1st International Conference on Frontiers in Academic Research

ICFAR

https://www.icfarconf.com/

February 18-21, 2023, Konya, Turkey

Staircase Re-sampling Technique as an Independent Strategy for Noisy Optimization Problems

O. Tolga Altinoz*

[1]*Department of Electrical and Electronics Engineering, Ankara University, Turkey*

**(taltinoz@ankara.edu.tr) Email of the corresponding author*

***Abstract* –** Engineering problems generally based on collecting and processing of the data taken from real-life problems. Even the data represents the real-word data quantitative and qualitative properties, the data contains noise -measurement noise or disturbance-. To get an accurate solution for this noisy optimization problem some additional techniques are formed inside the optimization algorithm to handle the noise. For this reason, in this research staircase-like dynamic re-sampling method is proposed. This technique is based on increase the amount of re-calculation of the objective functions as the generations goes on. The impact of this technique on the benchmark problems are empirically demonstrated by using four different optimization algorithms and proposed dynamic re-sampling method is compared with the static re-sampling method. The results show that the impact of the proposed algorithm increases as the problem becomes more complex and harder to solve with respect to the computational resources.

Keywords – Noisy Benchmark Problems, Evolutionary Algorithm, Optimization, Multiobjective Optimization, Noise

I. INTRODUCTION

Real word engineering problems generally contains control, design or planning of the processes which are formulated as optimization problems so that the best possible solution or solution set generated for the decision maker to select the most suitable solution among the set. These or similar engineering problems are generated by using the model of the corresponding process, and these models generated from the measurements which are taken from the observation of the systems. Since the measurements and systems contains measurement noise and disturbance, the objective of the optimization problem contains noise. For these reasons the attempts to solve these problems are called noisy optimization problems.

Unlike single objective optimization problems, in multiobjective optimization problem the noise may be added to all objectives or some of the objectives. Hence, noisy multiobjective optimization problems consider as a separate case. Traditional gradient-based/derivative-based optimization techniques could not be very suitable for noisy problems due to the inherent discontinuity of the noisy objectives [1].

In [10] variable noise is defined to the CEC 2009 benchmark set. The noise variance is depended on the objective space position of the solution candidate [10]. It is showed that noiseless objectives can be accurately estimated by using relatively large number of computational sources. Also, small noise values help to correct the noisy problem. As dynamic re-sampling proposal in [11] density plus technique was proposed. By using the crowding principle (noise is normal distribution

with 0.20 standard deviation). The3re are many possible proposals for dynamic re-sampling which are find and discussed in [12,13]. Two types of noise are proposed in [14] and [15] as proportionate noise ($1+\eta_M$) and multiplicative noise with the additive noise which is used in this research and in [10,11,12,13].

In nosy optimization problems the objective functions are contaminated where due to the noisy sensor data at the engineering problem are added to the objective function [1]. Since in many problems, it is not possible to get the noise-statistical properties, it is difficult the get the solutions [1]. The noisy multiobjective optimization is defined as

$$F_{noisy} = \{f_1 + \eta_1, f_2 + \eta_2, \dots, f_M + \eta_M\} \quad (1)$$

where additive noise (h) is applied to each objective value. In this research the noise is modelled as Normal distribution with zero mean and 0.15 standard deviation. This random number changes for each function evaluations.

This paper is organized as four section begins with the introduction. Then optimization algorithms benchmark problems and proposed technique is explained in section 2. In section 3 implementation results are given as empirical study and finally the conclusion of this research presented.

II. METHODS

In this section first the optimization algorithms are explained briefly so that the reader can be look at their references for more detailed. Then benchmark problems are given as mathematical formulations and their properties will explain. Next the proposed re-sampling method is given.

A. *Optimization Algorithms*

Four multiobjective optimization algorithms are selected (evolutionary algorithms -genetic operator-base) to solve the problem and the development environment for the proposed technique. These algorithms MOEA/DDYTS [2], NSGA-II [5], NSGA-III [6], and MOEA/D [7] are selected as the optimizer of this research since the performance of the algorithm on BT benchmark problem may not reported yet. The algorithms and proposed technique were imported in the PlatEmo platform [3,4].

B. *Adaptive operator selection based MOEA/D (MOEA/DDYTS [2])*

Adaptive operator selection based MOEA/D is a variant of the MOEAD algorithm such that a new mechanism is design to adaptively choose the appropriate operator for balancing exploration and exploitation by using dynamic Thompson sampling according to the latest search dynamic. As the crossover (recombination) operator pool differential evolution formulations are discussed. Two parameters of the beta distribution are used to decide the operator by changing the values of these two parameters.

C. *Nondominated Sorting Genetic Algorithm II (NSGA-II [5])*

Nondominated sorting Genetic Algorithm (NSGA-II) is an evolutionary algorithm so that it contains crossover mutation and selection operators. The important part of the NSGA-II algorithm is the operator called nondominated sorting. After the offspring is generated and by using the mutation operator they are changed. After that the two populations are sorted and based on the sorting operator rand value is assigned to the members. The lower ranked members survived to the next generator. For the remaining members by using the operator called crowding distance, are added to the surviving member list.

D. *Nondominated Sorting Genetic Algorithm (NSGA-III [6])*

NSGA-III is an improved version of the NSGA-II (actually R-NSGA-II algorithm) algorithm especially for the many objective optimization problems since as the number of objectives increased in number the desired computational resources for the sorting operator is became huge. The reference points are generated at the beginning of the algorithm. After recombination (crossover) and mutation operators helps to generate offspring. Next all population is classified into different ranks like NSGA-II. In the algorithm for each reference point a solution is expected therefore a traditional selection operator is not needed for NSGA-III algorithm. Beginning with the lowest rank, each member in the rank normalized and associated with the reference point with respect to the distance to the reference vector. The shortest distance member for each reference point is survived to the next generator.

Table 1. The IGD metric values for BT Benchmark problem (noiseless)

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1	2	30	1.5192e+0 (7.90e-1) -	6.1466e-3 (1.11e-3) +	1.1687e-2 (2.73e-3) +	7.9691e-2 (4.87e-2)
BT2	2	30	2.6390e-1 (1.98e-2) -	5.6744e-2 (1.18e-2) +	1.0935e-1 (2.24e-2) -	8.9486e-2 (1.54e-2)
BT3	2	30	1.0491e+0 (4.09e-1) -	5.4295e-3 (2.52e-4) +	1.5641e-2 (3.73e-3) +	3.7424e-2 (2.12e-2)
BT4	2	30	4.4631e-1 (2.67e-1) -	1.0114e-2 (1.54e-3) +	1.9218e-2 (2.66e-3) +	2.8172e-2 (9.09e-3)
BT5	2	30	1.5751e+0 (2.21e-1) -	8.6810e-3 (1.06e-2) +	1.2506e-2 (1.06e-2) +	2.3559e-1 (8.22e-2)
BT6	2	30	6.0239e-1 (2.24e-6) -	2.9216e-1 (5.38e-2) =	3.1322e-1 (1.82e-2) +	3.3142e-1 (1.11e-2)
BT7	2	30	1.5207e-1 (9.65e-2) +	1.7788e-1 (1.08e-1) +	1.8235e-1 (5.00e-2) +	3.3118e-1 (1.45e-1)
BT8	2	30	2.6308e-1 (1.77e-1) =	3.1091e-1 (2.73e-2) +	3.1519e-1 (2.03e-2) +	3.8512e-1 (8.75e-2)
+/-/=			1/6/1	7/0/1	7/1/0	

Table 2. The IGD metric values for BT Benchmark noisy problem solving with proposed Case 1 technique.

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1N4	2	30	3.6021e+0 (3.26e-1) +	3.3096e+0 (1.10e-1) +	3.0792e+0 (3.79e-1) +	4.0564e+0 (1.80e-1)
BT2N4	2	30	1.2554e+0 (2.03e-1) =	8.3323e-1 (9.62e-2) +	6.9088e-1 (5.85e-2) +	1.6987e+0 (5.12e-1)
BT3N4	2	30	3.0221e+0 (3.37e-1) +	1.9461e+0 (8.68e-1) +	9.2588e-1 (3.30e-1) +	4.1118e+0 (3.16e-1)
BT4N4	2	30	3.2578e+0 (4.35e-1) +	1.6620e+0 (6.23e-1) +	8.6780e-1 (5.74e-1) +	3.6743e+0 (3.67e-1)
BT5N4	2	30	3.5499e+0 (8.28e-2) +	3.3097e+0 (8.28e-2) +	3.2343e+0 (1.37e-1) +	3.8778e+0 (1.79e-1)
BT6N4	2	30	9.5050e-1 (4.41e-2) -	4.3728e-1 (2.01e-1) =	4.5412e-1 (1.68e-1) =	7.7371e-1 (7.04e-1)
BT7N4	2	30	4.9791e-1 (1.19e-1) -	3.5830e-1 (9.88e-2) =	3.5343e-1 (1.15e-1) =	3.6130e-1 (1.55e-1)
BT8N4	2	30	9.9738e-1 (1.09e-1) +	4.9603e-1 (2.23e-1) +	4.2368e-1 (1.76e-1) +	3.4423e+0 (7.88e-1)
+/-/=			5/2/1	6/0/2	6/0/2	

E. A multiobjective evolutionary algorithm based on decomposition (MOEA/D [7])

A multiobjective evolutionary algorithm based on decomposition (MOEA/D) is a multiobjective evaluation algorithm that is depended on decomposition as the selection operator. Like other evolutionary algorithms, MOEAD begins with the crossover operator and SBX is selected as this operator. The offspring is generated. Then by using the polynomial mutation the members of the population alter. As the most important and distinguish property of the MOEAD, the3 best members are selected by using the decomposition operator. The decomposition is the aggregation operation to fragment the multiobjective optimization problem into single objective many optimization problems by using the weight vector. Randomly selected members in the neighborhood compares by using the decomposition and best members survives to the next generator.

F. Noisy Benchmark Problems

Benchmark problems are a set of mathematical problems and their solutions are known. These problems are proposed to text and evaluate the performance of the proposed algorithms and it makes easy to compare algorithms even they are presented different research. For these reasons the benchmark problem complexity must be closer to

Table 3. The IGD metric values for BT Benchmark noisy problem solving with proposed Case 2 technique.

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1N8	2	30	3.4576e+0 (2.79e-1) +	3.1980e+0 (4.51e-1) +	2.7597e+0 (7.92e-1) +	4.0897e+0 (1.86e-1)
BT2N8	2	30	1.0530e+0 (7.93e-2) =	7.7472e-1 (2.35e-1) +	6.2429e-1 (1.09e-1) +	1.6192e+0 (7.87e-1)
BT3N8	2	30	3.0491e+0 (4.26e-1) +	9.5499e-1 (7.96e-1) +	6.5458e-1 (4.77e-1) +	3.6647e+0 (7.69e-1)
BT4N8	2	30	2.7330e+0 (4.35e-1) +	8.7299e-1 (6.20e-1) +	3.0786e-1 (1.78e-1) +	3.6030e+0 (9.67e-1)
BT5N8	2	30	3.5272e+0 (1.74e-1) +	3.1689e+0 (3.19e-1) +	2.8744e+0 (3.83e-1) +	4.0553e+0 (2.44e-1)
BT6N8	2	30	8.1099e-1 (2.23e-1) -	2.8618e-1 (8.93e-2) =	2.8511e-1 (7.05e-2) =	5.1583e-1 (5.87e-1)
BT7N8	2	30	4.4447e-1 (1.36e-1) -	2.6483e-1 (1.40e-1) =	3.6992e-1 (1.53e-1) =	3.0744e-1 (1.34e-1)
BT8N8	2	30	9.6535e-1 (3.64e-2) +	6.0795e-1 (8.92e-1) +	4.6539e-1 (3.86e-1) +	2.2651e+0 (7.09e-1)
+/-/=			5/2/1	6/0/2	6/0/2	

Table 4. The IGD metric values for BT Benchmark noisy problem-solving with $5x10^5$ function evaluations

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1N	2	30	3.2138e+0 (4.20e-2) -	2.8378e+0 (1.22e-1) -	2.6890e+0 (2.27e-1) -	6.7222e-1 (1.77e-1)
BT2N	2	30	9.4747e-1 (1.10e-1) =	3.8105e-1 (1.18e-1) +	3.5062e-1 (9.08e-2) +	1.0247e+0 (1.38e-1)
BT3N	2	30	3.0561e+0 (2.87e-1) +	1.7737e+0 (4.53e-1) +	9.6840e-1 (4.54e-1) +	3.7816e+0 (1.32e-1)
BT4N	2	30	3.2086e+0 (1.40e-1) =	1.4732e+0 (3.99e-1) +	9.9270e-1 (3.93e-1) +	3.3383e+0 (3.78e-1)
BT5N	2	30	3.2090e+0 (7.50e-2) +	2.7762e+0 (2.99e-1) +	2.5803e+0 (2.86e-1) +	3.6614e+0 (1.06e-1)
BT6N	2	30	9.0893e-1 (2.55e-1) -	6.5331e-1 (1.16e-1) -	7.1734e-1 (1.53e-1) -	3.8280e-1 (1.94e-1)
BT7N	2	30	7.0744e-1 (1.71e-1) -	6.3077e-1 (9.97e-2) -	6.0303e-1 (1.18e-1) =	4.9322e-1 (9.83e-2)
BT8N	2	30	9.6516e-1 (2.31e-1) +	6.8341e-1 (3.94e-1) +	6.0293e-1 (4.04e-1) +	2.5292e+0 (8.64e-1)
+/-/=			3/3/2	5/3/0	5/2/1	

the real-life engineering problems. To make benchmark problems relatively complex one method is to add bias to the existing problems. In [9], two types of biases are added, and corresponding benchmark problems are called as BT problems. AS discussed, and defined in [9], these benchmark problems are good testing environment for bi-objective -multiobjective- problems. In this research eight BT problems (BT1-BT8) are used. Since these benchmark problems are not noisy, the random number generator is added to the problems as given in Eq. 1. As the noise normal -Gaussian- distribution with zero mean and 0.15 standard deviation is added to make all objective functions noisy.

G. *Proposed Dynamic Re-sampling Technique*

Re-sampling method is a method which is based on calculating functions more than one and taking the average of their value. By this way the noise in the data can be reduced. The static re-sampling method is based on calculating the objective at every generation. However, from the previous results it is indicated and proved that at the early stage of the convergency of the optimization algorithm the error due to the noisy data can be neglectable and has a slightly impact on the performance of the algorithm. Therefore, in this research a new technique named as Staircase Dynamic Re-sampling method is proposed.

Table 5. The IGD metric values for BT Benchmark noisy problem-solving with $8.75x10^5$ function evaluations

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1N	2	30	3.1895e+0 (4.84e-2) -	2.8502e+0 (2.54e-1) -	2.5780e+0 (3.80e-1) -	6.7222e-1 (1.77e-1)
BT2N	2	30	8.5682e-1 (9.87e-2) +	3.3956e-1 (7.15e-2) +	2.0822e-1 (6.39e-2) +	1.0247e+0 (1.38e-1)
BT3N	2	30	2.9562e+0 (2.17e-1) +	1.0031e+0 (4.63e-1) +	6.3742e-1 (3.40e-1) +	3.7816e+0 (1.32e-1)
BT4N	2	30	3.0840e+0 (1.83e-1) =	1.3424e+0 (5.08e-1) +	7.8418e-1 (3.56e-1) +	3.3383e+0 (3.78e-1)
BT5N	2	30	3.1505e+0 (1.03e-1) +	2.6361e+0 (1.65e-1) +	2.5582e+0 (2.58e-1) +	3.6614e+0 (1.06e-1)
BT6N	2	30	8.2742e-1 (2.64e-1) -	7.2956e-1 (9.30e-2) -	8.6126e-1 (1.05e-1) -	3.8630e-1 (1.02e-1)
BT7N	2	30	7.1696e-1 (1.57e-1) =	7.1148e-1 (1.89e-1) =	7.2411e-1 (1.37e-1) =	6.6102e-1 (1.97e-1)
BT8N	2	30	1.0398e+0 (8.83e-2) +	4.7828e-1 (2.38e-1) +	4.6624e-1 (2.32e-1) +	2.4534e+0 (5.67e-1)
+/-/=			4/2/2	5/2/1	5/2/1	

Table 6.The IGD metric values for BT Benchmark noisy problem-solving with static re-sampling with Sample Size=4

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1NS3	2	30	3.5613e+0 (5.44e-2) +	2.8757e+0 (2.89e-1) +	2.4393e+0 (5.16e-1) +	3.7010e+0 (1.03e-1)
BT2NS3	2	30	1.1514e+0 (6.69e-2) -	5.1386e-1 (6.82e-2) +	4.1883e-1 (5.78e-2) +	9.6348e-1 (1.25e-1)
BT3NS3	2	30	3.0817e+0 (3.02e-1) +	9.9743e-1 (5.02e-1) +	9.0012e-1 (4.92e-1) +	3.5068e+0 (4.60e-1)
BT4NS3	2	30	3.0343e+0 (1.62e-1) =	1.0823e+0 (3.40e-1) +	6.6251e-1 (1.52e-1) +	2.9299e+0 (3.98e-1)
BT5NS3	2	30	3.4822e+0 (1.36e-1) +	2.7616e+0 (3.02e-1) +	2.3440e+0 (3.32e-1) +	3.7644e+0 (1.43e-1)
BT6NS3	2	30	5.5515e-1 (1.80e-1) -	4.0141e-1 (1.43e-1) =	4.4856e-1 (5.67e-2) -	3.0607e-1 (7.65e-2)
BT7NS3	2	30	3.6202e-1 (8.62e-2) =	3.1151e-1 (1.02e-1) +	3.0357e-1 (1.57e-1) +	4.5526e-1 (1.40e-1)
BT8NS3	2	30	7.5368e-1 (2.90e-2) +	4.3747e-1 (2.99e-1) +	3.5964e-1 (1.36e-1) +	2.9824e+0 (7.41e-1)
+/-/=			4/2/2	7/0/1	7/1/0	

The idea is based on increase the sampling size as the generation increase. Figure 1 and Figure 2 show this idea.

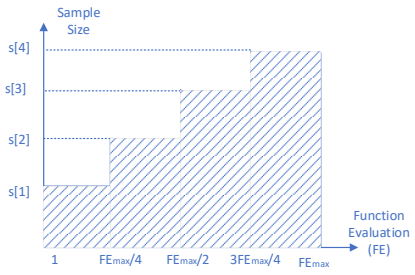

Figure 1. Proposed Staircase Dynamic Re-sampling method – Case 1

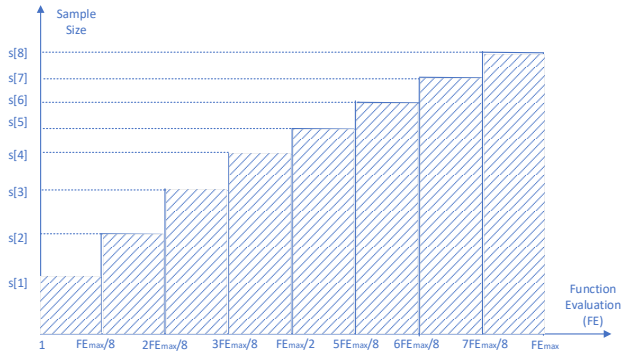

Figure 2. Proposed Staircase Dynamic Re-sampling method – Case 2

Table 7.The IGD metric values for BT Benchmark noisy problem-solving with static re-sampling with Sample Size=8

Problem	M	D	MOEADDYTS	NSGAII	NSGAIII	MOEAD
BT1NS7	2	30	3.4173e+0 (1.81e-1) =	2.2015e+0 (3.18e-1) +	1.8832e+0 (3.30e-1) +	3.5239e+0 (3.29e-1)
BT2NS7	2	30	1.0811e+0 (4.95e-2) -	4.4977e-1 (6.55e-2) +	3.6000e-1 (5.97e-2) +	8.3908e-1 (9.33e-2)
BT3NS7	2	30	2.8932e+0 (2.59e-1) =	4.0697e-1 (2.66e-1) +	1.9671e-1 (1.47e-1) +	2.5594e+0 (5.81e-1)
BT4NS7	2	30	2.7903e+0 (3.15e-1) -	4.9656e-1 (2.91e-1) +	3.8188e-1 (2.11e-1) +	2.3066e+0 (2.24e-1)
BT5NS7	2	30	3.3998e+0 (1.49e-1) =	2.1172e+0 (3.61e-1) +	1.8702e+0 (3.84e-1) +	3.5730e+0 (2.08e-1)
BT6NS7	2	30	6.1112e-1 (1.10e-1) -	2.7198e-1 (6.75e-2) =	3.1449e-1 (1.01e-1) =	2.8583e-1 (4.70e-2)
BT7NS7	2	30	2.8032e-1 (9.63e-2) +	2.6102e-1 (9.56e-2) +	2.7300e-1 (1.33e-1) +	4.0511e-1 (1.29e-1)
BT8NS7	2	30	6.6061e-1 (7.54e-3) +	4.3668e-1 (3.64e-1) +	2.9335e-1 (4.64e-2) +	2.2606e+0 (6.20e-1)
+/-/=			2/3/3	7/0/1	7/0/1	

Figure 1 and 2 gives the two different cases considered in this paper. In Figure 1 (Case 1) The maximum number of function evaluation is divided into four stages. Each state several samplings is calculated (s={0,1,2,3}). Similarly Figure 2 gives for Case 2 for eight different stages (s={0,1,2,3,4,5,6,7}). By this way the aim is to reduce the number of additional function evaluation to get rid of the noise.

III. IMPLEMENTATION

The aim of this research is to empirically demonstrate the performance of the proposed technique by implementing is to solve eight benchmark problems on four different optimization algorithms. The framework for this research begins with the comparing the four algorithms on two objective BT problems (noiseless). First, the algorithms are compared, and the results are reported in Table 1 for BT benchmark problems.

In Table 1, the performance of the algorithms on BT benchmark problems are demonstrated by using the IGD metric (for maximum function evaluation (FEmax)= $2x10^5$). The results are clearly showed that the NSGAII algorithm presents the best results among all other algorithm since the other algorithms are generally developed and tested on many-objective optimization problems. After that NSGA-III gives promising results among all four algorithms. Next, the proposed staircase technique is applied to the noisy benchmark problems, Table 2 and Table 3 gives the results.

In this research two different cases are considered (case 1 and case 2) and the obtained results for Case 1 and Case 2 given in Table 2 and Table 3, respectively. In the given technique the repeated function calculations are increase in number to 5x105 and 8.75x105 for Case 1 and Case 2, respectively for the same iteration given in Table 1.

Comparing Table 2 and Table 3: These two tables are gives for two different cases. For both of them, NSGAIII gives the best results for both tables. Case 2 (Table 3) gives the better results than Case 1. However, in Case 2 more functions are evaluated to get a better result which is an expected outcome. For this reason, instead of comparing Case 1 and Case 2 it is better to compare dynamic re-sampling method with static methods. Before that, the algorithms without using re-sampling methods with given maximum function evaluations are applied to compare the results. Table 4 and Table 5 gives the results for $5x10^5$ function evaluations and $8.75x10^5$ function evaluations. The aim is to answer the question "Do we need re-sampling method?"

Comparing Table2 (Case 1) and Table 4: İt is not possible to mention about one of them is better from the other. Since the benchmark problem sorted from relatively easy problem to the hardest, almost upper top of the problems in Table 4 is better than Table 2 which means that instead of using re-sampling method and allocate computational resources, evolutionary algorithms give better results. However, as the problem

be3came harder than additional techniques are needed. Similar conclusion can be obtained for Case 2 when comparing Table3 (Case 2) and Table 5.

Next, it is better to compare with static re-sampling method. These two cases are the dynamic resampling method for decreasing the number of function evaluations when compared to the static re-sampling method. For this reason, the static re-sampling is implemented for the given maximum number of function evaluations. Table 6 and Table 7 are given for this purpose.

Finally static re-sampling method is implemented with the same number of function evaluations. Table 6 and Table 7 compared the difference of the static methods with different sample size. As the sample size is increased with the same implementation number the performance of the algorithm is clearly improved for all of the benchmark problems. As the final comparison of this research, if the static resampling method is compared with the proposed method under the same conditions, it is empirically showed that the proposed method gives better results. From this result not only the computational resources but also techniques to improve the performance is needed to get a better performance

IV. CONCLUSION

Dynamic re-sampling method is an important and cost-efficient method that decreases the number of function evaluations and saves the computational resources. In this study a new technique called staircase dynamic re-sampling method to improve the performance of the algorithms. To demonstrate the impact of the proposed technique, it is compared with static re-sampling method and traditional optimization algorithm results. For these reasons four algorithms MOEADDYTS, NSGAII, NSGAIII, MOEAD are applied to the problems. The results indicate that for a relatively simple problems even they are noisy, no additional method or technique is needed for getting good results. However, as the problems became harder with a challenging objective space, the techniques are needed and their impact increases. In addition, the static re-sampling technique looks not efficient for solving noisy problem. As future study different -variable- noise will be defined as the noisy problem and investigate the effect of this problem set.

REFERENCES

[1] P. Rakshit, A. Konar, S. Das, "Noisy evolutionary optimization algorithms – A comprehensive survey," Swarm and Evolutionary Computation, vol. 33, pp. 18-45, 2017.

[2] L. Sun and K. Li, Adaptive operator selection based on dynamic Thompson sampling for MOEA/D, Proceedings of the International Conference on Parallel Problem Solving from Nature, 2020, 271-284.

[3] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]," IEEE Computational Intelligence Magazine, vol. 12, no. 4, pp. 73-87, 2017.

[4] Y. Tian, W. Zhu, X. Zhang, and Y. Jin, "A practical tutorial on solving optimization problems via PlatEMO," Neurocomputing, vol. 518, pp. 190-205, 2023.

[5] K. Deb, A.Pratap, S.Agarwal, T.Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol. 6, no. 2, pp. 182-197, 2002

[6] K. Deb, H. Jain. 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, pp. 577–601, 2014.

[7] Q. Zhang, L. Hui "MOEA/D: A multiobjective evolutionary algorithm based on decomposition." IEEE Transactions on Evolutionary Computation, vol. 11, no.6, pp. 712-731, 2007.

[8] F. Siegmund, Amos H.C, K. Deb, "A Comparative Study of Dynamic Resampling Strategies for Guided Evolutionary Multi-Objective Optimization," Congress on Evolutionary Computation, pp. 1826-1835, 2013.

[9] H. Li, Q. Zhang, J. Deng, "Biased Multiobjective Optimization and Decomposition Algorithm," IEEE Transactions on Cybernetics, vol. 47, no. 1, pp. 52-66, 2017.

[10] J.E.Fieldsend, R.M. Everson, "The Rolling Tide Evolutionary Algorithm: A Multiobjective Optimizer for Noisy Optimization Problems," IEEE Transactions On Evolutionary Computation, vol. 19, no. 1, pp. 103-117, 2015.

[11] L. Siwik, P. Sroka, M. Psiuk, "Flock-based Evolutionary Multi-Agent System in Solving Noisy Multi-Objective Optimization Problems," IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 3404-3412, 2008.

[12] P. Rakshit, A. Konar, and S. Das, "Noisy evolutionary optimization algorithms–a comprehensive survey," Swarm and Evolutionary Computation, vol. 33, pp. 18–45, 2017.

[13] P. Rakshit and A. Konar, Principles in Noisy Optimization. Springer, 2018
[14] D. V. Arnold and H.-G. Beyer“On the benefits of populations for noisy optimization,” Evolutionary Computation, vol. 11, no. 2, pp. 111–127, 2003.
[15] M. Hellwig and H.-G. Beyer, “Evolution under strong noise: A selfadaptive evolution strategy can reach the lower performance bound-the pcCMSA-ES,” in International Conference on Parallel Problem Solving from Nature. Springer, 2016, pp. 26–36.