Abstract—In Evolutionary Multi-objective Optimization many solutions have to be evaluated to provide the decision maker with a diverse choice of solutions along the Pareto-front, in particular for high-dimensional optimization problems. In Simulation-based Optimization the modeled systems are complex and require long simulation times. In addition the evaluated systems are often stochastic and reliable quality assessment of system configurations by resampling requires many simulation runs. As a countermeasure for the required high number of simulation runs caused by multiple optimization objectives the optimization can be focused on interesting parts of the Pareto-front, as it is done by the Reference point-guided NSGA-II algorithm (R-NSGA-II) [9]. The number of evaluations needed for the resampling of solutions can be reduced by intelligent resampling algorithms that allocate just as much sampling budget needed in different situations during the optimization run. In this paper we propose and compare resampling algorithms that support the R-NSGA-II algorithm on optimization problems with stochastic evaluation functions.

Keywords—Evolutionary multi-objective optimization, simulation-based optimization, guided search, reference point, decision support, stochastic systems, dynamic, resampling.

I. INTRODUCTION

In Multi-objective Optimization many solutions have to be evaluated to obtain a diverse and converged set of solutions that covers the Pareto-front. This set of Pareto-optimal solutions is presented to the decision maker who chooses only one of them for implementation. In particular for many-objective optimization problems a large number of Pareto-optimal solutions is required to provide the decision maker with a diverse set to choose from. In Simulation-based Optimization the number of available optimization function evaluations is very limited. This is because the modeled systems are usually complex and require time-consuming simulation runs. This leads to that it is not possible to find a diverse, converged set of solutions in the given time. If preference information is available however, the available function evaluations can be used more effectively by guiding the optimization towards interesting, preferred regions. One such algorithm for guided search is the Reference point-guided NSGA-II (R-NSGA-II) [9]. It is an Evolutionary Algorithm (EA) and Multi-Criteria Decision Making (MCDM) technique that uses reference points in the objective space provided by the decision maker and guides the optimization towards areas of the Pareto-front close to the reference points.

In Simulation-based Optimization the modeled and simulated systems are often stochastic. To perform an as exact as possible simulation of the systems this stochastic behavior is often built into the simulation models. When running the simulation this modeled uncertain behavior expresses itself in deviating result values of the simulation. This means that if the simulation is run multiple times for a selected parameter setting the result value is slightly different for each simulation run. In the literature this phenomenon of stochastic evaluation functions sometimes is called Noise, respectively Noisy Optimization [1], [3].

If an EA is run without countermeasure on an optimization problem with a noisy evaluation function 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 misjudgement will occur. The EA will see bad solutions as good and select them into the next generation. Good solutions might be assessed as inferior and can be discarded. 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 in the knowledge the algorithm has about the solutions. Resampling techniques sample the fitness of a solution several times and use the mean fitness values. In this paper, resampling methods are investigated that are designed to help R-NSGA-II to make good and reliable selection decisions. Another aspect of resampling, which is not considered, is the accuracy of the knowledge about solution quality after the optimization has completed. After the end of the optimization, the user has to decide which solution shall be implemented. For this purpose, the user needs accurate knowledge about the solutions and necessary samples needed to improve accuracy in the end are not available to the optimization procedure anymore.

There are basic resampling techniques that assign the same number of samples to all solutions or for example they assign the samples dependent on the elapsed optimization time. More advanced techniques base their sampling allocation on the individual characteristics of the solutions. Those Dynamic Resampling techniques assign a different number of samples to different solutions. After each new sample drawn from the fitness of a solution the sample mean usually will change.
More advanced dynamic resampling techniques perform Sequential Sampling [3], which means that they adjust their sampling allocation after each new evaluation of the fitness, if new information about the fitness is available. Dynamic Resampling techniques that perform Sequential Sampling for single-objective optimization problems have been proposed by for instance [6], [11], [4], [20]. Examples of multi-objective dynamic resampling algorithms are the Multi-Objective Optimal Computing Budget Allocation (MOCBA) [13] which has been applied to an EA [5] and Confidence-based Dynamic Resampling (CDR) [17]. Those algorithms are advanced and due to their complexity they are difficult to implement by optimization practitioners and they have limitations. That is why in this paper we propose and compare elementary and fundamental dynamic resampling algorithms with straight-forward implementation for multi-objective evaluation functions with the aim to support guided search. In addition, we present a resampling algorithm specifically adapted to R-NSGA-II. Some of the evaluated resampling algorithms are extensions of the single-objective versions. A classification of resampling algorithms is given.

For the resampling techniques we evaluate how well they can support R-NSGA-II as an example for a guided Evolutionary Multi-objective Optimization (EMO) algorithm. R-NSGA-II is based on the Nondominated Sorting Genetic Algorithm II [8] which is a widely-used multi-objective evolutionary algorithm. NSGA-II sorts the solutions in the combined set of population and offspring into different non-dominated fronts. Selected are all solutions in the fronts that fit completely into the next population, starting with the best fronts. From the front that only fits partially, those solutions are selected into the next population that have a high crowding distance, which is a diversity metric. The R-NSGA-II algorithm replaces the crowding distance operator by the distance to reference points. Solutions that are closer to a reference point get a higher selection priority. The reference points are defined by the decision maker in areas that are interesting and where solutions shall be found. As a diversity preservation mechanism R-NSGA-II uses clustering. Solutions that are too close to each other are considered a low priority. The reference points can be created, adapted, or deleted interactively during the optimization run. If no reference point is defined R-NSGA-II works as NSGA-II. A general flowchart of R-NSGA-II is displayed in Figure 1. Due to the complexity of the selection step it is displayed in detail in Figure 2.

The rest of the paper is structured as follows. In Section II a more detailed introduction to dynamic resampling is given and a classification for resampling algorithms is proposed. In Section III different resampling techniques are presented that are independent of the optimization algorithm, i.e. they can be used to support all multi-objective optimization algorithms and thereby also guided EMO algorithms. The resampling techniques are classified according to Section II. In Section IV a dynamic resampling technique explicitly designed for R-NSGA-II is proposed. In Section V the described resampling techniques together with R-NSGA-II are evaluated on benchmark functions. The test environment is described and the experiment results are analyzed. In Section VI conclusions are drawn and possible future work is pointed out.

II. NOISE COMPENSATION VIA RESAMPLING

To be able to assess the quality of a solution according to a stochastic evaluation function, statistical measures like sample mean and sample standard deviation can be used. By executing the simulation model multiple times, a more accurate value of the solution quality can be obtained. This process is called resampling. We denote the sample mean value of the objective function \( F_i \) and solution \( s \) as follows: \( \mu_n(F_i(s)) = \frac{1}{n} \sum_{j=1}^{n} F_i^j(s) \) and the sample variance of objective function
Considering variance comparison resampling coupled resampling

Large. Therefore Dynamic Resampling is often done sequentially.

Samples, if the initial estimate of the sample mean was too small for reaching the error threshold, in case the sample mean changes as new samples are added, this one-shot sampling allocation might not be small for reaching the error threshold, in case the sample mean changes as new samples are added, this one-shot sampling allocation might not be

However, since the sample mean changes as new samples are added, this one-shot sampling allocation might not be optimal. The number of fitness samples drawn might be too small for reaching the error threshold, in case the sample mean has shown to be larger than the initial estimate. On the other hand, a one-shot strategy might add too many samples, if the initial estimate of the sample mean was too big. Therefore Dynamic Resampling is often done sequentially.

For Sequential Dynamic Resampling often the shorter term Sequential Sampling is used.

Sequential Sampling adds a fixed number of samples at a time. After an initial estimate of the sample mean and calculation of the required samples it is checked if the knowledge about the solution is sufficient. If needed, another fixed number of samples is drawn and the number of required samples is recalculated. This is repeated as long as no additional sample needs to be added. The basic pattern of a sequential sampling algorithm is described in Algorithm 1. Through this sequential approach the number of required samples can be determined more accurately than with a one-shot approach. It guarantees to sample the solution sufficiently often, and can reduce the number of excess samples.

Input: Solution $s$

1. Draw $b_{min}$ initial samples of the fitness of $s_i F(s)$

2. Calculate mean of the available fitness samples for each of the $m$ objectives:

$$\mu_n(F_i(s)) = \frac{1}{n} \sum_{j=1}^{n} F_i(s), \quad i = 1, \ldots, m$$

3. Calculate objective sample standard deviation with available fitness samples:

$$\sigma_n(F_i(s)) = \sqrt{\frac{1}{n-1} \sum_{j=1}^{n} (F_i(s) - \mu_n(F_i(s)))^2}, \quad i = 1, \ldots, m$$

4. Evaluate termination condition based on $\mu_n(F_i(s))$ and $\sigma_n(F_i(s))$, $i = 1, \ldots, m$

   Stop if termination condition is satisfied, Otherwise sample the fitness of $s$ another $k$ times and go to step 2.

Algorithm 1: Basic sequential sampling algorithm pattern

Dynamic resampling techniques can therefore be classified as one-shot sampling strategies and sequential sampling strategies. In the following section we present several other aspects which can be used to classify resampling strategies.

B. Resampling Algorithm Classification

In this section we present a classification of resampling techniques. In the previous section two classification categories have been described. First, sampling algorithms that consider objective value variance and solutions that do not consider variance like Static Resampling. And second, Dynamic Resampling algorithms that calculate the sampling budget as one-shot and Dynamic Resampling algorithms that calculate the sampling budget sequentially. More classification categories can be defined. A list of categories is presented in Table I.

<table>
<thead>
<tr>
<th>Classification of Resampling Strategies</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not considering variance</td>
<td>Considering variance</td>
</tr>
<tr>
<td>One-shot sampling</td>
<td>Sequential sampling</td>
</tr>
<tr>
<td>Single-objective by aggregation</td>
<td>Multi-objective</td>
</tr>
<tr>
<td>Independent resampling</td>
<td>Coupled resampling</td>
</tr>
<tr>
<td>Individual resampling</td>
<td>Comparative resampling</td>
</tr>
</tbody>
</table>

Another way to determine the number of samples is to aggregate the objective values in a single value. For example, the R-NSGA-II algorithm provides such a scalar aggregation value on which the sampling budget allocation can be based: The distance to the reference point. A Distance-based Resampling strategy could allocate few samples in the beginning...
of the optimization runtime when the found solutions are far away from the reference point. Towards the end of the runtime more objective value samples can be drawn which helps to distinguish the many solutions that are close to the Pareto-front. On the contrary, a truly multi-objective resampling strategy would be to assign the maximum number of samples to solutions with Pareto-rank 1 and least samples to solutions with the maximum Pareto-rank.

If we use Distance-based Resampling on R-NSGA-II we classify the use of this resampling technique as a coupled resampling strategy application. On the other hand if Static Resampling is applied on an algorithm we classify its use as an independent resampling strategy application, since it is not taking any characteristics of the optimization algorithm into account. This classification category is not always applicable in clear way. If for example the Pareto-rank-based Resampling algorithm is applied on the R-NSGA-II algorithm then only one part of the algorithm selection procedure is supported. The secondary selection criterion, the distance to a reference point, is not supported.

The sampling strategies introduced in the previous section consider only one single solution at a time when making sampling allocation decisions. The Pareto-rank-based sampling strategy however is comparing solutions when allocating samples. The former strategy can therefore be classified as Individual Resampling and the latter strategy as Comparative Resampling.

C. Non-linear sampling allocation

Many Dynamic Resampling strategies limit the number of samples that can be allocated to a solution by an upper bound and an explicit lower bound, as for example Distance-based Resampling. For each solution a value is calculated that indicates the relative need of resamplings. Based on this value the algorithm designer can then decide how many samples shall be allocated to a solution. The relative sampling need can be translated directly into samples. For example, if a sampling need of 50% is indicated 50% of the span between the maximum and minimum allowed number of samples can be allocated, on top of the minimum resamplings. This would be a linear sampling allocation.

On the other hand the algorithm designer might transform this linear allocation. A deferred or decelerated sampling allocation for example might assign 10% of the maximum allowed samples to solutions with 50% sampling need and 80% of the maximum samples to solutions with sample need 90%. The opposite, an accelerated function could reach 50% of the allowed samples already at 10% sampling need. The formulas for the decelerated and accelerated transformation functions of the sampling need $\nu$ could be: $\nu^\nu$, and $\sqrt{\nu}$, $\nu > 1$. Mixtures of both function types are also meaningful. One example is the logistic distribution function of the Weibull distribution [12]:

$$1 - e^{-(\frac{x}{\lambda})^k}$$

with scale parameter $\lambda > 0$ and shape parameter $k > 1$. The distribution allows to delay the allocation in the beginning for lower sampling needs and to accelerate for higher sampling needs. Another example is the logistic growth [16]:

$$\frac{1}{1 + e^{-\gamma(x-\mu)}}$$

where $\gamma$ is the growth rate, $\mu$ the point of highest growth, and $\nu > 0$ determines if the maximum growth occurs close to the lower or the upper asymptote. The transformation functions are displayed in Figure 3.

III. ALGORITHM INDEPENDENT STRATEGIES

In this section several resampling algorithms are proposed and described that can be used to support all types of multi-objective optimization algorithms, inclusive guided EMO algorithms like R-NSGA-II. They do not use any specific algorithm characteristics and are universally applicable. We classify them according to Table I.

We denote: Sampling need for solution $s$: $x_s$, minimum and maximum number of samples for an individual solution: $b_{min}$ and $b_{max}$. The transformation functions explained in the previous section are applied on $x_s$ to obtain the relative resampling budget $b_s$. It is then discretised in the following way which guarantees that $b_{max}$ is assigned already for $b_s < 1$:

$$\min \{b_{max}, \lfloor b_s(b_{max} - b_{min} + 1) \rfloor + b_{min}\}$$

A. Static Resampling

Static Resampling samples all solution the same amount of times. Regarding the classification, it is not considering variance, a one-shot allocation, the number of objectives is not considered, it is an independent strategy as all strategies in this section, and an individual sampling allocation. The sampling budget is constant, $b_{min} = b_{max}$.

B. Time-based Resampling

Time-based Resampling allocates a small sampling budget in the beginning of the optimization and a high sampling budget towards the end of the optimization. The strategy of this resampling technique is to support the algorithm when the solutions in the population are close to the Pareto-front and to save sampling budget in the beginning of the optimization when the solutions are still far away from the Pareto-front.

Time-based Resampling is a dynamic resampling technique that is not considering variance, a one-shot allocation, the objective functions are not considered, and an individual
sampling allocation. We denote: \( B = \text{maximum overall number of simulation runs} \), \( B_t = \text{current overall number of simulation runs} \).

\[
x_s = \frac{B_t}{B}
\]

C. Domination-Strength Resampling

The term domination-strength is inspired by the strength value that is used in the SPEA algorithm [19]. It indicates the potential of a solution to dominate other solutions. Domination-Strength Resampling assigns more samples to solutions that dominate many other solutions. Those solutions have a higher probability to stay in the population during the next generations. Therefore, accurate knowledge about their fitness will be beneficial for algorithm performance. A disadvantage is that Domination-Strength Resampling is not effective in many objective optimization where almost all solutions are non-dominated during most of the optimization runtime.

Domination-Strength Resampling is not considering variance, performs sequential sampling, is purely multi-objective, and a comparative resampling technique. We denote: \( S = \text{solution set of current population and offspring}, \) \( \text{dom}(s, S) = \text{number of dominated solutions by } s \) in \( S \), \( M = \max_{s \in S} \text{dom}(s, S) \).

\[
x_s = \begin{cases} \frac{\text{dom}(s, S)}{M} & \text{for } M > 1 \\ \max \{0.5, \frac{\text{dom}(s, S)}{M} \} & \text{for } M = 1 \\ 1 & \text{for } M = 0 \end{cases}
\]

For \( M = 1 \) we increase the sampling need for solutions that do not dominate other solutions, in order to balance the allocation among non-dominated solutions. We assume that if \( M = 1 \) also most dominated solutions with \( \text{dom}(s, S) = 0 \) will be close to the Pareto-front and should receive more resampling budget.

D. Rank-based Resampling

Rank-based Resampling is similar to Domination-Strength Resampling. It assigns more samples to solutions in the leading fronts and less samples to solutions in the last fronts, to save evaluations. In a well-converged population most solutions will have Pareto-rank 1 and get the maximum number of samples. Like Domination-Strength Resampling it is not applicable in many-objective optimization.

Rank-based Resampling is not considering variance, performs sequential sampling, is purely multi-objective, and a comparative resampling technique. We denote: \( S = \text{solution set of current population and offspring}, \) \( r_s = \text{Pareto-rank of solution } s \) in \( S \), \( R = \max_{s \in S} r_s \).

\[
x_s = 1 - \frac{r_s - 1}{R - 1}
\]

E. Standard Error Dynamic Resampling

The working principle of Standard Error Dynamic Resampling (SEDR) was introduced in Section II-A. It adds samples to a solution sequentially until the standard error of the objectives falls below a chosen threshold value. It was proposed in [11] for single-objective optimization problems. In this study we apply SEDR on multi-objective problems by aggregating all objective values to a scalar value. As aggregation the median of the objective standard errors is used. A pseudocode description of SEDR is listed in Algorithm 2.

Regarding the classification, it is considering variance, is a sequential sampling procedure, uses an aggregated value of the objectives, and determines the resampling budget for all solutions individually. Since the algorithm is using the standard deviation multiple initial resamplings are needed for all solutions.

\[\text{input : Solution } s\]

1. Draw \( b_{\text{min}} \geq 2 \) initial samples of the fitness of \( s \), \( F(s) \)
2. Calculate mean of the available fitness samples for each of the \( m \) objectives: \( \mu_i(s), i = 1, \ldots, m \)
3. Calculate objective sample standard deviation with available fitness samples:
   \[
   \sigma_i(s) = \sqrt{\frac{1}{n-1} \sum_{j=1}^{n} (F_i^j(s) - \mu_i(s))^2}, \text{ for } i = 1, \ldots, m
   \]
4. Calculate objective standard errors
   \[
   \sigma_i(s) = \frac{\sigma_i(s)}{\sqrt{n}}, \text{ for } i = 1, \ldots, m
   \]
5. Calculate an aggregation \( \overrightarrow{s}(s) \) of the objective standard errors \( \sigma_i(s) \)
6. Stop if \( \overrightarrow{s}(s) < s_{\text{thr}} \) or if \( b_s = b_{\text{max}} \), otherwise sample the fitness of \( s \) another \( k \) times

Algorithm 2: Standard Error Dynamic Resampling (SEDR)

F. Combinations of multiple criteria

The resampling algorithms described so far all have shortcomings. Combinations of several algorithms which consider multiple criteria are expected to improve the sampling allocation. The following combinations are conceivable:

- Time-based + SEDR
- Domination strength/Pareto-rank + Time-based
- Domination strength/Pareto-rank + SEDR
- Even combinations of all three are imaginable

Combinations with time can be realized by increasing \( b_{\text{max}} \) with time. Combinations with SEDR can be done in two ways: The progress indicated by the supporting resampling technique can control either \( b_{\text{max}} \) or \( s_{\text{thr}} \).

In the next section we present a resampling algorithm that is specifically adapted to the R-NSGA-II algorithm.

IV. DISTANCE-BASED DYNAMIC RESAMPLING

In this section we propose a resampling strategy that uses the information that is available through the reference points of R-NSGA-II. It assigns more samples to solutions that are close to the reference points. We call it Distance-based Dynamic Resampling. The intention of allocating samples in this way is that solutions that are close to a reference point are likely to be close to the preferred area on the Pareto-front. They have converged and it is important to be able to tell reliably which solutions are dominating other solutions.

According to the classification in Table I Distance-based Dynamic Resampling is not considering variance, performs sequential sampling, uses the reference point distance which is an aggregated form of the objective values, is a coupled resampling method that allocates the sampling budget for each solution individually. We denote: \( R_s \) is the reference point that is closest to solution \( s \). In this study, we use an Achievement Scalarization Function [15] as distance metric. \( \delta_{\text{ASF}}(F(s), R_s) \) is the absolute reference point distance. For the sampling allocation we use the normalized reference point
distance \( d_s = \min \{1, \frac{\delta_{s \in \mathcal{F}}(F(s), R)}{D}\} \), where \( D \) is the maximum reference point distance of the initial population of R-NSGA-II. An intuitive way to assign resamplings based on reference point distance is as follows:

\[ x_s = 1 - d_s \]

Reference points are usually not chosen on the Pareto-front. Therefore adaptive non-linear transformation functions are needed which are described in the following section.

A. Adaptive Transformation Functions

In Section II-C we proposed transformation functions with the purpose of optimizing the sampling allocation. For Distance-based Resampling however, more complex transformation functions are needed since in most cases the reference points will not be situated on the Pareto-front. They are either feasible or infeasible and both cases need to be considered separately. The definition of a feasible reference point is that it is part of the Pareto-front or a solution can be found that dominates it. An infeasible reference point is not on the Pareto-front and no solution can be found that dominates it. Both infeasible and feasible cases require an approach that adapts the transformation function dynamically during the optimization runtime. It is not known from the beginning whether the reference point is feasible or infeasible. In the following two subchapters we propose adaptive transformation functions for both cases. First it is assumed that the reference point is infeasible and as soon as a solution is found that dominates the reference point the transformation function for the feasible case is used.

1) Adaptive Strategy for Infeasible Reference Points: In this section we propose a method how to adapt the transformation function in the case of an infeasible reference point. We use this transformation function as long as no solution is found that dominates the reference point. That means that feasible reference points on the Pareto-front are handled in the same way as infeasible reference points. In case of an infeasible reference point the reference point distance of the sought-after, focused Pareto set cannot be reduced below a certain distance \( \delta \) by optimization. This scenario is displayed in Figure 4. That means that if this lower bound \( \delta \) of the reference point distance is large enough and a linear or delayed distance-based allocation scheme is used the maximum sampling budget will never be reached during the optimization runtime. Too few samples will be assigned to the last populations that include the best found solutions. Therefore, in order to create a good distance-based sampling strategy for infeasible reference points, the budget allocation has to be accelerated. Then, it is not necessary to come close to the reference point to get the highest number of samples. The full number of samples will already be allocated to solutions at a certain distance to the reference point. Since the lower bound of the reference point distance is not known the sampling allocation has to be accelerated iteratively. The challenge is to adapt the transformation function automatically so that in the end of the optimization run the samples are distributed among all found solutions as if the reference point was feasible and on the Pareto-front.

To achieve this the distance indicator is combined with two additional criteria, as it was done for the algorithm independent strategies in Section III-F:

1. The progress of the population during the optimization runtime and 2. The elapsed time since the start of the optimization. In contrary to the algorithm independent strategies this algorithm has access to distance information to reference points and can measure progress. The optimization progress will slow down as soon as the population approaches the Pareto-front which is at some distance to the reference point. In this way it can be detected that the algorithm is converging towards the preferred area and that more samples per solution are needed. However, the reference point distance and the optimization progress alone would be not sufficient since the algorithm might temporarily get slowed down by local optima in an early stage and show only little progress. This can for example occur for the ZDT4 function [18] which features many local Pareto-fronts. In that case the goal is to escape the local optimum which does not require more exact knowledge about the solutions and no increased number of samples. That is why in the case of infeasible reference points the reference point distance, progress, and elapsed time together are taken into account for the sampling allocation.

We propose to measure the optimization progress by the average reference point distance of the population, on average for the last \( n \) populations. This metric will be described in detail in the experiment section of this paper, Section V-A. The average improvement in reference point distance of the last \( n \) populations is used as progress indicator. The transformation function is designed as an ascending polynomial function. Initially, it will assign less than the maximum allowed samples to the solution that is closest to the reference point in order to be prepared for the case of a feasible reference point. If the reference point is feasible the solutions that dominate the reference point should get the highest number of samples instead. As long as we can assume infeasibility and if the optimization progress per generation drops the slope of the transformation function will be increased in several steps. If the progress for a generation is less than 10% per generation the transformation function will assign the maximum allowed number of samples only to the (hypothetical)solutions that are closest to the reference point. If the average progress is below 5% the transformation function will be adapted in a way that at least 10% of the population are guaranteed full samples. Accordingly, a progress \( < 2.5\% \) corresponds to 20% of the population with full samples and a progress \( < 1\% \) to 40%.

As transformation function \( x_s^2 = (1 - d_s)^2 \) is used. The calculation is done by taking a percentage of the population and measuring the maximum relative distance \( m \) of this subset \( S \) to the reference point, \( m = \max_{s \in S} d_s \). The transformation function is adapted through a factor \( r \) that makes sure that the
maximum number of samples is reached already at distance \( m \) to the reference point.

\[
    r(1 - m)^2 := 1 \Rightarrow r = \frac{1}{(1 - m)^2}
\]

However, as mentioned above, if the average population progress is above 10\% then the sampling allocation will be reduced by using \( r = 1 - m \), where \( m \) is the maximum distance of the best 10\% of the population. In case that it will be detected that the reference point is feasible this allows a smooth transition to the feasible case which uses a similar allocation function. The transformation functions are defined as

\[
    \min\{1, r(1 - d)^2\}.
\]

This allocation scheme is combined with a sample allocation based on the elapsed optimization runtime. The elapsed time is measured by the number of executed solution evaluations in relation to the maximum number of evaluations, as it is done by Time-based Resampling in Section III-B. The transformation function is adapted by reducing \( r \) to delay the incline of the sampling allocation. Before 50\% of the time has passed the distance \( m \) will be reduced to \( m_0 = 0 \), i.e. \( r = 1 \). Until 65\% \( m \) will be reduced to \( m_1 = 1/3m \) and until 80\% it will be reduced to \( m_2 = 2/3m \). The transformation functions are shown in Figure 5.

\[
    \begin{align*}
    \text{Sampling need } x &= 1 - d \quad \text{(\% of max)} \\
    \text{Assigned samples } \% &= \frac{1}{(1 - m)^2} \quad \text{if } d \geq 100 \\
    &\quad \text{or } \min\{1, r(1 - d)^2\} \quad \text{if } 0 < d < 100
    \end{align*}
\]

![Fig. 5. Transformation functions for infeasible reference point. The sampling need \( x \) in percent based on the relative reference point distance \( d \) is translated into a relative number of samples that are allocated to a solution.](image)

As soon as a solution is found that dominates the reference point the sampling allocation will switch to the allocation function for the case of feasible reference points, starting with the next generation of R-NSGA-II. If this happens at an early stage then only a lesser change in sampling allocation will occur and a smooth transition will happen since then the transformation function for infeasible reference points is similar to the one for feasible reference points. The feasible case is described in the next section.

2) Adaptive Strategy for Feasible Reference Points: In the beginning of the optimization runtime the decision maker might define a reference point that is feasible. That means that it is possible to find a solution that features exactly the same objective values as the reference point. This may happen since in most real-world cases the Pareto-front is not known before the optimization is started. The reference point might be set too conservatively and during the optimization run it might be detected that solutions better than the reference point can be found. A scenario with a feasible reference point is displayed in Figure 6. R-NSGA-II can handle this case [9], i.e. the optimization will still move towards the preferred part of the Pareto-front. This is because the Pareto-dominance is the primary fitness criterion that is checked before the solutions are compared by the reference point distance.

For the sampling allocation however, this would mean that the allocation would be reversed. Solutions that are closer to the Pareto-front are further away from the reference point and would be sampled less. We therefore propose a sampling allocation that is based on an imaginary reference point that is not dominated by any solution in the population.

For R-NSGA-II we recommend that the reference point is adapted in case of a feasible reference point, given that the decision maker is available. This is because an infeasible reference point or a point on the Pareto-front will help R-NSGA-II to converge faster. This is important since we assume a limited optimization time. One might also think of automatic methods for reference point adaption based on previously found information about the optimization problem. In this paper those approaches are not considered. Instead an adaptive transformation function is proposed that will increase the sampling allocation for solutions that are expected to be close to the interesting area on the Pareto-front.

We propose a distance-based sampling allocation function that is activated as soon as a solution is found that dominates the reference point. It is designed to imitate the sampling allocation as if the reference point would be infeasible or situated on the Pareto-front. For this purpose the reference point is replaced by an imaginary reference point and the sampling allocation is based on the distances to this point. As the imaginary reference point the objective vector of the solution is used that is non-dominated, dominates the reference point, and is closest to it. This point is only used for the sampling allocation, not for guidance of the algorithm.

If the imaginary reference point is far from the interesting area then this could lead to a suboptimal sampling distribution. This is likely to happen if the Pareto-front is for example almost vertical or horizontal in a 2D objective space. The interesting area is a set of solutions on the Pareto-front that are
closest to the reference point, for example the closest solution and all solutions on the Pareto-front that have a 10% higher distance. A solution that is in the interesting area but does not dominate the reference point will be closer to the reference point than the imaginary reference point which is far from the interesting area. Thereby, it will not get the maximum possible number of samples. This is a disadvantage of our approach, to only define solutions as imaginary reference points that dominate the reference point. This disadvantage is accepted in order to keep the approach as simple as possible. An example scenario is displayed in Figure 7. However, this suboptimal sampling allocation is only temporary. R-NSGA-II itself is not directly influenced by the imaginary reference point, only the sampling allocation is. As the optimization progresses the (sub-)population and thereby the imaginary reference point will move towards the interesting area.

Until 65% has passed $r$ is chosen as $r = 1/2$, until 80% $r$ is $3/4$. After 80% of the time the allocation is not slowed down. The transformation functions are displayed in Figure 8.

Likewise to the infeasible case we would like to use the progress and the elapsed time as additional criteria for sampling allocation. The time can be used in the same way as for the infeasible case but the progress is difficult to measure. The imaginary reference point will move as the optimization is progressing which means that if the population follows this point then the progress is not measurable by the distance to the imaginary reference point. For progress measurements a fixed point close to the preferred area on the Pareto-front would be needed. For reasons of simplicity we set aside progress measurements for the feasible case in this paper.

The transformation function maps the relative sampling need $x_s = 1 - d_s$ to the sampling allocation in percent, where $d_s$ is normalized reference point distance as defined above, however to the imaginary reference point. The function that is used is defined as

$$r(1 - d_s)^2, \quad r \leq 1.$$  

The factor $r$ is controlled by the elapsed time. Until some time has passed it is expected that the population is still progressing. Many solutions would be assigned a high sampling budget since the imaginary reference point is defined as a solution that is part of the population which is converging towards each other in a cluster close to the preferred area. Therefore the allocation is delayed initially. Unless 50% of the time has passed the sampling allocation is slowed down with $r = 1/4$. The factor $r$ will move towards the interesting area. Thereby, it will not get the maximum possible point than the imaginary reference point which is far from the interesting area. This disadvantage is accepted in order to keep the approach as simple as possible. An example scenario is displayed in Figure 7. However, this suboptimal sampling allocation is only temporary. R-NSGA-II itself is not directly influenced by the imaginary reference point, only the sampling allocation is. As the optimization progresses the (sub-)population and thereby the imaginary reference point will move towards the interesting area.

![Fig. 7. Bi-objective min/min-problem scenario with feasible reference point $R$ and almost horizontal Pareto-front. The imaginary reference point $IR$ is the solution that is non-dominated, dominates $R$, and is closest to $R$ among the solutions that dominate $R$. In this scenario $IR$ is distant from the preferred area due to the characteristics of the Pareto-front. Solution $P$ however is preferred but will not get the maximum number of samples.](image)

**B. Considering Variance**

Analogously to Section III-F Distance-based Dynamic Resampling can be combined with SEDR. Then all of distance, progress, time, and variance would be considered together.

**V. NUMERICAL EXPERIMENTS**

In this section the proposed resampling techniques in combination with R-NSGA-II are evaluated on benchmark functions. Experiments are done for the two-dimensional ZDT benchmark function ZDT1 [18], with an additive noise of $\sigma = 0.15$.

**A. Evaluation**

To measure the result quality of the optimization, each solution that is part of the found non-dominated set is sampled additional 100 times after the optimization is finished. In this way accurate objective values can be obtained. The sampling budget needed for the post-processing is not taken from the total sampling budget of the experiments.

As evaluation metric, we use the average distance of the result population to the reference point. To eliminate the influence of outliers, only an $\alpha$-subset of the population is used in the calculation, $\alpha \in [0, 1]$. The subset contains the solutions that are closest to the reference point. For the experiments $\alpha = 0.5$ is used, which means that half of the population influences the performance measurement.

Performance measurements are made after every generation. Due to the resampling techniques, the number of evaluations per generation varies between different generations, algorithms, and even different optimization runs. In order to obtain measurement values in equidistant time intervals, the measurement data is interpolated. All experiments are run 5 times, and average interpolated performance measurement values are used to compare the algorithms.
B. Parameter Configuration

For all experiments a simulation budget of 5000 is used. R-NSGA-II parameter values are: population size=50, number of offspring=50, crossover probability=0.8, and mutation probability=0.07.

C. Proof of concept

In Figure 9 the optimization process of R-NSGA-II with Time-based Resampling is displayed. The optimization process with SEDR is shown in Figure 13, and Distance-based Resampling in Figure 14. The color values indicate the number of samples (red=low, green=high). The sampling allocation of the time-based experiment is shown in Figure 10. If after some time, the number of allocated samples has increased, the new number of resamplings will be applied as soon as a new generation has started. This short-time allocation delay is done in order to keep the allocation transparent and understandable. A linear transformation function was used in this experiment. If a delaying transformation function would be used, the allocation looks as in Figure 11. With the delayed allocation more different solutions can be evaluated, since the samples are spent more economically on the solutions.

D. Performance measurements

In this section the resampling strategies are tested in combination with R-NSGA-II. They are Static, Time-based, Domination strength-based, Pareto rank-based, SEDR, and Distance-based. The upper sampling limit is \( b_{\text{max}} = 3 \). The performance metric values over time are shown in Figure 12. It can be observed that Static Resampling performs worst, the time-based strategy and Distance-based Resampling perform best. The Domination Strength- and Rank-based algorithms show medium performance. They allocate a higher sampling budget throughout the runtime which gives them a disadvantage. The same effect, but worse, can be seen for SEDR which allocates many samples to compensate for the high noise level.

VI. Conclusions

It has been shown that all proposed resampling strategies perform better than Static Resampling. The specifically adapted Distance-based Resampling algorithm for R-NSGA-II and Time-based Resampling show the best performance. Comparative resampling algorithms like Domination Strength Resampling are promising methods.
A. Future work

For future work the following studies are proposed:

- The performance metric that is used measures only how well the algorithm has converged towards the reference point. A future task would be to create a performance metric for guided search that also measures diversity.

- Another worthwhile study would be to create a re-sampling algorithm that explicitly supports selection sampling, i.e. that maximizes the probability of correct selections.

- The trade-off between achieving a good optimization result and an accurate knowledge of the fitness of the result population will be investigated.

**Fig. 13.** EMO with R-NSGA-II and SEDR on ZDT1, noise level $\sigma = 0.15$, reference point $(0.5, 0)$, 5000 evaluations.

**Fig. 14.** EMO with R-NSGA-II and Distance-based Resampling on ZDT1, noise level $\sigma = 0.15$, reference point $(0.5, 0)$, 5000 evaluations.

ACKNOWLEDGMENTS

This study was partially funded by VINNOVA, Sweden, through the FFI-HSO project. The authors gratefully acknowledge their provision of research funding.

**REFERENCES**


