorithm

For evaluation of the Dynamic Resampling algorithms in this paper, the preference-based multi-objective Evolutionary Algorithm R-NSGA-II (Reference point-based Non-domination Sorting Genetic Algorithm II) is used, as well as an R-NSGA-II algorithm variant.

The R-NSGA-II algorithm was proposed in [9] and is based on the NSGA-II algorithm [6]. NSGA-II is a population-based Evolutionary Algorithm for optimization problems with multiple objectives. In the selection step, NSGA-II sorts the population and offspring into multiple fronts and selects front by front into the next population, as long as the fronts can be selected as a whole. If the next front can only be selected partially, NSGA-II uses a clustering method called Crowding Distance to determine which solutions of the front shall be selected, in a way that increases population diversity. The parental selection for offspring generation follows this fitness hierarchy.

R-NSGA-II replaces the secondary fitness criterion Crowding Distance by the distance to user-specified reference point(s) R, which helps the Evolutionary Algorithm to focus on a preferred region instead. An example for the R-NSGA-II selection step in the objective space is depicted in Figure 1. In order to not lose too much population diversity, a clustering mechanism with a minimum objective vector distance ϵ is used. The same fitness hierarchy, i.e., dominance first, then reference point distance, is used for parental selection.

A. Extended R-NSGA-II - Scalar fitness

In this paper, an extended variant of the R-NSGA-II algorithm is proposed, which which is designed according to two objectives. In this paper, we propose a Dynamic Resampling algorithm which only works on a scalar fitness function. We therefore propose an R-NSGA-II extension where the scalar fitness criterion is used exclusively during a large part of the algorithm runtime. This distance-based fitness function supports a second design objective. It can speed-up the convergence towards the preferred area in the objective space. We call this modification Delayed Pareto-fitness, and the algorithm variant DPF-R-NSGA-II. This method works in a way that, until a certain point of the elapsed optimization runtime, fitness comparisons are done only based on the distance to the reference point(s), which otherwise is the secondary fitness criterion of R-NSGA-II (Figure 2). In this way, the optimization algorithm does not aim to cover a whole front and can thereby converge faster towards the preferred area. After a certain point in time has been reached, the normal operation mode of R-NSGA-II is resumed. This guarantees a diverse and focused final result front.

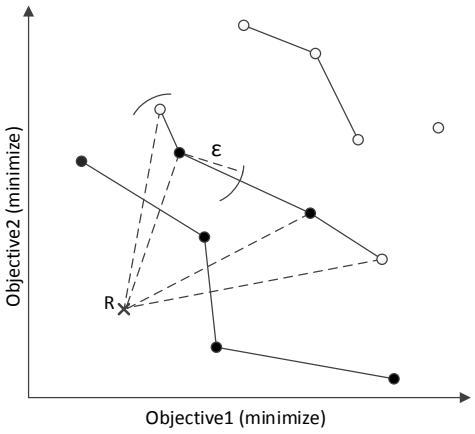

Fig. 1. R-NSGA-II algorithm – Example for selection step in objective space. The population size is $|P| = 6$. Four solutions are selected our of the first front, and two out of the second front.

Our results on the bi-objective ZDT benchmark function suite with deterministic objective functions [34] have shown that R-NSGA-II in the standard configuration of [9] with population size 50 will first focus on towards the Ideal Point [24], then the population will explore the Pareto-front, and first then it will focus towards the reference point of R-NSGA-II. This leads to a delay in convergence towards the reference point. Our experience shows that this effect is even stronger for optimization problems with noisy objective functions.

An algorithm similar to DPF-R-NSGA-II has been proposed in [16]. This algorithm compares two solutions s_1 and s_2 by the Pareto-dominance relation, if they are within a certain region around the reference point R with radius $d_{min} + \delta$ to R in the current population. d_{min} is the current best distance to R, and δ a user defined parameter. If both s_1 and s_2 are outside the region, then they are compared by their reference point distance only. Solutions inside of the region are preferred to solutions outside. Since d_{min} is continuously updated as the population progresses, the are comparisons by dominance and distance in every generation of the EA. Since we want to employ a Dynamic Resampling algorithm which works on a scalar fitness function, we designed DPF-R-NSGA-II in a way so that the scalar fitness criterion is used exclusively during a large part of the algorithm runtime. The extension will be described in the next subsection.

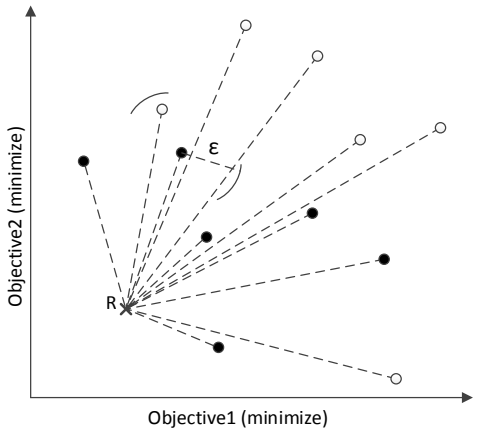

Fig. 2. R-NSGA-II algorithm with delayed Pareto-fitness (DPF-R-NSGA-II) – Example for selection step in objective space in the same scenario as in Figure 1. The distances from the objective vectors to the reference point R are compared and the best $|P| = 6$ solutions are selected.

B. Delayed Pareto-fitness

The selection step of the original R-NSGA-II algorithm in a two-dimensional objective space was shown in Figure 1. As mentioned, the predominance of the Pareto-fitness in the algorithm (the non-domination sorting has a higher priority than the reference point distance) can lead to that the convergence is delayed. The algorithm loses times when converging towards the Ideal Point or while exploring the whole Pareto-front. The distance-based fitness function in DPF-R-NSGA-II can speed-up the convergence. However, an algorithm which only uses a scalar fitness criterion cannot maintain diversity in the population. We therefore include the R-NSGA-II ϵ-based clustering method in DPF-R-NSGA-II. The solutions are first sorted based on their reference point distance. The solution with the lowest reference point distance is selected, and all other solutions in its ϵ-vicinity in the objective space are added to a cluster of solutions that are not considered for selection. The same procedure is done repeatedly for the remaining solutions, until $|P|$ solutions outside of each others ϵ-vicinity have been selected. In the case that less than $|P|$ solutions have been selected, the process is restarted with the solutions that have not been considered for selection.

The Pareto-dominance has an even stronger effect on population diversity than the ϵ-clustering method, and therefore the standard operation mode of R-NSGA-II is started as soon as a certain time-threshold θ_t is reached. Also, the standard mode with Pareto-dominance and reference point distance is activated as soon as a solution is found that dominates the reference point, and if the euclidean distance metric is used. This means that the reference point is attainable, and a fitness function based on the euclidean distance to the reference point alone does not work for attainable reference points.

The time threshold control of DPF-R-NSGA-II can be replaced by a distance-based control. For example, a progress measurement can be used and the operation mode can be switched back and forth adaptively if the progress falls below a threshold, $p < \theta_p$. Another example would be to use an

adaptive control similar to the distance-progress-time dynamic resampling allocation function, $x_{s_m}^{DDR} < \theta_d$, where s_m is the closest solution to the reference point in the population, $d_m = \min_k\{d_k\}$.

An Evolutionary Algorithm using a only a distance-based fitness function but no diversity preservation mechanism was done by [30] and is called Visual Steering. They guide the search towards an arbitrary attractor point in the objective space. However, their guided search by attractor point is not using Pareto-dominance which leads to the described effect, that for attainable reference points the best solutions cannot be found. Instead, an accumulation of solutions around the reference point is found as result set.

IV. Dynamic Distance-based Sampling Allocation

In the extended R-NSGA-II, during the major part of the runtime, the fitness value is the distance to a reference point. Therefore, during this time with this operational mode, DPF-R-NSGA-II works as a single-objective optimization algorithm (except for the ϵ-clustering mechanism. Out of the solution set $S = P \cup O$ of parents and offspring, the best $m = |P|$ solutions are selected. This means we need to partition S into the set of the best m solutions, S_m and the rest $S \setminus S_m$. This is a classic Ranking and Selection problem [12], and we try to solve it with the OCBA-m algorithm (Optimal Computing Budget Allocation) described in [3], which allocates the samples in a way to make the best decision which solutions have a fitness above and below a threshold c. Or in other words, it aims at increasing the probability for correct selection according to the scalar distance-based fitness value. We call this algorithm D-OCBA-m.

A. Distance-based OCBA-m (D-OCBA-m)

Since we want to do an optimal selection sampling to support a single objective evolutionary algorithm we can make use of existing sampling strategies. The hypervolume-based evolutionary algorithm tested in this paper generates a set of offspring solutions and selects a subset of these solutions into the next generation. There is a sampling algorithm that has been developed for this purpose: The OCBA-m algorithm [3]. OCBA-m identifies the top m solutions of a given solution set based on their scalar fitness value. It chooses a threshold c based on the current knowledge about the solutions so that the best m solutions have a higher fitness value than c. These best m solutions are selected and all solutions with a lower fitness value than m are discarded. The sampling budget for a solution is then assigned based on the difference of the sampling mean of the solution fitness to c, and on the variance of the solution. It should be noted that OCBA-m assumes all fitness values to be non-correlated [13], [14]. For our purpose, we use the distance to R as solution fitness.

Since this paper focuses on optimization with a limited time-budget, our experience shows that is is beneficial to limit the maximum number of samples by b_{max}, as we have done in this paper. In the original algorithm specification, there is no upper limit for the maximum number of samples per solution. This led to sampling allocations up to five times as high as b_{max} which we use for other dynamic resampling algorithms. In our previous publications, we found that the trade-off between problem exploration and exploitation within the limited time budget limits the number of samples which can be spent on one solution in a meaningful way. There is a diminishing marginal utility in adding more samples to a solution, since this means that the optimization process has less samples available to explore the search space, leading to worse optimization results.

In the following, the OCBA-m algorithm is described in detail with slight notation variation in comparison to the original formulation. At first, the following notation is introduced in Table I.

TABLE I. Notation for OCBA-m applied to the distance-based selection problem, called D-OCBA-m.

r_{s_k}	Reference point closest to solution s_k.
$\bar{d}_k$	Mean of the distance of the objective vector of solution s_k to reference point $r_{s_k}, d_k = d(s_k, r_{s_k})$.
σ_k	Standard deviation of stochastic variable d_k.
N_k	Number of samples for solution k.
$\bar{\sigma}_k$	Standard deviation of the mean $\bar{d}_k$. $\bar{\sigma}_k = \frac{\sigma_k}{\sqrt{N_k}}$.
n_0	Minimum number of samples for all solutions.
N_{total}	The overall number of samplings that can be executed.
S_m	The subset containing the best m solutions.

As objective function to be maximized, the probability of correct selection is chosen. This means that all solutions in the set S_m have a distance value smaller than c, and all solutions in the complementary set have a fitness value higher than c. The problem is formulated as follows in Equation 2.

$$\begin{aligned} \max_{N_1 \dots N_{|S|}} & \prod_{s_k \in S_m} P\left\{\bar{d}_k \le c\right\} \prod_{s_k \notin S_m} P\left\{\bar{d}_k \ge c\right\} \\ & \text{where } c = \frac{\bar{\sigma}_{k_{m+1}} \bar{d}_{k_m} + \bar{\sigma}_{k_m} \bar{d}_{k_{m+1}}}{\bar{\sigma}_{k_m} + \bar{\sigma}_{k_{m+1}}} \\ \text{s.t.} \quad & \sum_{k=1}^{|S|} N_k \le N_{total}, \\ & N_k \in \mathbb{N}, n_0 \le N_k \le b_{max}, k = 1 \dots |S| \end{aligned} \tag{2}$$

In Equation 2, k_m is the index of the m-th best solution. This means that in the list of solutions sorted by reference point distance, there are $m-1$ solutions that are closer to r than for solution m. The index k_{m+1} belongs to the next best solution in the ordered list. This solution and all other worse solutions are not selected into the next population.

With Karush-Kuhn-Tucker the allocation is optimal if the following holds [3] for all solutions i, j (Equation 3).

$$\frac{N_i}{N_j} = \left(\frac{\sigma_i / \bar{\delta}_i}{\sigma_j / \bar{\delta}_j}\right)^2, \quad \text{where } \bar{\delta}_i = \bar{d}_i - c. \tag{3}$$

This allocation can be achieved by selecting one solution arbitrarily (let it be solution s) and putting all other in relation to it, as in Equation 4.

$$\beta_i = \frac{N_i}{N_s}, i = 1 \dots k. \tag{4}$$

The β_i can be calculated using Equation 3. The allocation for solution i, with respect to the problem constraint $\sum_{k=1}^{|S|} N_k \le N_{total}$, can now be calculated approximately as in Equation 5. This allocation will most probably not be optimal, since δ_i and σ_i can only be estimated and are not known accurately. This allocation is done based on the n_0

initial samples for each solution. After the sampling based on the allocation in Equation 4 is performed, a repeated allocation would yield very different new N_i. N_{total} needs to be chosen as $N_{total} > |S|n_0$, in order to allow the minimum required allocation n_0 for all solutions.

$$N_i = \left\lfloor \frac{\beta_i}{\sum_j \beta_j} N_{total} \right\rfloor \tag{5}$$

Each N_i is assigned its correct share of N_{total}: $N_i = \lfloor \frac{\beta_i}{\sum_j \beta_j} N_{total} \rfloor = \lfloor \frac{\frac{N_i}{N_s}}{\sum_j \frac{N_j}{N_s}} N_{total} \rfloor = \lfloor \frac{N_i}{\sum_j N_j} N_{total} \rfloor$.

B. Iterative D-OCBA-m procedure

On all occasions where the mean $\bar{d}_k$ or the standard deviation σ_k of a solution is required, we approximate them by the sample mean $\mu_n(d_k)$ and the sample standard deviation $\sigma_n(d_k)$. The new notation is given in Table II. Since the information about a solution becomes more accurate with every sample drawn, the budget allocation should be recalculated after a few new samplings have been executed. Based on the better knowledge about the solutions better allocation decisions can be made. Therefore, D-OCBA-m is executed in an iterative manner. It is a sequential sampling strategy. Asymptotically, the allocation will be optimal. For the variance estimation to be good enough, the authors in [3] suggest $n_0 \geq 5$.

TABLE II. ADDITIONAL NOTATION FOR DISTANCE-BASED OCBA-M FOR ITS ITERATIVE APPLICATION.

$\mu_n(d_k)$	Sample mean of the distance of the objective vector of solution s_k to reference point r_{s_k}.
$\sigma_n(d_k)$	Sample standard deviation of d_k.
N_{max}	The current maximum budget that can be assigned to all solutions, i.e. $\sum_{k=1}^{\lvert S\rvert} N_k \leq N_{max}$. N_{max} is increased iteratively.
Δ	N_{max} is increased iteratively by Δ.
δ	Maximum additional samples per solution and iteration.
ν	Iteration counter.

As distance function the Achievement Scalarization Function is used. The sample mean of the distance $\mu_n(d_k)$ is calculated as $\bar{d}_k \approx \mu_n(d_k) = \mu_n(d(s_k, r_{s_k})) = \delta_{ASF}(\mu_n(F(s_k)), r_{s_k}) = \max_{i=1}^{H}\{\mu_n(F_i(s_k)) - r_{ski}\}$. The sample standard deviation $\sigma_n(d_k)$ is based on $\mu_n(d_k)$: $\sigma_k^2 \approx \sigma_n^2(d_k) = \frac{1}{n-1}\sum_{j=1}^{n}(d_k^j - \mu_n(d_k))^2$, where d_k^j is the j-th sample of d_k.

In every iteration ν, only a few new samples are drawn based on the following rule: $\min\left\{\delta, \max\left\{N_k^{\nu+1} - N_k^{\nu}, 0\right\}\right\}$. A maximum of δ new samples are executed and if the new budget $N_k^{\nu+1}$ is smaller than the old budget N_k^{ν} no additional samples are executed. For this paper, the sampling algorithm is implemented in the way that if solution k has already been sampled before more than or exactly $N_k^{\nu} + \delta$ times, no new samplings are executed. This is a common case since solutions that are selected into the next population are sampled again in the next iteration. Therefore, the overall number of executed samples by the sampling algorithm can change with every iteration of the evolutionary algorithm.

If the fitness values can be calculated fast δ should be chosen small (e.g. $\delta = 1$). Many iterations will be executed and a good budget allocation will be the result. If the fitness values are expensive to calculate δ should be set higher. The algorithm runtime will decrease but the sampling allocation will not be as accurate anymore. The decision for a certain δ is dependent on the evaluation time of the function to be optimized. If the algorithm running time is very short in relation to the function execution time (e.g. for simulation optimization in our case) a small δ can be chosen. The sequential procedure is written down in Algorithm 1.

Algorithm 1 OCBA-m Sequential Sampling Procedure based on [3], [4], and [12].

1: Generate initial population and sample each solution n_0 times.
2: Calculate sampling allocation $N_k = \left\lfloor \frac{\beta_k}{\sum_j \beta_j} N_{max} \right\rfloor$, $k = 1, \ldots, |S|$, as in Equation 5.
3: Assign each solution the additional budget $\min\left\{\delta, \max\left\{0, N_k^{\nu+1} - N_k^{\nu}\right\}\right\}$.
4: Perform the sampling. If the sum of all assigned budget $\geq N_{total}$, then sample only the solutions with the highest β, until N_{total} is reached.
5: Increase N_{max} by Δ.
6: Stop if the sum of all assigned budget $\geq N_{total}$, otherwise go to step 2.

In this paper, a variant of Algorithm 1 is used. We allow to exceed N_{total} slightly in the last iteration of the algorithm to allow sampling according to the sampling allocation (remove the condition in Step 4).

C. Dynamic-Best Sampling vs. Selection Sampling

We propose an alternative to the Selection Sampling procedure that the original OCBA-m was designed as: Dynamic-Best Sampling, abbr. DB Sampling. Similar to MOCBA [5], which identifies the non-dominated solutions, this variant of the OCBA-m algorithm identifies the best, or the better solutions, according to the reference point distance. Like for the MOCBA sampling algorithm, this set of identified best solutions does not need to be the same as the set of selected solutions by the EMO algorithm. For the Dynamic-Best Sampling OCBA-m variant, less solutions than the selected solutions are identified. The reason for this is that the Selection Sampling OCBA-m with $m = |P|/2$ will spend a lot of sampling budget on the other half of the solutions which will be discarded by the evolutionary algorithm during the selection process. Usually, the most samples are allocated to solutions which are almost inside or outside of the identified set S_m. By keeping m below $|P|/2$, those solutions are selected into the next population, and the optimization algorithm can still make use of those solutions and the invested samples. For $m = 1$, we call this algorithm variant Best Sampling.

During the optimization runtime, we change the size of the set of best solutions S_m which shall be identified. In the beginning, we only identify the best solution by resampling. Later during the runtime, we identify the set of m_{max} solutions. The number of best identified solutions m_p is set dynamically as in Equation 6. We set m_{max} to $m_{max} = |P|/2$, and $m_{min} = 1$.

$$m_p = \lfloor (1-p)(m_{max} - m_{min}) + m_{min}) \rceil . \tag{6}$$

$p \in [0, 1]$ in Equation 6 can be set as the elapsed optimization time as in Section II-A2, or as the reference point distance as in Section II-B2.

V. Hybrid OCBA-m

In [10], [23], [25], [26], [27], and [28], it has been shown that it is beneficial for the algorithm performance if the dynamic resampling algorithm makes use of additional resampling criteria. The sampling allocation level of the D-OCBA-m procedure can be controlled by these criteria. For example, this could be the elapsed optimization time, or the distance to the Pareto-front, approximated by the distance to a user-specified reference point, as in section II, or the (objective-wise) distance to the Ideal Point [11].

The additional criterion or combination of criteria is integrated into the OCBA-m procedure in the following way. N_{total} is made adaptive by specifying two quantities, $\bar{b}_{min}$ and $\bar{b}_{max}$, which determine the initial and final number of samples. The quantities specify a number of samples per solution and are multiplied by the number of considered solutions $|S|$, as in Equation 7. $|S|\bar{b}_{min}$ is the minimum value for N_{total} and $|S|\bar{b}_{max}$ the maximum.

$$N^p_{total} = \left\lfloor |S|(p(\bar{b}_{max} - \bar{b}_{min}) + \bar{b}_{min}) \right\rceil . \tag{7}$$

$\bar{b}_{min}$ should be chosen as $\bar{b}_{min} > n_0 \geq b_{min}$, and $\bar{b}_{max}$ as $\bar{b}_{max} \leq b_{max}$. It shall be noted that the number of samples for each individual solution is not limited by these quantities. The individual number of samples is limited by b_{max}, b_{min}, and n_0. The parameter p is set based on another resampling criterion, like the elapsed optimization time, distance or progress to a reference point, or combinations thereof, which makes the D-OCBA-m procedure a hybrid dynamic resampling algorithm. Those different hybrid D-OCBA-m algorithms are described in the following subsections. In addition, a different way to control the D-OCBA-m allocation based on the Pareto-rank of solutions is presented, which can be combined with the adaptive control in Equation 7.

A. Time-based D-OCBA-m

The additional criterion p is set as the elapsed optimization time.n_0 is usually chosen as $n_0 \geq 2$, in order to be able to calculate a standard deviation. This leads to that a lot of samples are spent on bad solutions which are discarded anyways. In order to avoid this waste of samples, b_{max} and thereby n_0 and b_{min} are reduced to 1, when the time-based allocation $b_s = \min\{b_{max}, \lfloor p_t(s)(b_{max} - b_{min} + 1)\rfloor + b_{min}\}$ indicates a sample value of 1. This way of saving sampling budget was proposed for the hybrid MO-SEDR variants in [28].

B. Rank-Time-based D-OCBA-m

For the Rank-Time-based D-OCBA-m, we propose to make use of the maximum Pareto-rank parameter of RankMaxN-Time-based Resampling Solutions with a rank of n or worse are assigned one sample only, and are not considered for Dynamic Resampling by the OCBA-m procedure. The RankMaxN allocation formula, proposed in [27], is given in Equation 8. The Rank-based control can be combined with the time-based control of D-OCBA-m, resulting in the RankMaxN-Time-based D-OCBA-m algorithm, or RnT-D-OCBA-m. It shall be noted that, despite the name, it is not the RankMaxN-based DR algorithm [27] that is used to control D-OCBA-m, because the Pareto-rank information is used in a different way.

$$x^{Rn}_s = 1 - \left(\frac{\min\{n, R_s\} - 1}{\min\{n, R_{max}\} - 1} \right)^a . \tag{8}$$

C. DDR-D-OCBA-m

The distance to a reference point in the objective space can be used to control D-OCBA-m. For the distance-based allocation, the reference point distance of the solution with index k_m is used, i.e. the m-th best solution according to the reference point distance. Since it is desirable to also include the progress to the reference point and the elapsed optimization time, we use the allocation function of the DDR algorithm [25] to control OCBA, $p = x^{DDR}_{s_{km}}$.

D. DR2-D-OCBA-m

This algorithm combines the DDR-D-OCBA-m with the Rank-based control in Formula 8. According to the name of the Distance-Rank-based Dynamic Resampling algorithm [27], DR2, we call this dynamic resampling algorithm DR2-D-OCBA-m. The distance indicator the reference point distance of the solution with index k_m is used, i.e. the m-th best solution according to the reference point distance. $p = x^{DR2}_{s_{km}}$. As for Rank-Time-based D-OCBA-m in section V-B, it shall be noted that, despite the name, it is not the DR2 algorithm [26] that is used to control D-OCBA-m, because the Pareto-rank information is used in a different way.

VI. Optimization Problems

Variance-based Dynamic Resampling algorithms have an advantage over resampling algorithms not considering objective variance if they are used in the optimization of functions with variable noise. This function property is called Noise Landscape [10].

In this paper, we evaluate the variance-based DR algorithms on three different variants of the ZDT4 benchmark function from the ZDT benchmark problem suite [34], which we proposed in [28]. In the original version the function output of these functions is deterministic. In order to create noisy variants, we add a variable noise landscape on the output values. We use relative noise, i.e. the general noise level is given in percent of the objective ranges. We designed the noise landscapes to change the noise level depending on the quality of the solutions. In other words, a solution with objective values closer to the Pareto-front than other solutions will have a different noise level. The ZDT functions have a component which controls this distance. It is called g-function, and we use its normalized value in Equation 9 to control the noise level.

$$l(s) = (g(s) - 1)/(g_{max}(s) - 1). \tag{9}$$

We perform experiments on two different noise landscapes which can be used on all ZDT benchmark functions. A noise landscape with changing noise level for the ZDT functions can be created as in Equation 10. The hills and valleys are created by the sine function and we therefore call it trigonometric noise landscape.

$$L_{trig}(s) = -(1 - L_{min})|\sin(N\pi l(s) - \phi)|^a + 1. \tag{10}$$

A graph of the trigonometric noise level function is shown in Figure 3. The challenge of this landscape are its many hills and valleys, similar to the Cosinusoidal noise landscape in [10]. $N \in \mathbb{N}$ allows to specify the number of peaks, while $a \geq 1$ influences the width of the peaks. By changing the phase offset ϕ, a low or high noise level can be set at the Pareto-front.

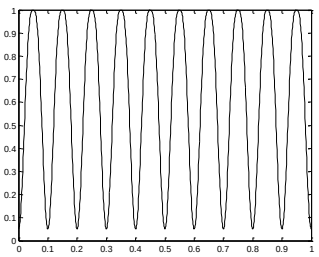

Fig. 3. This Figure shows the trigonometrical variable noise level $L_{trig}(s)$ based on the g-function in the ZDT benchmark problem suite. The parameters are set to $N = 10$, $a = 3$, $\phi = \pi/2$, and $L_{min} = 0.05$.

The variable noise landscape for the ZDT functions is then created by multiplying the original fixed standard deviation of the objective functions, σ_1, σ_2, with the variable noise level L, as in Equation 11.

$$\begin{pmatrix} \mathcal{N}(0, \sigma_1^v(s)) \\ \mathcal{N}(0, \sigma_2^v(s)) \end{pmatrix} = \begin{pmatrix} \mathcal{N}(0, L(s)\sigma_1) \\ \mathcal{N}(0, L(s)\sigma_2) \end{pmatrix} \tag{11}$$

In the following section, EMO algorithm performance is evaluated on ZDT functions with the variable noise landscape described here.

VII. Numerical Experiments

In this section, the described resampling techniques in combination with R-NSGA-II are evaluated on benchmark functions with different noise landscapes. To facilitate comparison, the experiments are grouped in experiments where no preference information is used for resampling and experiments where the resampling algorithm uses information about the distance to a reference point R defined for R-NSGA-II. The reference point is chosen to be unattainable in all experiments. The definition of an unattainable reference point is that it is not on the Pareto-front and no solution can be found that dominates it [21]. This case of pessimistic or attainable reference points is not considered in this article. We refer to the feature of R-NSGA-II that allows the interactive adaption of a pessimistic reference point, that has been identified as an attainable one during the optimization process, and to move it forward to turn it into an unattainable point. A Distance-based Dynamic Resampling algorithm for attainable reference points has been proposed in [25]. Future work will cover a detailed evaluation of Dynamic Resampling algorithms for attainable reference points.

A. Problem settings

For the ZDT4 function with invariant noise level, the relative added noise (20%) is $(\mathcal{N}(0, 0.2), \mathcal{N}(0, 20))$, as the relevant objective ranges are $[0, 1] \times [0, 100]$). For the optimization problems with variable noise landscape, the objective standard deviations are adjusted by the noise level $L(s)$ as in Equation 11. In the following, this problem is called ZDT4-20%.

The phase offset ϕ of the trigonometric noise landscape function is set to $\phi = \pi/2$, the number of peaks to $N = 10$, and the peak-width parameter a to 3, for all functions. The minimum noise level is set to $L_{min} = 0.05$. The reference point is chosen as $R = (0.05, 0.5)$.

B. Algorithm parameters

The limited simulation budget is chosen as 10,000 solution replications. R-NSGA-II is run with population size $|P| = 50$, a crossover rate of $p_c = 0.8$, SBX crossover operator with $\eta_c = 2$, Mutation probability $p_m = 0.07$, and Polynomial Mutation operator with $\eta_m = 5$. The Epsilon clustering parameter is chosen as $\epsilon = 0.001$. The R-NSGA-II variant with delayed Pareto-fitness, DPF-R-NSGA-II, is run with a time threshold $\theta_t = 0.75$. This means that until 7,500 function evaluations, a purely distance-based optimization algorithm is used for optimization. After this time limit, the ordinary R-NSGA-II is run.

For all resampling algorithms, the minimum budget to be allocated is $b_{min} = 1$ and the maximum budget is $b_{max} = 10$. Static Resampling is run in configurations with $b_s \in [1, 5]$. Time-based Resampling uses uses a linear allocation, $a = 1$. Rank-Time-based Dynamic Resampling is run as RankMax5-based DR and uses linear allocation for both the rank-based ($b = 1$) and time-based criteria ($a = 1$). Rank-Time-based Resampling uses the minimum of the Pareto-rank-based allocation and the time-based allocation: $x_s^{RT} = \min\{x_s^T, x_s^R\}$. Progress-based Dynamic Resampling is done with the average progress $\overline{P}$ of the last $n = 3$ populations. Distance-Progress-Time-based Dynamic Resampling DDR uses delayed ($a = 2$) distance-based allocation. Distance-Rank-based Dynamic Resampling DR2 uses the same parameters as the underlying resampling algorithms.

The SEDR threshold is set to $\text{se}_{th}(s) = 20$, and $b_{min} = 2$. If it is adapted during the optimization runtime, it can vary between $\text{se}_{th}(s, p) \in [10, 20]$.For the Rank-based hybrid MO-SEDR algorithms, i.e., SE-RT-DR, SE-DDR, and SEDR2, we accelerate the allocation by choosing the time-based acceleration parameter $a = 0.5$, in order to facilitate higher resampling rates in the beginning of the optimization run.

D-OCBA-m is configured with $n_0 = 2$, $\bar{b}_{min} = 2.5$, $\bar{b}_{min} = 5$, $\delta = 1$, and $\Delta = 0.1|P| = 5$. For the Dynamic Best Selection D-OCBA-m algorithm, the m_p can vary between $m_{min} = 1$ and $m_{max} = |P|/2 = 25$. The adaptation is controlled by the elapsed optimization time. For Selection Sampling D-OCBA-m, m is set to $|P|$.

C. Performance measurement

All experiments performed in this study are replicated 10 times and mean performance metric values are calculated. Additional $b_f = 25$ samples are added to the final population to allow confident implementation decisions. The Focused Hypervolume performance measure which is described in the following is used to quantify the performance comparison between different DR algorithms. For the figures where the performance measurement has been performed over time during the optimization run, an interpolation of the metric values is done, on intervals of 100 function evaluations.

1) Focused Hypervolume F-HV: To measure and compare the results of the different resampling algorithms together with R-NSGA-II the Focused Hypervolume performance metric for ϵ-dominance based EMOs (F-HV) is used [26], [27]. Figure 4 shows the points in the objective space which define the F-HV measurement. F-HV applies a cylinder filter along an axis from R perpendicular to the Pareto-front. All solutions with objective vectors within this cylinder are passed on to the regular hypervolume measurement. The cylinder has a radius based on ϵ and the population size of R-NSGA-II. F-

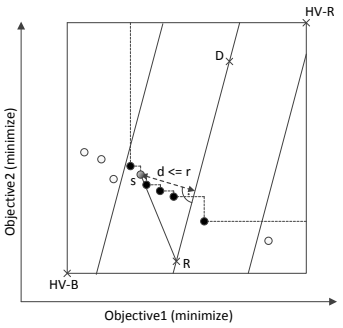

Fig. 4. The Focused Hypervolume (F-HV) performance measure with cylinder filter of radius r. Solution s is only passed on to the regular hypervolume measurement, if it is contained in the cylinder, which means $d \leq r$. Solutions outside of the cylinder subspace are not considered.

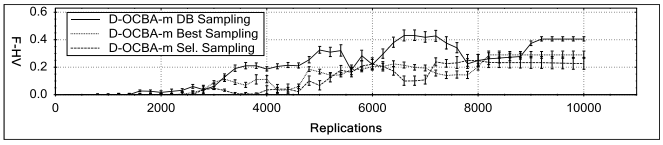

Fig. 5. F-HV measurement results on the ZDT4-20% benchmark function with trigonometric noise landscape for different sampling modes of Time-based D-OCBA-m on DPF-R-NSGA-II. For the last population, $b_f = 25$ final samples per solution have been executed. For all the other measurement/interpolation points (interval 200), 25 post-processing samples are calculated, which are not included in the simulation budget limit B =10,000. The error bars show the standard error of the mean of five experiment runs.

HV requires four points to be specified: A reference point R, a direction point D, and a hypervolume reference point and base point HV-R and HV-B. For ZDT4-20%, we use R =(0.05, 0.5), HV-R = (1, 50) and HV-B = (0, 0), and HV-D=HV-R. Due to the high noise levels of the used benchmark function and the limited sampling budget, the cylinder radius r is increased to $r = c\frac{|P|}{2}\epsilon$, $c \geq 1$. An increased value of $c = 2$ is used. Other performance metrics for Preference-based EMO algorithms were created by [22] and [7].

D. Experiment results

In Figure 5, we show a continuous F-HV measurement on ZDT4-20% with trigonometric noise landscape on the DPF-R-NSGA-II algorithm. We compare the three different sampling modes for D-OCBA-m. The results show that Dynamic-Best Sampling performs best. In Figure 6, the same comparison is done for the original R-NSGA-II. As expected, here the order of the three modes is different. Since we want to choose one of the three modes for our experiments, we choose Dynamic-Best Sampling due to the better result with DPF-R-NSGA-II.

Having identified Dynamic-Best Sampling as best operation mode for D-OCBA-m, we compare this result with optimization runs of the Dynamic Resampling algorithms from Section II, shown in Table III. The results show that D-OCBA-m is able to deliver better results than the best variance-based and non-variance-based DR algorithms, on the DPF-R-NSGA-II algorithm variant. As expected for the standard R-NSGA-II, most D-OCBA-m variants cannot achieve significanly better results.

VIII. Conclusions

The results show that the distance-based D-OCBA-m procedure is able to support the R-NSGA-II algorithm for problems with variable noise when the Dynamic-Best Sampling variant is used. Together with the R-NSGA-II variant with delayed Pareto-fitness DPF-R-NSGA-II, the best results could be achieved. For future work we have the following ideas:

- Evaluate the proposed variance-based resampling algorithms on more benchmark functions with different complexity from the ZDT benchmark problem suite, and on a real-world simulation optimization problems with variable noise landscapes.
- Perform a study on test problems with different noise strength in order to see how the algorithms behaves under different noise levels. In particular, the relationship between the objective noise level of the problem and the total simulation budget and maximum allowed number of samples per solution shall be investigated.
- Compare the D-OCBA-m algorithm with other variance-based Dynamic Resampling algorithms, such as MOCBA [5], CDR and MOPSA-EA [31].
- For extended analysis of the distance-based Dynamic resampling algorithm variants, future work will cover the their evaluation in optimization scenarios with attainable reference points.
- Propose and evaluate different methods to control the adaptive delay of the DPF-R-NSGA-II, which consider the distance to a reference point.

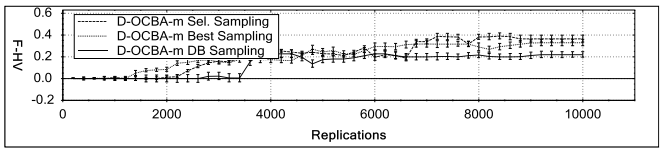

Fig. 6. F-HV measurement results as in Figure 5, but with the original R-NSGA-II algorithm.

TABLE III. F-HV measurement results on the ZDT4-20% benchmark function for different Dynamic Resampling algorithms on R-NSGA-II and its variant with delayed Pareto-fitness, DPF-R-NSGA-II. The measurement is performed on the last population where $b_f = 25$ final samples have been executed. D-OCBA-m is run as Dynamic-Best Sampling.

F-HV ZDT4-20%	DPF	Default
Static1	0.0762	0.1135
Static2	0.3247	0.3192
Static3	0.1914	0.3577
Static4	0.1623	0.1985
Static5	0.0000	0.0000
Time 1-10	0.2696	0.1577
R5T 1-10	0.3081	0.3120
DDR 1-10	0.2539	0.3402
DR2 1-10	0.3570	0.3620
SE-Time-DR 1-10	0.1623	0.2009
SE-RT-DR 1-10	0.2119	0.2319
SE-DDR 1-10	0.3630	0.2540
SEDR2 1-10	0.3241	0.3285
D-OCBA-m Time	0.4048	0.3647
D-OCBA-m R5T	0.3771	0.3249
D-OCBA-m DDR	**0.4263**	**0.4059**
D-OCBA-m DR2	0.3934	0.3437

ACKNOWLEDGMENT

This study was partially funded by the Knowledge Foundation, Sweden, through the BlixtSim and IDSS projects. The authors gratefully acknowledge their provision of research funding.

REFERENCES

[1] Bartz-Beielstein, T., Blum, D., and Branke, J.: Particle swarm optimization and sequential sampling in noisy environments. Metaheuristics – Progress in Complex Systems Optimization, 261–273 (2007)

[2] Branke, J., and Schmidt, C.: Sequential Sampling in Noisy Environments. Proceedings of the Parallel Problem Solving from Nature VIII conference - LNCS, vol. 3242, 202–211 (2004)

[3] Chen, C. H., He, D., Fu, M., and Lee, L. H.: Efficient Simulation Budget Allocation for Selecting an Optimal Subset. Informs Journal on Computing, vol. 20, no. 4, pp. 579–595 (2008)

[4] Chen, C. H. and Lin, J. and Yücesan, E. and Chick, S. E.: Simulation Budget Allocation for Further Enhancing the Efficiency of Ordinal Optimization. Discrete Event Dynamic Systems, vol. 10, no. 3, pp. 251–270, ISSN 0924-6703 (2000)

[5] Chen, C.-H. and Lee, L. H.: Stochastic Simulation Optimization - An Optimal Computing Budget Allocation. World Scientific Publishing Company, ISBN 978-981-4282-64-2 (2010)

[6] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, 182–197 (2002)

[7] Deb, K., Siegmund, F., and Ng, A. H.C.: R-HV : A Metric for Computing Hyper-volume for Reference Point Based EMOs. International Conference on Swarm, Evolutionary, and Memetic Computing 2014, Bhubaneswar, Odisha, India (2014)

[8] Deb, K., Sinha, A., Korhonen, P. J., and Wallenius, J.: An interactive evolutionary multi-objective optimization method based on progressively approximated value functions. IEEE Transactions on Evolutionary Computation, vol. 14, no. 5, pp. 723-739, (2010)

[9] Deb, K., Sundar, J., Bhaskara Rao N., U., and Chaudhuri, S.: Reference point based multi-objective optimization using evolutionary algorithms. International Journal of Computational Intelligence Research, vol. 2, no. 3, 273–286 (2006)

[10] Di Pietro, A., While, L., and Barone, L.: Applying evolutionary algorithms to problems with noisy, time-consuming fitness functions. Proceedings of the Congress on Evolutionary Computation 2004, vol. 2, 1254–1261 (2004)

[11] DiPietro A.: Optimizing Evolutionary Strategies for Problems with Varying Noise Strength. PhD thesis, University of Western Australia, Perth (2009).

[12] Fu, M. C.: Handbook of Simulation Optimization. International Series in Operations Research & Management Science, vol. 216, Springer-Verlag New York, ISBN 978-1-4939-1383-1 (2015)

[13] Fu, M. C. and Hu, J.-Q. and Chen, C.-H. and Xiong, X.: Optimal computing budget allocation under correlated sampling. Proceedings of the 2004 Winter Simulation Conference (2004)

[14] Fu, M. C. and Hu, J.-Q. and Chen, C.-H. and Xiong, X.: Simulation Allocation for Determining the Best Design in the Presence of Correlated Sampling. INFORMS Journal on Computing, vol. 19, no. 1, pages 101–111, ISSN 1526-5528 (2007)

[15] Goh, C. K. and Tan, K. C.: Evolutionary Multi-objective Optimization in Uncertain Environments: Issues and Algorithms. Springer-Verlag, ISBN 978-3-540-95976-2 (2009)

[16] Jaimes, A. L. and Arias Montaño, A. and Coello Coello, C. A.: Preference incorporation to solve many-objective airfoil design problems. Proceedings of the Congress on Evolutionary Computation 2011, New Orleans, USA, pp. 1605–1612, ISBN 978-1-4244-7834-7 (2011)

[17] Jin, Y., and Branke, J.: Evolutionary optimization in uncertain environments - a survey. IEEE Transactions on Evolutionary Computation, vol. 9, no. 3, pp. 303–317, ISSN 1089-778X (2005)

[18] Lee, L. H. and Chew, E. P.: Design sampling and replication assignment under fixed computing budget. Journal of Systems Science and Systems Engineering, Systems Engineering Society of China, vol. 14, no. 3, pp. 289–307, ISSN 1004–3756 (2005)

[19] Lee, L. H. and Chew, E. P. and Teng, S. and Chen, Y.: Multi-objective simulation-based evolutionary algorithm for an aircraft spare parts allocation problem. European Journal of Operational Research, vol. 189, no. 2, pp. 476–491, ISSN 1004–3756 (2008)

[20] Lee, L. H. and Chew, E. P. and Teng, S. Y. and Goldsman, D.: Finding the non-dominated Pareto set for multi-objective simulation models. IIE Transactions, vol. 42, no. 9, pp. 656–674 (2010)

[21] Miettinen, K.: Nonlinear Multiobjective Optimization. International Series in Operations Research & Management Science, vol. 12, ISBN 978-0-7923-8278-2 (1998)

[22] Mohammadi, A. and Omidvar, M. N. and Li, X.: A new performance metric for user-preference based multi-objective evolutionary algorithms. Proceedings of the Congress on Evolutionary Computation 2013, Cancún, pp. 2825–2832 (2013)

[23] Siegmund, F.: Sequential Sampling in Noisy Multi-Objective Evolutionary Optimization. Master thesis, University of Skövde, Sweden and Karlsruhe Institute of Technology, Germany (2009). Available at http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-3390

[24] Siegmund, F., Ng, A. H.C., and Deb, K.: Finding a preferred diverse set of Pareto-optimal solutions for a limited number of function calls. Proceedings of the IEEE Congress on Evolutionary Computation 2012, Brisbane, Australia, pp. 2417–2424, ISBN 978-1-4673-1508-1 (2012)

[25] Siegmund, F., Ng, A. H.C., and Deb, K.: A Comparative Study of Dynamic Resampling Strategies for Guided Evolutionary Multi-Objective Optimization. Proceedings of the IEEE Congress on Evolutionary Computation 2013, Cancún, Mexico, pp. 1826–1835, ISBN 978-1-4799-0454-9 (2013)

[26] Siegmund, F., Ng, A. H.C., and Deb, K.: Hybrid Dynamic Resampling for Guided Evolutionary Multi-Objective Optimization. Proceedings of the 8th International Conference on Evolutionary Multi-Criterion Optimization, Guimarães, Portugal, pp. 366–380, ISBN 978-3-319-15934-8 (2015)

[27] Siegmund, F., Ng, A. H.C., and Deb, K.: Dynamic Resampling for Preference-based Evolutionary Multi-Objective Optimization of Stochastic Systems. Submitted to European Journal of Operational Research (2016). Available at http://www.egr.msu.edu/ kdeb/papers/c2015020.pdf

[28] Siegmund, F., Ng, A. H.C., and Deb, K.: Standard Error Dynamic Resampling for Preference-based Evolutionary Multi-objective Optimization. Submitted to Computers & Operations Research (2016). Available at http://www.egr.msu.edu/ kdeb/papers/c2015021.pdf

[29] Siegmund, F., Ng, A. H.C., and Deb, K.: Hybrid Dynamic Resampling Algorithms for Evolutionary Multi-objective Optimization of Invariant-Noise Problems. Accepted for publication in Proceedings of the Evostar-EvoSTOC 2016 conference, Porto, Portugal (2016). Available at http://www.egr.msu.edu/ kdeb/papers/c2015023.pdf

[30] Stump, G. and Simpson, T. W. and Donndelinger, J. A. and Lego, S. and Yukish, M.: Visual Steering Commands for Trade Space Exploration: User-Guided Sampling With Example. Journal of Computing and Information Science in Engineering, vol. 9, no. 4, pp. 044501:1–10, ISSN 1530-9827 (2009)

[31] Syberfeldt, A. and Ng, A. H. C. and John, R. I. and Moore, P.: Evolutionary optimisation of noisy multi-objective problems using confidence-based dynamic resampling. European Journal of Operational Research, vol. 204, no. 3, 533–544, ISSN 0377-2217 (2010)

[32] Tan, K. C. and Goh, C. K.: Handling Uncertainties in Evolutionary Multi-Objective Optimization. Proceedings of the World Congress on Computational Intelligence 2008, Hong Kong, China, pp. 262–292, ISBN 978-3-319-15934-8 (2008)

[33] Van Veldhuizen, D. A.: Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. PhD-thesis, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, USA (1999)

[34] Zitzler, E., Deb, K., and Thiele, L.: Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation, vol. 8, no. 2, 173–195 (2000)

[35] Zitzler, E., and Thiele, L.: Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. Proceedings of the Parallel Problem Solving from Nature V conference - LNCS, 292–301 (1998)