Int j simul model 20 (2021) 1, 123-133

Original scientific paper

INITIAL POPULATION INFLUENCE ON HYPERVOLUME CONVERGENCE OF NSGA-III

Glamsch, J.; Rosnitschek, T. & Rieg, F.

Chair of Engineering Design and CAD, University of Bayreuth,
Universitätsstr. 30, 95447 Bayreuth, Germany

E-Mail: johannes.glamsch@uni-bayreuth.de

Abstract

A common method for solving multi-objective optimization problems are evolutionary algorithms (EA), which are utilizing an iterative population-based approach and do not need prior information about the problem to be solved. These algorithms require a variety of control parameters, e. g. the three evolutionary operators (selection, crossover and mutation), a termination criterion and the population size, which are subject of many studies. In contrast to these a less considered factor is the initialization of the first population. This paper analyses the influence of different initialization methods besides the classic sampling with a pseudo-random number generator on the convergence behaviour of the algorithm NSGA-III.

It can be shown that different sampling methods affect the convergence behaviour significantly, whereby some methods increase while others decrease the convergence speed. The results also show a strong dependency and interaction between the initialization method and the optimization problem.

(Received in November 2020, accepted in December 2020. This paper was with the authors 1 week for 1 revision.)

Key Words: Evolutionary Algorithm, Multi-Objective Optimization, NSGA-III, Sampling, Initial Population

1. INTRODUCTION

The process of optimizing a system occurs almost everywhere. In technical context some examples are the optimization of die-casting process parameters [1], CNC tool paths [2], machine parameters [3], manufacturing scheduling [4] and many more [5-8].

In mathematics, optimization basically describes the problem of finding the minimal value of an objective function f depending on its decision variables x_i. In many cases further conditions must be met. For example, various constraints, e. g. (in-)equality constraints $g_j(x_i)$ respectively $h_k(x_i)$ or box constraints (x_i^L, x_i^U) of the decision variables must be satisfied. Therefore, the whole optimization problem can be written as follows [9-12]:

$$\min f(x_i)$$
$$\text{subject to} \begin{cases} g_j(x_i) \le 0 & j = 1, \dots, J \\ h_k(x_i) = 0 & k = 1, \dots, K \\ x_i^L \le x_i \le x_i^U & i = 1, \dots, n \end{cases} \tag{1}$$

In real-world applications, the systems to be optimized are often complex and often cannot be described by only one objective function $f(x_i)$. In fact, there are multiple objective functions $f_m(x_i)$ which should be considered simultaneously. This leads to the following mathematical definition of a multi-objective optimization problem [9-12]:

$$\min f_m(x_i) \quad m = 1, \dots, M$$
$$\text{subject to} \begin{cases} g_j(x_i) \le 0 & j = 1, \dots, J \\ h_k(x_i) = 0 & k = 1, \dots, K \\ x_i^L \le x_i \le x_i^U & i = 1, \dots, n \end{cases} \tag{2}$$

In order to illustrate this general formulation we look at the following example from product development, adapted from [1]: When optimizing the die-casting process, the solidification

https://doi.org/10.2507/IJSIMM20-1-549

time $f_1(x_i)$ and the probability of defects due to insufficient filling $f_2(x_i)$ should both be minimized, which leads to a trade-off between two objective functions. Therefore, no single best/minimal solution exists, but a set of multiple best solutions, the so-called Pareto-optimal solutions, see Fig. 1, left. This implies that one objective can only be improved by worsening another objective. The Pareto-optimal front in objective space (Fig. 1, right) is formed by all Pareto-optimal solutions which are not dominated by another solution. In this context, dominating means that one solution is no worse in all objective functions and at least strictly better in one objective function than another solution [9-12].

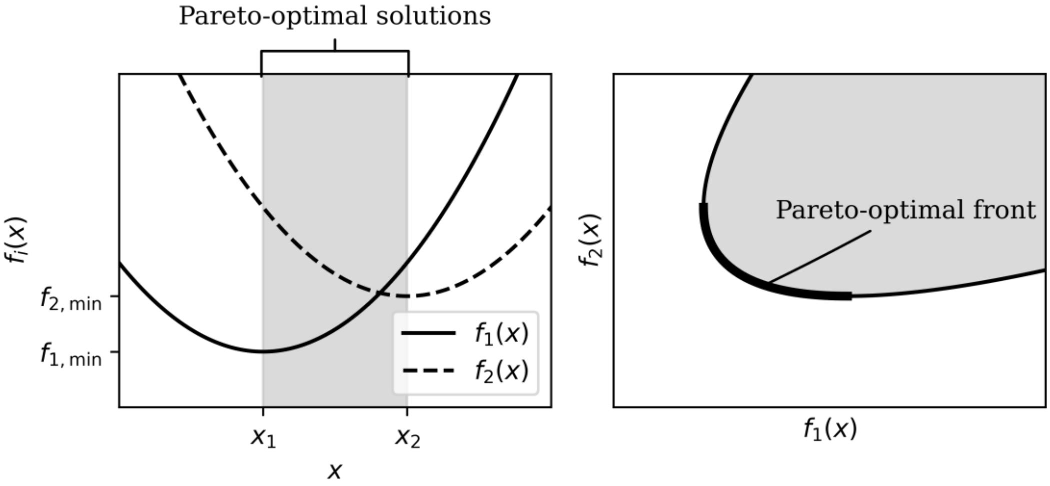


Figure 1: Visualization of the Pareto-optimal solutions and the Pareto-optimal front in decision (right) and objective (left) space [9].

To solve this kind of problems with conventional optimization algorithms, the optimization problems are often transformed into a single-objective problem. Possible approaches are for instance summing up all weighted objective functions or considering all but one objective as restrictions. However, these methods have various disadvantages (e. g. only one solution is generated per optimization run) [9-12].

Another way besides the conventional algorithms for solving multi-objective optimization problems is the use of evolutionary algorithms (EA). These heuristic-based algorithms use an iterative population-based approach to imitate the biological evolution, the “survival of the fittest”. A population consists of multiple individuals, which represent the decision variables. This approach implies that a first, initial population has to be generated. At present, the decision variables of these individuals are mostly sampled randomly [9-12]. At our best knowledge, there are only a few studies (e. g. [13-17]) which examine the influence of other initialization methods, especially for multi-objective optimization problems.

This paper extends this research by analysing and evaluating various alternatives for initializing the first population to improve the convergence behaviour of state of the art multi-objective EAs without further effort or knowledge of the system.

2. MULTI-OBJECTIVE OPTIMIZATION USING EVOLUTIONARY ALGORITHMS

As stated in the previous chapter, EAs try to imitate the biological evolution. For this purpose, an initial population consisting of *N* individuals has to be generated, whereby each individual represents one possible solution of the optimization problem. The fundamental procedure of most EAs is the same:

In an iteration loop over multiple generations an offspring population is generated by using the evolutionary operators selection (select promising individuals for mating), crossover (exchanging information between two or more parents) and mutation (randomly perturbing the decision variables of the child solutions). The offspring population will then be evaluated and the best individuals of the offspring population or the combination of the parent and the offspring population will be carried over into the next population [9-12].

There is a large number of various multi-objective evolutionary algorithms. One of the most common algorithms is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [18] and its successor NSGA-III [19, 20]. The latter was chosen for the investigations presented in this paper because of its very good convergence behaviour on many optimization problems.

The iteration loop of NSGA-III is briefly explained as follows: After creating an offspring population using the evolutionary operators, the combination of the parent and offspring population (containing 2*N* individuals) is divided into non-dominating fronts. Therefore, individuals of first front are not dominated by any other individuals, individuals in the second front are only dominated by individuals of the first front and so on. Afterwards, all individuals of the fronts are taken over, which fit completely into the next population of the size *N*. From the last front, which can no longer be fully adopted, those individuals are selected which are closest to a pre-defined set of reference points in objective space. In this way, an even distribution of the solutions should be achieved. More details on the algorithm can be found in [19, 20].

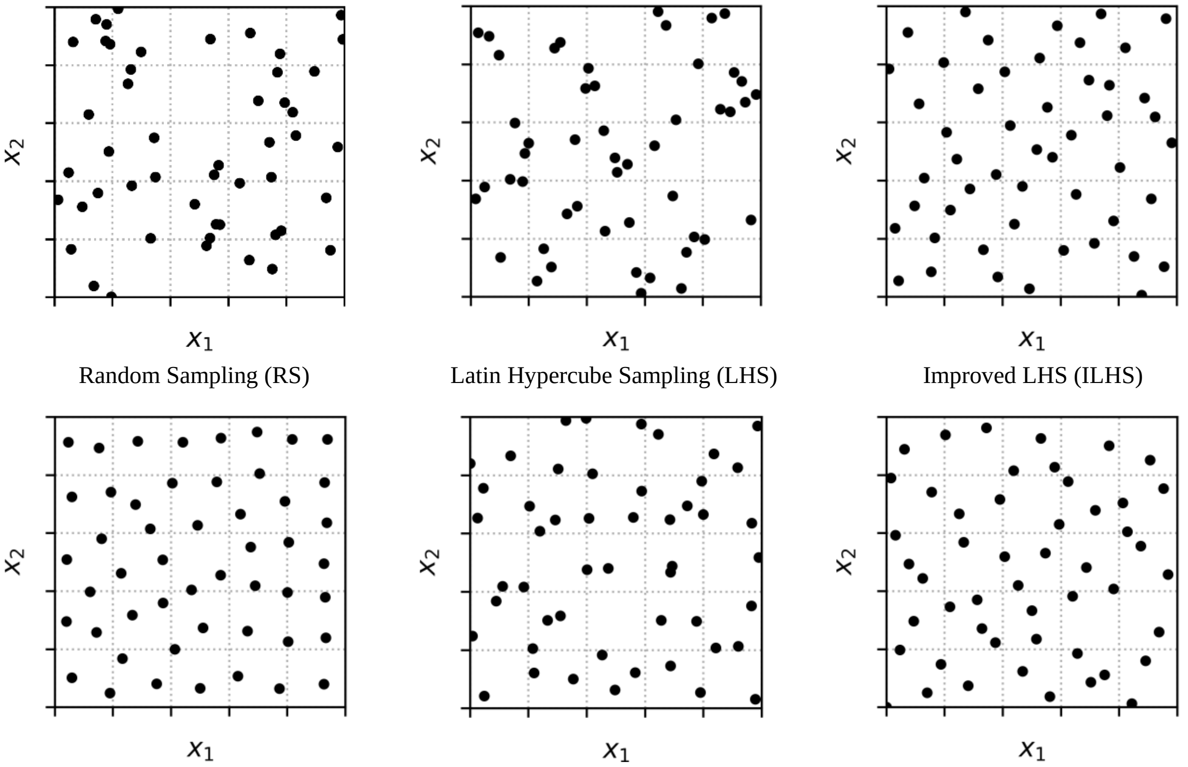
3. DESCRIPTION OF THE ANALYSIS METHOD

When optimizing a system, the user often has no prior knowledge of the problem behaviour, which can be used to generate a "good" initial population, and considers it as a black-box [12, 21]. Since therefore no application specific initialization methods can be used, mostly pseudo-random number generators (PRNG) are being utilized generate the decision variables of the initial population and to cover all regions of the search space uniformly [21]. The individuals of the initial population are the starting values of optimization and the choice of these affects the convergence behaviour. [21] summarises, that due to an improved initial population the probability of finding the global minimum can be increased and the variance of the solution can be reduced. The aim of this study is to examine if other, simple initialization methods, which need no prior knowledge of the system, perform better than the classic random sampling for NSGA-III because they cover the design and objective space "better" or more appropriate.

For this paper the following sampling methods are selected [22, 23]:

- Random Sampling (RS) [22]:
  Random uniform sampling used as a reference.
- Latin Hypercube Sampling (LHS) [24]:
  This method divides each decision variable range into equal sized intervals and retrieves a sample from each interval.
- Improved Latin Hypercube Sampling (ILHS) [25, 23]:
  This method is a further development of the classic Latin Hypercube Sampling which tries to improve the distribution of the sampled points.
- *k*-means Sampling (KS) [26]:
  The *k*-means sampling leads to a Voronoi tessellation of the decision space.
- Stratified Sampling (STS) [24, 27]:
  Using this method, the design space is divided into subcells and one point is drawn from each cell.
- Halton Sampling (HS) [23, 28]:
  The Halton sequence is a quasi-random sequence (deterministic) for creating uniformly distributed points.

The implementation of these algorithms is based on the software package *diversipy* (Version 0.8) [22]. To visualize the differences between the selected methods, Fig. 2 shows a distribution of 50 two dimensional points generated by each sampling method. Clear differences are easily discernible: Some methods, like for example KS sampling, look more uniform than others, for instance LHS. These different distributions lead to the question of whether these different approaches have any influence on the convergence behaviour of NSGA-III and if so, to what extent do they have an effect on different optimization problems.



Random Sampling (RS)   Latin Hypercube Sampling (LHS)   Improved LHS (ILHS)

k-means Sampling (KS)   Stratified Sampling (STS)   Halton Sampling (HS)

Figure 2: Visualization of the used sampling methods.

To compare the results of different optimization runs, a quality criterion that rates the outcome is required. [12] defines two main goals of multi-objective optimization: Finding solutions which lie on the Pareto-optimal front and which are simultaneously diverse enough to cover the whole Pareto-optimal front. To measure the quality of an optimization result, various performance metrics have been developed. According to a study of [29], one of the most common metrics is the so-called hypervolume (HV), which describes the volume enclosed between a reference point and the Pareto-optimal solutions and therefore should be maximized [30, 31]. Benefits of this metric are that the cardinality, accuracy and diversity of the optimization results can be rated simultaneously [29]. It should be noted that recent research shows that comparing the HV of different optimizations is more complicated than it seems. For example, the population size, the number of non-dominated solutions and the choice of the reference point have an impact on the HV [32]. We compose the required reference point of the maximum values of each objective of the known real Pareto-optimal front, like stated in [31].

To analyse the performance of multi-objective EAs, various test functions were published in the literature. Most common are the ZDT [33] and DTLZ [34] problem collections. The ZDT test problems consist of two objectives and a configurable number of decision variables n. We use $n = 20$, $n = 30$ and $n = 40$ for the problems ZDT1 to ZDT3 and $n = 5$, $n = 10$ and $n = 20$ for ZDT4 & ZDT6. The DTLZ problems have a configurable number of objectives M and number of decision variables n. For this paper, $M = 3$ was selected. The number of decision variables was set to $n = 5$, $n = 7$ and $n = 10$ for DTLZ1 respectively $n = 5$, $n = 10$ and $n = 20$ for DTLZ2 to DTLZ4. The middle value of n of each optimization problem corresponds to the default value in [35].

As stated, the algorithm NSGA-III will be used for this study. The control parameters are selected according to [19]: We use a population size of $N = 92$ individuals and select a maximum number of generations t as a termination criterion. The value of t is set to 1000, even if this leads to no full convergence for all test problems. The simulated binary crossover operator is used with a distribution index of $\eta_c = 30$ and a crossover probability of $p_c = 1.0$. As mutation operator the polynomial mutation with a distribution index of $\eta_m = 20$ and a probability of $p_m = 1/n$ (with the number of decision variables n) is used. The implementation of the algorithms and the test problems is based on the software package *pymoo* (Version 0.4.2.1) [35].

Each combination of test problem and sampling method was repeatedly computed 200 times to reduce the influence of the nondeterministic behaviour. The results are shown and discussed in the next section.

4. RESULTS AND DISCUSSION

As an introductory example, Fig. 3 shows the mean HV for ZDT1 ($n = 30$) depending on the number of generations t for the sampling methods RS, KS and HS.

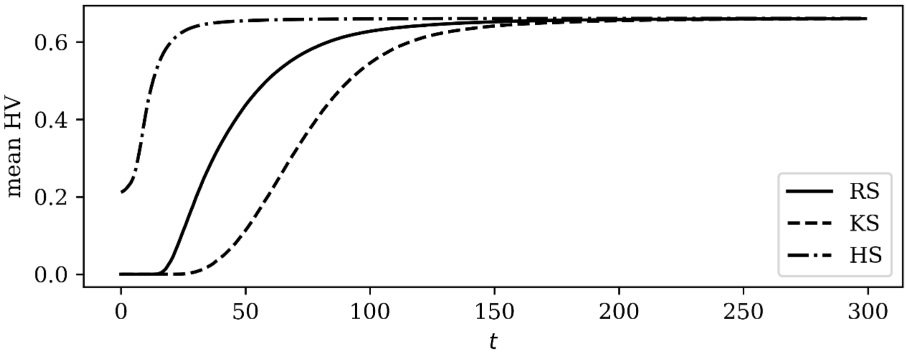


Figure 3: Mean HV depending on number of generations t (ZDT1, $n = 30$).

It is clearly visible that the sampling method affects convergence behaviour of the algorithm: The optimization runs using HS converge much faster than RS, while KS shows an even worse convergence behaviour than RS. Because this kind of visualization of the results is impractical and hard to interpret not all results a presented in this form. Instead, all results are normalized using the highest mean HV of all 200 optimization runs of the RS initialization method for each problem. Then it is evaluated at which generation the normalized mean HV of RS reaches at least 0.5, 0.75 and 0.95 (HV_{rel}). Afterwards, a table shows by what percentage the HV of the other sampling methods differs from the HV of RS at this generation. The higher this value is, the better the sampling method, because it converges earlier. The highest (best) value of each row is highlighted in green (🟩), the second highest value in light green (🟩). The lowest (and therefore worst) value is underlined and highlighted in light red (🟥). In all results the mean HV of all 200 runs was used. In addition, it is calculated whether the deviation of the mean HV may only be subject to statistical fluctuations and therefore the significance level is indicated (*** ≙ $p < 0.001$; ** ≙ $p < 0.01$; * ≙ $p < 0.05$) using Welch's t-test [36].

4.1 ZDT problem collection

Table I shows the results for the ZDT problem collection in dependency of their number of decision variables n.

Table I: Results for the ZDT problem collection.

Problem			**LHS**	**ILHS**	**KS**	**STS**	**HS**
Name	***n***	***HV***$_{\text{rel}}$					
ZDT1	20	0.50	+2.27 %	+3.30 %**	-99.32 %***	-0.36 %	+87.25 %***
		0.75	+1.13 %	+1.50 %**	-85.79 %***	-0.35 %	+30.01 %***
		0.95	+0.19 %	+0.29 %*	-22.89 %***	+0.04 %	+4.81 %***
	30	0.50	+3.86 %***	+4.31 %***	-87.25 %***	-0.77 %	+94.10 %***
		0.75	+1.38 %*	+1.81 %***	-61.38 %***	-0.23 %	+32.31 %***
		0.95	+0.16 %	+0.18 %	-12.53 %***	+0.01 %	+4.99 %***
	40	0.50	+5.24 %***	+4.26 %***	-41.00 %***	+1.16 %	+95.75 %***
		0.75	+2.26 %***	+1.84 %***	-21.39 %***	+0.45 %	+32.43 %***
		0.95	+0.22 %*	+0.21 %*	-2.99 %***	+0.07 %	+5.11 %***
ZDT2	20	0.50	-0.64 %	-1.44 %	-94.86 %***	-5.55 %	+92.99 %***
		0.75	+0.60 %	+0.67 %	-64.44 %***	-0.47 %	+31.51 %***
		0.95	+0.28 %	+0.16 %	-17.02 %***	+0.17 %	+5.08 %***
	30	0.50	+0.81 %	+3.32 %	-76.60 %***	+1.36 %	+94.78 %***
		0.75	+0.00 %	+1.38 %	-47.23 %***	+2.71 %	+32.43 %***
		0.95	+0.23 %	+0.57 %	-9.18 %***	+0.87 %	+5.11 %***
	40	0.50	+7.72 %	+3.37 %	-37.96 %***	-10.14 %	+99.50 %***
		0.75	+1.78 %	-0.33 %	-20.60 %***	-6.27 %*	+32.52 %***
		0.95	-0.19 %	+0.24 %	-3.85 %***	-0.74 %	+5.17 %***
ZDT3	20	0.50	+1.61 %	+2.80 %*	-94.03 %***	+1.07 %	+84.60 %***
		0.75	+1.68 %**	+2.09 %***	-71.55 %***	+0.57 %	+31.50 %***
		0.95	+0.47 %**	+0.48 %**	-17.25 %***	+0.29 %	+4.89 %***
	30	0.50	+1.83 %	+3.95 %***	-75.88 %***	+0.17 %	+91.89 %***
		0.75	+1.59 %**	+2.46 %***	-51.32 %***	+0.38 %	+32.04 %***
		0.95	+0.09 %	+0.23 %*	-11.75 %***	-0.00 %	+5.07 %***
	40	0.50	+2.70 %**	+2.54 %**	-39.72 %***	-0.67 %	+94.85 %***
		0.75	+1.55 %**	+1.65 %**	-21.83 %***	-0.16 %	+32.44 %***
		0.95	+0.20 %*	+0.03 %	-3.70 %***	-0.10 %	+5.19 %***
ZDT4	5	0.50	-3.33 %	-2.57 %	+4.97 %	-7.81 %	+11.08 %**
		0.75	-3.60 %	-2.67 %	-0.08 %	-4.88 %*	+3.71 %
		0.95	-1.41 %*	-0.54 %	-0.38 %	-1.43 %*	-0.21 %
	10	0.50	-0.31 %	+4.09 %	-32.68 %***	-6.15 %	+1.01 %
		0.75	-0.79 %	+1.11 %	-10.86 %***	-2.91 %	+0.74 %
		0.95	-0.49 %	+0.17 %	-1.48 %**	-0.51 %	+0.16 %
	20	0.50	+1.56 %	+4.62 %	+1.86 %	-3.42 %	+8.23 %
		0.75	+0.29 %	+2.66 %	+2.83 %	+1.88 %	+3.60 %
		0.95	-0.33 %	+0.40 %	+0.19 %	+0.36 %	+0.12 %
ZDT6	5	0.50	+0.16 %	+1.32 %	-23.22 %***	+2.71 %	+95.32 %***
		0.75	-0.09 %	+0.33 %	-9.08 %***	+0.98 %	+32.95 %***
		0.95	+0.02 %	+0.01 %	-3.23 %***	+0.12 %	+4.68 %***
	10	0.50	-0.56 %	+0.81 %	-35.98 %***	+0.50 %	+96.85 %***
		0.75	-0.30 %	+0.11 %	-13.64 %***	+0.30 %	+33.05 %***
		0.95	-0.42 %	+0.03 %	-3.95 %***	-0.06 %	+5.12 %***
	20	0.50	+0.58 %	+1.23 %	-37.03 %***	-0.02 %	+98.09 %***
		0.75	+0.03 %	+0.29 %	-13.25 %***	-0.10 %	+33.15 %***
		0.95	-0.05 %	+0.14 %	-3.37 %***	+0.02 %	+5.11 %***

Table I makes it clear that the different sampling methods affect the convergence behaviour of the algorithm NSGA-III significantly. At first sight, deviations from almost -100 % to +100 % of the normalized HV can be observed. The most obvious thing to notice is the outstanding performance of HS. Using this initialization method, the occurring HV is much higher for almost every considered optimization problem. In particular for lower values of HV_{rel} (and therefore a lower number of generations t) the observed deviations with regard to RS are relatively high. For a larger number of generations (HV_{rel} = 0.95) the deviations become smaller but still greater than the other sampling methods. This observation is also confirmed by Fig. 3. The second-best sampling method for the ZDT problem collection appears to be ILHS. The improvements of the HV compared to HS are lower, but still noticeable with many problems. Also, here the trend is recognizable that with a higher number of generations the differences between ILHS and RS become smaller. LHS performs similar to ILHS, but generally slightly worse. This was expected because ILHS tries to improve LHS with regard to the distribution of the sampled points. In most optimization problems STS performs comparable to RS. Mostly only small improvements or degradations can be observed which are often not statistically significant. The worst convergence behaviour by far can be observed for KS. For almost all optimization problems this sampling method leads to a HV which is much lower than the reference result of RS.

The results lead to the question why HS performs significantly better than the other methods. A characteristic aspect of this sampling method is its quasi-random character [28]. Therefore, always the same initial values are generated which could reduce the spread of the observed results. Hence, Fig. 4 shows the minimal and maximal observed HV of all 200 runs in dependency of the number of generations t (ZDT1, n = 30). As a reference the same graph is shown for RS and KS. It can be seen that HS converges faster and not for any generation t a higher HV is obtained by using RS or KS. This leads to the conclusion that HS seems to be a very good choice for the considered problems, but leaves the underlying effect unanswered.

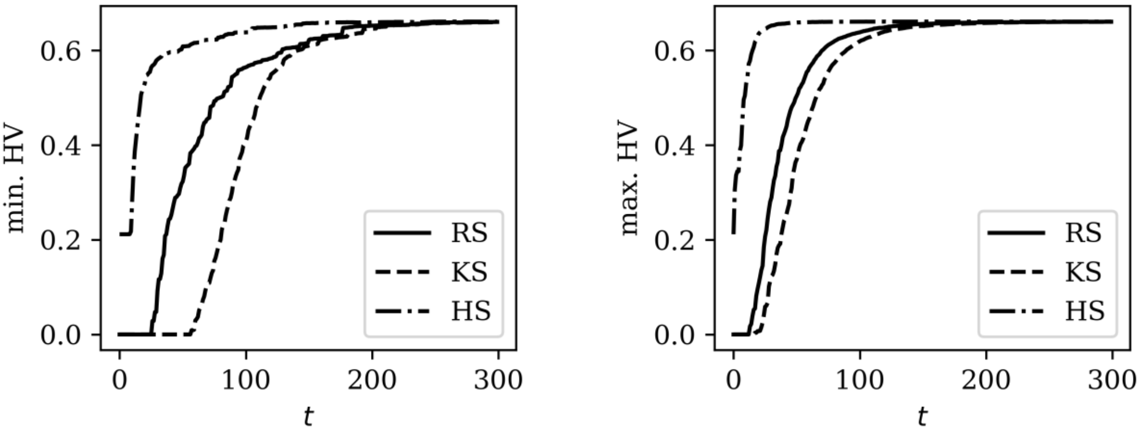


Figure 4: Minimal and maximal HV depending on number of generations t (ZDT1, $n = 30$).

4.2 DTLZ problem collection

The study is performed analogously for the DTLZ problem collection. The results are shown in the following Table II.

The obtained results are generally much more diverse than for the ZDT problem collection. The previously observed performance of HS cannot be reproduced for the DTLZ problem collection. HS does no longer return the best results for almost all optimization problems, instead it often shows a much worse convergence behaviour than the classical RS, for example DTLZ3, n = 20. Like for the ZDT problem collection, the convergence of STS is comparable to RS, whereby the small differences are also statistically not significant. In contrast to the previous section, KS does not perform bad for all optimization problems. For some problems (e. g. DTLZ3, n = 20) even large improvements in comparison to RS are obtained, whereby for other problems (e. g. DTLZ4) the convergence behaviour is way worse. The results of LHS and ILHS are to be emphasized here. These sampling methods lead to a higher HV in many optimization problems which is mainly statistically significant for DTLZ4. Only in rare cases a worsening is observed.

Table II: Results for the DTLZ problem collection.

Problem			**LHS**	**ILHS**	**KS**	**STS**	**HS**
Name	***n***	***HV*rel**					
DTLZ1	5	0.50	+6.99 %	+5.24 %	+25.01 %***	+2.99 %	+28.40 %***
		0.75	+2.94 %	+5.23 %	+8.84 %*	+2.99 %	+14.73 %***
		0.95	-0.83 %	+1.57 %	+1.35 %	+1.20 %	+2.27 %
	7	0.50	+13.86 %	+5.53 %	-48.08 %***	-5.61 %	+19.60 %**
		0.75	+2.23 %	+3.07 %	-26.18 %***	-2.99 %	+5.43 %
		0.95	+2.06 %	+2.13 %	-2.33 %	-0.35 %	+0.89 %
	10	0.50	-1.17 %	-2.76 %	-10.49 %	+1.24 %	-3.16 %
		0.75	-1.06 %	-2.85 %	-3.99 %	-0.33 %	-4.90 %
		0.95	+1.65 %	+0.54 %	+0.56 %	+1.47 %	+0.77 %
DTLZ2	5	0.50	+1.70 %*	+4.35 %***	-8.31 %***	+0.91 %	+10.44 %***
		0.75	+1.58 %***	+1.70 %***	-6.26 %***	+0.57 %	+2.06 %***
		0.95	+0.13 %	+0.18 %*	-0.09 %	+0.08 %	-0.35 %***
	10	0.50	+0.93 %	+2.08 %*	-13.42 %***	+1.73 %	-8.50 %***
		0.75	+0.25 %	+0.52 %	-9.21 %***	+0.15 %	-2.70 %***
		0.95	-0.09 %	-0.03 %	-0.58 %***	+0.07 %	-0.69 %***
	20	0.50	+2.44 %*	+3.74 %***	+28.48 %***	+0.99 %	-3.31 %**
		0.75	+0.61 %	+1.42 %**	+8.40 %***	+0.25 %	-0.74 %
		0.95	+0.12 %	+0.16 %*	+1.09 %***	+0.07 %	-0.24 %**
DTLZ3	5	0.50	-9.73 %	-4.38 %	+27.05 %***	+5.30 %	+1.55 %
		0.75	-4.40 %	-1.29 %	+8.65 %**	+2.63 %	+2.13 %
		0.95	-0.23 %	-0.93 %	+0.42 %	+0.80 %	+1.31 %*
	10	0.50	+3.03 %	+10.49 %	-12.38 %	-7.21 %	-16.93 %*
		0.75	+4.77 %	+5.79 %	-4.98 %	-2.49 %	-1.00 %
		0.95	+1.46 %	+0.95 %	-0.80 %	-0.19 %	+0.37 %
	20	0.50	+20.90 %*	+18.12 %*	+45.79 %***	+12.15 %	-83.30 %***
		0.75	+10.44 %	+4.07 %	+20.21 %***	+5.73 %	-64.79 %***
		0.95	+6.47 %	+3.40 %	+13.13 %***	+2.83 %	-31.42 %***
DTLZ4	5	0.50	+17.04 %***	+13.41 %***	-86.13 %***	+0.16 %	-20.85 %***
		0.75	+13.10 %***	+12.95 %***	-75.58 %***	+0.57 %	-10.69 %***
		0.95	+3.89 %**	+4.91 %***	-60.80 %***	+0.15 %	-6.79 %**
	10	0.50	+19.80 %***	+21.02 %***	-96.55 %***	+3.56 %	-17.58 %***
		0.75	+18.77 %***	+18.26 %***	-95.27 %***	+3.81 %	-8.69 %*
		0.95	+10.65 %***	+9.89 %***	-92.24 %***	+2.51 %	+0.16 %
	20	0.50	+21.50 %***	+21.12 %***	-97.96 %***	+4.14 %	-16.42 %***
		0.75	+21.17 %***	+21.50 %***	-98.34 %***	+5.03 %	-19.74 %***
		0.95	+7.82 %***	+6.77 %**	-97.57 %***	+0.50 %	-7.13 %*

5. CONCLUSION

In this paper it is investigated whether and how various sampling methods of the initial population influence the convergence of multi-objective EA. The analysis was done by using the optimization algorithm NSGA-III, whereby the influence on the HV of the ZDT and the DTLZ problem collections in dependency of the number of generations *t* was conducted. The following sampling methods were considered: Random Sampling (RS), Latin Hypercube Sampling (LHS), Improved Latin Hypercube Sampling (ILHS), *k*-means Sampling (KS), Stratified Sampling (STS) and Halton Sampling (HS).

The obtained results show a significant influence of the sampling method on the convergence behaviour. This influence varies according to the considered optimization problem, whereby for some combinations very large improvements or deteriorations can be achieved, while others perform similar to the classic RS. For example, HS leads to very good results for the ZDT problem collection and mixed results for the DTLZ problem collection. Therefore, a dependency on the optimization problem formulation (number and behaviour of the objective functions, number of decision variables etc.) is presumed. However, from the results can be concluded, that using LHS or ILHS seems to be a good choice if there is no prior knowledge of the optimization problem. For the considered problems, these methods often lead to a slightly better convergence behaviour and a worsening only rarely occurred.

Since this correlation has been demonstrated, it is interesting for following investigations to further examine the crucial factors for selection a sampling method. Additionally, it should be examined why the HV convergence of some methods is faster, which, in addition to better coverage of search space, may also be due to a higher number of non-dominated solutions. This could be an effect for the results of HS and the ZDT problem collection. Furthermore, a more detailed statistical evaluation (e.g. of the significance level and the data distribution) could lead to more comprehensive insights. Also, a different choice of the reference point, like suggested in [32], can be looked at. Furthermore, additional sampling methods, optimization problems and optimization algorithms could be examined to obtain more information about this research field. In addition, a dependency between the sampling method and the evolutionary operators is presumed, which have to be investigated. The authors further examine the influence on technical real-world applications, for example the optimization of mechanical structures and process parameters of the die-casting process based on metamodels.

ACKNOWLEDGEMENT

This publication was funded by the German Research Foundation (DFG) and the University of Bayreuth in the funding programme Open Access Publishing.

REFERENCES

[1] Krimpenis, A.; Benardos, P. G.; Vosniakos, G.-C.; Koukouvitaki, A. (2006). Simulation-based selection of optimum pressure die-casting process parameters using neural nets and genetic algorithms, *The International Journal of Advanced Manufacturing Technology*, Vol. 27, No. 5-6, 509-517, doi:10.1007/s00170-004-2218-0

[2] Klancnik, S.; Brezocnik, M.; Balic, J. (2016). Intelligent CAD/CAM system for programming of CNC machine tools, *International Journal of Simulation Modelling*, Vol. 15, No. 1, 109-120, doi:10.2507/IJSIMM15(1)9.330

[3] Sterpin Valic, G.; Cukor, G.; Jurkovic, Z.; Brezocnik, M. (2019). Multi-criteria optimization of turning of martensitic stainless steel for sustainability, *International Journal of Simulation Modelling,* Vol. 18, No. 4, 632-642, doi:10.2507/IJSIMM18(4)495

[4] Chen, W.; Hao, Y. F. (2018). Genetic algorithm-based design and simulation of manufacturing flow shop scheduling, *International Journal of Simulation Modelling,* Vol. 17, No. 4, 702-711, doi:10.2507/IJSIMM17(4)CO17

[5] Du, Y.; Wang, J. L.; Lei, L. (2019). Multi-objective scheduling of cloud manufacturing resources through the integration of Cat swarm optimization and Firefly algorithm, *Advances in Production Engineering & Management*, Vol. 14, No. 3, 333-342, doi:10.14743/apem2019.3.331

[6] Liu, Y. F.; Zhang, Q. S. (2018). Multi-objective production planning model for equipment manufacturing enterprises with multiple uncertainties in demand, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 429-441, doi:10.14743/apem2018.4.301

[7] Arab, R.; Ghaderi, S. F.; Tavakkoli-Moghaddam, R. (2020). Two efficient meta-heuristic algorithms for the robust inventory routing problem with backhaul, *Technical Gazette*, Vol. 27, No. 3, 793-802, doi:10.17559/TV-20180814091028

[8] Sivadasan, J.; Willjuice Iruthayarajan, M. (2018). Tuning of nonlinear PID controller for TRMS using evolutionary computation methods, *Technical Gazette*, Vol. 25, No. 1, 105-111, doi:10.17559/TV-20170612090511

[9] Harzheim, L. (2019). *Strukturoptimierung: Grundlagen und Anwendungen*, 3rd edition, Europa-Lehrmittel, Haan

[10] Coello Coello, C. A.; Lamont, G. B.; van Veldhuizen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd edition, Springer, New York, doi:10.1007/978-0-387-36797-2

[11] Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*, Wiley, Hoboken

[12] Deb, K. (2011). Multi-objective optimization using evolutionary algorithms: an introduction, Wang, L.; Ng, A. H. C.; Deb, K. (Eds.), *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, Springer, New York, 3-34, doi:10.1007/978-0-85729-652-8_1

[13] Maaranen, H.; Miettinen, K.; Penttinen, A. (2007). On initial populations of a genetic algorithm for continuous optimization problems, *Journal of Global Optimization*, Vol. 37, No. 3, 405-436, doi:10.1007/s10898-006-9056-6

[14] Hernandez-Diaz, A. G.; Coello Coello, C. A.; Perez, F.; Caballero, R.; Molina, J.; Santana-Quintero, L. V. (2008). Seeding the initial population of a multi-objective evolutionary algorithm using gradient-based information, *Proceedings of the 2008 IEEE Congress on Evolutionary Computation*, 1617-1624, doi:10.1109/CEC.2008.4631008

[15] Poles, S.; Fu, Y.; Rigoni, E. (2009). The effect of initial population sampling on the convergence of multi-objective genetic algorithms, Barichard, V.; Ehrgott, M.; Gandibleux, X.; T'Kindt, V. (Eds.), *Multiobjective Programming and Goal Programming*, Springer, Berlin, 123-133, doi:10.1007/978-3-540-85646-7_12

[16] Friedrich, T.; Wagner, M. (2015). Seeding the initial population of multi-objective evolutionary algorithms: A computational study, *Applied Soft Computing*, Vol. 33, 223-230, doi:10.1016/j.asoc.2015.04.043

[17] Hamdan, M.; Qudah, O. (2015). The initialization of evolutionary multi-objective optimization algorithms, Tan, Y.; Shi, Y.; Buarque, F.; Gelbukh, A.; Das, S.; Engelbrecht, A. (Eds.), *Advances in Swarm and Computational Intelligence*, Springer, Cham, 495-504, doi:10.1007/978-3-319-20466-6_52

[18] Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, 182-197, doi:10.1109/4235.996017

[19] Deb, K.; Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints, *IEEE Transactions on Evolutionary Computation*, Vol. 18, No. 4, 577-601, doi:10.1109/TEVC.2013.2281535

[20] Jain, H.; Deb, K. (2014). An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, Part II: Handling constraints and extending to an adaptive approach, *IEEE Transactions on Evolutionary Computation*, Vol. 18, No. 4, 602-622, doi:10.1109/TEVC.2013.2281534

[21] Kazimipour, B.; Li, X.; Qin, A. K. (2014). A review of population initialization techniques for evolutionary algorithms, *Proceedings of the 2014 IEEE Congress on Evolutionary Computation*, 2585-2592, doi:10.1109/CEC.2014.6900618

[22] Wessing, S. Diversipy 0.8 Documentation, from *https://www.simonwessing.de/diversipy/doc/*, accessed on 30-10-2020

[23] Wessing, S. (2015). *Two-stage Methods for Multimodal Optimization*, PhD Thesis, Technische Universität Dortmund, Dortmund, doi:10.17877/DE290R-7804

[24] McKay, M. D.; Beckman, R. J.; Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics*, Vol. 21, No. 2, 239-245, doi:10.1080/00401706.1979.10489755

[25] Beachkofski, B. K.; Grandhi, R. V. (2002). Improved distributed hypercube sampling, *Proceedings of the 43rd AIAA/ASME/AS-CE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 7 pages, doi:10.2514/6.2002-1274

[26] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations, *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 281-297

[27] Wessing, S. (2018). Experimental analysis of a generalized stratified sampling algorithm for hypercubes, *arXiv*, Paper 1705.03809v2, 23 pages

[28] Niederreiter, H. (1992). *Random Number Generation and Quasi-Monte Carlo Methods*, Society for Industrial and Applied Mathematics, Philadelphia, doi:10.1137/1.9781611970081

[29] Riquelme, N.; von Lücken, C.; Baran, B. (2015). Performance metrics in multi-objective optimization, *Proceedings of the 2015 Latin American Computing Conference*, 1-11, doi:10.1109/CLEI.2015.7360024

[30] Zitzler, E.; Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 4, 257-271, doi:10.1109/4235.797969

[31] Zitzler, E. (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*, PhD Thesis, ETH Zürich, Zürich

[32] Ishibuchi, H.; Imada, R.; Setoguchi, Y.; Nojima, Y. (2018). How to specify a reference point in hypervolume calculation for fair performance comparison, *Evolutionary Computation*, Vol. 26, No. 3, 411-440, doi:10.1162/evco_a_00226

[33] Zitzler, E.; Deb, K.; Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: empirical results, *Evolutionary Computation*, Vol 8, No. 2, 173-195, doi:10.1162/106365600568202

[34] Deb, K.; Thiele, L.; Laumanns, M.; Zitzler, E. (2002). Scalable multi-objective optimization test problems, *Proceedings of the 2002 Congress on Evolutionary Computation*, 825-830, doi:10.1109/CEC.2002.1007032

[35] Blank, J.; Deb, K. (2020). Pymoo: multi-objective optimization in Python, *IEEE Access*, Vol. 8, 89497-89509, doi:10.1109/ACCESS.2020.2990567

[36] The SciPy community. Scipy.stats.ttest_ind – SciPy v1.5.2 Reference Guide, from *https://docs.scipy.org/doc/scipy-1.5.2/reference/generated/scipy.stats.ttest_ind.html*, accessed on 30-10-2020