All Sciences Proceedings
http://as-proceeding.com/

1st International Conference on Trends in Advanced Research

March 4-7, 2023 : Konya, Turkey

ICTAR

Performance of the Swarm-based Multiobjective Optimization Algorithms under Chaotic Noisy Problems

O. Tolga ALTINOZ[1*]

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

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

Abstract – Chaotic systems are non-linear dynamic real-life systems which has randomized nature that cannot be modelled, and chaotic maps are functions that generate a chaotic behavior from a relatively simple formulation. Chaos can be observed at the real-life engineering systems and generally the chaotic behavior of these systems omitted due to the insufficient mathematical tools and irregular nature of the chaotic influence. Since they are existing and can be considered in the engineering system. Chaotic maps can be used to generate random numbers. Because of the chaotic nature of this randomize data it is hard -impossible- to handle these signals. The chaotic maps can be used as noise, and in this research, it is applied to the objective functions to generate chaotic noise, and the problems set is named as chaotic noisy benchmark problems (CNBP). In this research the performance of the swam-based multiobjective optimization algorithms is evaluated and analysis under CNBP. The solution for the question "Can evolutionary algorithms solve CNBP?" will be answered. It is showed after the empirical studies that the Chaotic map-oriented random numbers are relatively hard to handle when compared with Gaussian noise.

Keywords – Optimization, Chaotic Maps, Mutiobjective Optimization, Evolutionary Algorithm, Noisy Problem

I. Introduction

The noise in the form of disturbance and/or measurement noise exits in the real-life engineering problems. Generally, the effect of the noise is neglected, and design continue. The given specifications are not met with the desired performance criteria, the design process is repeated until the desired performance is reached. For this process the noise exists however neglected. The form of the noise is important because if the statistical property of the noise is known it is possible to countermeasure calculations to get rid of the noise. In this case the chaotic map based random number-oriented noise can be hard to discussed due to their unknown nature. For this reason, in this research the Chaotic map -Logistic map- is used to generate noise for the benchmark problems and the performance of the swarm-based algorithms are evaluated and discussed. The noisy multiobjective optimization is defined as

$$F_{noisy} = \{f_1 + r_1, f_2 + r_2, \dots, f_M + r_M\} \quad (1)$$

where noise (r) is added to each objective value (for Case 1 and Case 2 which are going to explained in the next section). In this research two noises are considered which are modelled as Chaotic map and Normal distribution with zero mean and 0.15 standard deviation, respectively.

There are much research can be found in literature that uses Chaotic maps. In [8], Rauf et. Al presented a multi population-based chaotic differential evolution algorithm, and they are applied to CEC 2020 benchmark problems. In the proposed algorithm authors divides the population into two parts and they are initialized with a chaotic-based improvement method where Baker's map and Arnold's Cat Map are preferred in the research and integrated to Differential Evolution update formula. The empirical study shows the impact of the proposal. Similarly, in [9] for structural damage with noisy measurement a chaos game optimization-based model was proposed. In the study, a new algorithm called chaos game optimization was proposed for constraint

Table 1.IGD Metric values for BT benchmark problems

Problem	M	D	CMOPSO	GPSOM	MOPSO	NMPSO	SMPSO
BT1	2	30	2.1436e+0 (3.74e-1) +	3.8545e+0 (2.47e-1) -	4.6296e+0 (1.85e-1) -	2.8739e-2 (1.56e-2) +	3.7139e+0 (5.18e-2)
BT2	2	30	1.4331e-1 (3.80e-2) +	3.0126e+0 (1.27e-1) -	3.1000e+0 (2.27e-1) -	2.2762e-2 (4.69e-3) +	6.8538e-1 (1.38e-1)
BT3	2	30	1.6235e+0 (2.20e-1) +	3.3670e+0 (4.29e-1) =	4.5739e+0 (1.56e-1) -	1.3626e-1 (4.81e-2) +	2.9905e+0 (5.14e-1)
BT4	2	30	1.4734e+0 (3.51e-1) +	3.1212e+0 (4.51e-1) =	4.4146e+0 (1.37e-1) -	7.8084e-2 (1.39e-2) +	3.3661e+0 (3.01e-1)
BT5	2	30	2.5577e+0 (1.70e-1) +	3.5744e+0 (5.58e-1) =	4.5037e+0 (9.65e-2) -	8.7755e-2 (2.12e-2) +	3.7285e+0 (5.16e-2)
BT6	2	30	2.1753e-1 (1.54e-2) +	2.5650e+0 (5.39e-1) -	2.4449e+0 (2.88e-1) -	3.3375e-1 (2.03e-1) =	4.8917e-1 (3.42e-1)
BT7	2	30	2.8478e-1 (2.01e-1) =	2.6504e+0 (1.80e+0) -	2.2044e+0 (3.12e-1) -	2.6785e-1 (2.34e-1) =	4.3942e-1 (6.98e-1)
BT8	2	30	2.3803e+0 (3.06e-1) +	2.9345e+1 (3.25e+0) -	5.2469e+0 (5.93e-1) -	6.6181e-1 (3.15e-1) +	3.7555e+0 (6.03e-1)
+/-/=			7/0/1	0/5/3	0/8/0	6/0/2	

optimization problem where producing the fractals is the main contribution of the chaos optimization.

Another chaos driven differential evolution algorithm is proposed in [10]. The algorithm is used to locate passive targets. The algorithm is applied to CCEC 2014 problem set. The chaotic maps is used as random number generator. Piecewise map, logistic map, sine maps, tent map and iterative maps are used for this purpose. In the proposed algorithm at the beginning of the optimization algorithm chaotic sequence with high values are considered as iteration increases its value decreases to encouraging the solution to explore the search space better. As the same manner the same authors proposed the chaotic-based PSO algorithm to solve the passive target localization [11] and showed that the chaotic map can be used with swam based optimization algorithms. In [12], chaotic map and jellyfish optimization algorithms are joined to form a three-layered optimization method as block cipher algorithm. In the study a new one-dimensional chaotic map is proposed after the investigation of Logistic map and Gompertz map where logistic and logistic Gompertz maps are multiplied.

This paper organized as four sections. After the introduction the optimization algorithm and the benchmark problems are given. Then the implementation results and their evaluation are presented. And finally, the conclusion of the research will be given.

II. Algorithms and Methods

The chaotic random number generation is the main topic of the research and the impact of Chaotic random numbers on the problem and the work/efficiency of the optimization algorithms on these noisy problems will be investigated in this research. For this purpose, five swarm-based optimization algorithms are selected as the toolset. In this subsection these algorithms are explained briefly. The reader can find detailed information on the given references for each algorithm.

A. *A competitive mechanism based multi-objective particle swarm optimizer with fast convergence (CMOPSO [1])*

This algorithm is a Particle Swarm Optimizer (PSO)-based proposal with two new mechanisms called competition based learning and Environmental selection mechanisms. Even this algorithm is a swarm-based algorithm the operators are like evolutionary algorithms such that an additional population is generated, and the best members survived to the next generation. The competition-based learning mechanism uses position and velocity update rules from PSO algorithm. Also, polynomial mutation is used inside the algorithm. In the environmental selection two members are selected randomly and the angle is calculated between these two members and an auxiliary member p so that the smaller angle is

Table 2. Performance evaluation for Case 1: Chaotic Maps – Logistic Maps

Problem	M	D	CMOPSO	GPSOM	MOPSO	NMPSO	SMPSO
BT1CC1	2	30	4.7364e+0 (5.49e-1) =	6.4662e+0 (2.75e-1) -	4.7963e+0 (1.94e-1) -	4.1691e+0 (6.05e-1) +	4.5729e+0 (5.81e-1)
BT2CC1	2	30	1.4662e+0 (6.00e-2) +	5.2954e+0 (1.58e-1) -	3.2047e+0 (1.07e-1) -	1.5672e+0 (9.32e-2) +	2.5374e+0 (2.54e-1)
BT3CC1	2	30	3.8116e+0 (2.26e-1) +	6.6554e+0 (2.40e-1) -	4.7541e+0 (1.05e-1) -	3.2986e+0 (2.04e-1) +	4.4606e+0 (1.58e-1)
BT4CC1	2	30	3.8125e+0 (1.31e-1) +	6.3183e+0 (2.20e-1) -	4.7432e+0 (1.04e-1) -	3.4144e+0 (1.87e-1) +	4.4777e+0 (1.90e-1)
BT5CC1	2	30	3.9791e+0 (5.19e-2) +	6.4063e+0 (2.19e-1) -	4.8048e+0 (9.09e-2) -	3.8892e+0 (1.24e-1) +	4.4705e+0 (1.63e-1)
BT6CC1	2	30	6.6014e-1 (1.70e-1) +	5.3336e-1 (7.52e-2) +	2.7550e+0 (1.77e-1) =	9.8725e-1 (4.61e-1) +	2.9982e+0 (4.28e-1)
BT7CC1	2	30	4.6811e-1 (1.59e-1) +	1.0744e+1 (6.81e+0) -	2.6670e+0 (2.89e-1) -	4.5533e-1 (1.11e-1) +	1.9968e+0 (5.23e-1)
BT8CC1	2	30	3.9285e+0 (3.77e-1) =	5.0058e-1 (9.23e-2) +	6.0479e+0 (4.30e-1) -	3.2032e+0 (1.18e+0) +	4.4786e+0 (7.03e-1)
+/-/=			6/0/2	2/6/0	0/7/1	8/0/0	

better therefore the member who gives the smaller angle survives to the next generation.

B. *Gradient-based particle swarm optimization algorithm (GPSOM [2])*

This algorithm is the multiobjective version of the gradient based PSO algorithm. The algorithm uses the deterministic derivative of the problem based on the local search to find and store the best solution. Like other PSO variants, the same position and velocity update rules are defined in the algorithm. Also, the domination idea is used to select/update the members in the population. The algorithm iterates until it is terminated.

C. *Multiple objective particle swarm optimization (MOPSO [3])*

This algorithm is a PSO based multiobjective optimization algorithm. It begins with the randomly assigned population members. The positions of the members are stored with respect to the nondomination idea. Then the hypercube of the explored space is generated. By using these data, the personal best members are stored (at the first iteration the initial position is the personal best position). The speed and velocity is updated. As the last step the stored data is updated. This process is repeated.

D. *Particle swarm optimization with a balanceable fitness estimation (NMPSO [4])*

In NMPSO algorithm a novel balanced fitness estimation method with velocity update equations is proposed for multiobjective PSO algorithm. The performance of the algorithm is evaluated on DTLZ and WFG problems with 4-10 objectives [4]. The algorithms begin with the randomly assigning positions to the members of the algorithm. The objective values are calculated. Initially the current position is assigned as the personal best position. Next the non-dominated solutions are selected, and the fitness values are calculated. The fitness values are the Euclidean distance from the objective value to the ideal points. Like other many-objective optimizations (especially NSGA-II) the perpendicular distance to the reference vector is calculated. Then the position and velocities are updated, and personal best is updated by using the domination idea. If the new solution is dominated the previous solution and replaced. Finally, the evolutionary search strategy is applied to the non-dominated set and archive is updated. This is repeated until the termination criteria is met.

E. *A PSO-based metaheuristic for multi-objective optimization (SMPSO [5])*

The velocity-oriented PSO-based algorithm is called SMPSO. This algorithm is based on producing new positions when the velocity becomes too high. In addition, from the genetic operator Polynomial Mutation is used in this algorithm. In other words, evolutionary operators are integrated into this PSO-based algorithm. The swarm is initialized with the archive of the leaders. At each generation speed and the position is updated (like crossover operator), then mutation operator is implemented. Based on the indicator index the leaders archive and particles memory (personal best) are updated. The generations are continuing in the SMPSO algorithm.

Table 3. Performance evaluation for Case 2: Random number

Problem	M	D	CMOPSO	GPSOM	MOPSO	NMPSO	SMPSO
BT1CC2	2	30	3.1933e+0 (6.83e-2) +	6.6507e+0 (3.18e-1) -	4.1841e+0 (1.28e-1) -	3.4724e+0 (1.61e-1) +	3.9880e+0 (2.23e-1)
BT2CC2	2	30	8.0256e-1 (1.33e-1) +	5.3753e+0 (4.56e-1) -	2.6851e+0 (1.14e-1) -	1.7154e+0 (2.99e-1) +	2.0667e+0 (1.58e-1)
BT3CC2	2	30	3.3120e+0 (6.17e-2) +	6.7922e+0 (3.12e-1) -	4.3280e+0 (1.31e-1) =	3.5290e+0 (1.46e-1) +	4.2060e+0 (3.93e-1)
BT4CC2	2	30	3.1561e+0 (1.15e-1) +	6.7057e+0 (2.83e-1) -	4.1469e+0 (1.10e-1) =	3.5448e+0 (2.73e-1) +	4.0399e+0 (1.73e-1)
BT5CC2	2	30	3.2295e+0 (8.68e-2) +	6.6651e+0 (3.54e-1) -	4.1823e+0 (1.03e-1) -	3.6766e+0 (3.42e-1) +	3.9824e+0 (2.11e-1)
BT6CC2	2	30	2.5322e-1 (9.67e-2) +	5.5784e-1 (1.38e-1) +	2.2197e+0 (1.45e-1) =	2.1064e+0 (6.57e-1) =	2.3950e+0 (3.88e-1)
BT7CC2	2	30	3.0894e-1 (2.00e-1) +	1.8079e+1 (1.73e+0) -	2.2295e+0 (3.84e-1) -	2.0358e+0 (3.76e-1) =	1.7373e+0 (5.62e-1)
BT8CC2	2	30	3.5420e+0 (2.36e-1) =	7.3472e-1 (5.16e-1) +	5.2625e+0 (4.48e-1) -	4.5133e+0 (6.38e-1) -	3.8336e+0 (4.82e-1)
+/-/=			7/0/1	2/6/0	0/5/3	5/1/2	

F. *Noise*

The Logistic map as a Chaotic map demonstrate the recurrence relation since the next value dependent on the previously generated value. The Logistic maps mathematically demonstrate in Eq. 2.

$$r_{n+1} = 4r_n(1 - r_n) \qquad (2)$$

where r is the Chaotic random number used in this research. The initial value for the r is determined randomly from uniformly random number generator. This Logistic map is used inside the problems as a random number/noise. Also, Gaussian distribution is used to compare the impact of the Chaotic maps so that Gaussian noise with 0.15 standard deviation is given as $f_{new} = f + N(0,0.15)$is also used as the additive noise.

In this research four different cases are considered for comparing and presenting the effect of the random numbers. In Case 1 and Case 2 the random numbers are added to all the objective functions as Chaotic map and random number, respectively. In Case 3 and Case 4 the random numbers are added randomly. If a variable -random number- is smaller than 0.5 (fifty-fifty chance) the random number -or chaotic map- is added to the objective function.

-Case 1: Chaotic Maps – Logistic Maps
-Case 2: Random number
-Case 3: Randomly applied chaotic map
-Case 4: Randomly applied random number

G. *Problems and Metrics*

To demonstrate the performance of the optimization algorithm, a set of solution-known problems called benchmark problems are defined in various papers. However, many of them are not suitable for many-objective problems or very easy so that it is not possible to distinguish algorithms with respect to their performances. Therefore, in this research BT benchmark problem set which is defined in [6] are selected. Since these problems contains bais on objective and decision variables, they are closer to the real-life engineering problems with respect to the complexity. In this research eight of BT problems BT1-BT8 is preferred. The index of BT indicated the complexity of the problem. These problems are two objective problems with 30 decision variables. In the next section the results are presented and discussed.

Table 4. Performance evaluation for Case 3: Randomly applied chaotic map

Problem	M	D	CMOPSO	GPSOM	MOPSO	NMPSO	SMPSO
BT1CC3	2	30	3.3252e+0 (1.63e-1) +	5.5106e+0 (5.70e-1) -	4.5802e+0 (2.08e-1) -	1.9895e+0 (1.99e-1) +	3.8361e+0 (2.82e-2)
BT2CC3	2	30	7.1315e-1 (4.52e-2) +	4.5040e+0 (6.89e-1) -	3.1256e+0 (1.49e-1) -	3.1911e-1 (3.59e-2) +	1.6269e+0 (1.95e-1)
BT3CC3	2	30	3.0091e+0 (2.08e-1) +	5.3614e+0 (6.74e-1) -	4.4300e+0 (2.15e-1) -	8.5635e-1 (1.70e-1) +	3.8870e+0 (1.52e-1)
BT4CC3	2	30	2.8143e+0 (1.40e-1) +	5.3311e+0 (4.30e-1) -	4.4946e+0 (1.35e-1) -	6.5315e-1 (2.06e-1) +	3.8036e+0 (1.18e-1)
BT5CC3	2	30	3.4854e+0 (1.72e-1) +	5.3980e+0 (7.86e-1) -	4.5274e+0 (1.24e-1) -	1.9389e+0 (1.29e-1) +	3.8677e+0 (4.70e-2)
BT6CC3	2	30	2.9067e-1 (7.23e-2) +	3.9311e-1 (2.88e-3) +	2.5514e+0 (1.91e-1) -	3.3785e-1 (1.79e-1) +	1.9094e+0 (5.43e-1)
BT7CC3	2	30	2.5338e-1 (3.86e-2) +	8.5371e+0 (3.21e+0) -	2.4447e+0 (2.61e-1) -	3.4396e-1 (1.26e-1) +	7.5773e-1 (4.66e-1)
BT8CC3	2	30	3.5095e+0 (3.41e-1) +	3.8859e-1 (9.72e-3) +	5.5575e+0 (3.59e-1) -	2.1057e+0 (6.28e-1) +	4.2126e+0 (5.15e-1)
+/-/=			8/0/0	2/6/0	0/8/0	8/0/0	

III. IMPLEMENTATION

The implementation in this research contains four cases as explained at thew previous section. In Case 1: as random number generator Chaotic map Logistic Map use as the random variable. As the second case zero mean and 0.15 standard deviation Gaussian random number generator is implemented. The other two cases are given for a random number at a random iteration. That means not for every iteration the noise is added to the objective function. For Case 3 Chaotic map and for Case 4 random number is added to the objective number if the uniformly generated random number smaller than 0.5.

Initially the BT benchmark problems are applied to the five swarm-based multiobjective optimization algorithms. Table 1 gives the results for the implementations. The implementations are evaluated for 100 population size with 2x105 maximum number of function evaluations. The implementations are repeated 10 times -10 independent run- and statistical properties like mean and standard deviation are recorded into the tables with the rank sun statistical test.

In Table 1, it is clearly demonstrated that NMPSO gives the best results among all algorithms, but CMOPSO gives the second-best result. However, NMPSO presents approximately 10 times better results whereas the problem becomes harder the improvement on the algorithm decreases. At BT6, CMPSO gives better result than NMPSO.

Now, the chaotic and random numbers are added to the objectives and the implementations are repeated for 10 times. The result is given numerically in Table 2 by using the IGD metric. Results are much closer now. Still the best performance can be got from NMPSO algorithm. When two tables, Table 1 and Table 2 compared, until BT5 problems the difference is relatively huge so that Table 1 results are almost 100 times better than Table 2. After BT5 the hardest problems are considered and the difference between these two tables decreases. The main reason is that since the problems becomes harder to solve the impact of the optimization algorithms on the harder problems decreases in other words the algorithms could not close enough to the Pareto front. For this reason, the effect of the chaotic maps cannot be observed from the results.

Next the zero mean and 0.15 standard deviation Gaussian noise is added to the objective functions. The results are reported in Table 3. An interesting conclusion is observed from Table 3 so that the best results are all from CMOPSO algorithm where the results are closer to the Table 1 than Table 2. That means the randomness of the Chaotic map is higher than Gaussian values.

Table 5. Performance evaluation for Case 4: Randomly applied random number

Problem	M	D	CMOPSO	GPSOM	MOPSO	NMPSO	SMPSO
BT1CC4	2	30	3.2073e+0 (3.93e-2) +	5.0543e+0 (7.18e-1) -	4.1702e+0 (1.81e-1) =	3.5851e+0 (1.50e-1) +	4.0066e+0 (1.84e-1)
BT2CC4	2	30	8.7747e-1 (7.03e-2) +	3.4172e+0 (6.94e-1) -	2.6938e+0 (1.69e-1) -	1.6725e+0 (3.79e-1) +	2.3037e+0 (3.59e-1)
BT3CC4	2	30	3.2988e+0 (1.39e-1) +	4.9845e+0 (7.17e-1) -	4.3343e+0 (1.30e-1) =	3.6604e+0 (2.66e-1) +	4.1908e+0 (2.39e-1)
BT4CC4	2	30	3.1883e+0 (7.72e-2) +	4.9306e+0 (8.47e-1) -	4.1185e+0 (1.07e-1) =	3.5739e+0 (2.52e-1) +	4.1323e+0 (2.41e-1)
BT5CC4	2	30	3.1804e+0 (1.06e-1) +	4.9260e+0 (1.06e+0) =	4.1213e+0 (1.16e-1) =	3.5406e+0 (2.22e-1) +	4.1388e+0 (2.53e-1)
BT6CC4	2	30	3.7819e-1 (2.47e-1) +	4.6610e-1 (4.60e-2) +	2.3420e+0 (1.94e-1) =	1.7033e+0 (9.96e-1) +	2.5487e+0 (4.48e-1)
BT7CC4	2	30	3.4547e-1 (1.34e-1) +	8.6542e+0 (4.64e+0) -	2.2538e+0 (3.72e-1) -	1.8155e+0 (8.39e-1) =	1.7003e+0 (6.43e-1)
BT8CC4	2	30	3.3886e+0 (4.21e-1) +	5.4136e-1 (1.52e-1) +	5.3199e+0 (5.65e-1) -	4.1671e+0 (1.66e+0) =	3.9131e+0 (3.29e-1)
+/-/=			8/0/0	2/5/1	0/3/5	6/0/2	

The next two cases are given for a randomly added chaotic/random number with a probability of 0.5. If a random value is smaller than 0.5 then chaos/random number is added to the objective value (not all objectives, it is random to add value to the objective function). In Table 4, the performance metric values are demonstrated on Randomly added Chaotic values. Table 4 can be compared with Table2 so that the performance in Table 2 is worse than Table 24which is expected since the chaotic random number always added to the objective (also not all objectives). Therefore, a nosy environment has more effect on a randomly added noise.

The obtained conclusion for the Chaotic maps is not valid for the random number when Table 3 and Table 5 compared with each other the results are almost same for both cases. Therefore, for zero mean Gaussian Noise it is not necessary to get all noisy environment, even randomly added noise changes the flow almost the same manner with respect to the multiobjective optimization idea.

IV. CONCLUSION

From the implementation and the responses getting from the multiobjective optimization the Chaotic map oriented random number changed objective value greatly when compared the Gaussian nose and the impact of the algorithms are lower than Gaussian problems. Also randomly added random numbers also has an effect for Chaotic map cases however do not influences Gaussian random number. Among all the swarm-based algorithms NMPSO and CMOPSO algorithms presents acceptable performance -NMPSO gives better results-. The results suggested that algorithms could not handle Chaotic noise additional tools are needed.

REFERENCES

[1] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, “A competitive mechanism based multi-objective particle swarm optimizer with fast convergence,” Information Sciences, 2018, 427: 63-76.

[2] M. M. Noel, “A new gradient-based particle swarm optimization algorithm for accurate computation of global minimum,” Applied Soft Computing, 2012, 12: 353-359.

[3] C. A. Coello Coello and M. S. Lechuga, “MOPSO: A proposal for multiple objective particle swarm optimization,” Proceedings of the IEEE Congress on Evolutionary Computation, 2002, 1051-1056.

[4] Q. Lin, S. Liu, Q. Zhu, C. Tang, R. Song, J. Chen, C. A. Coello Coello, K. Wong, and J. Zhang, “Particle swarm optimization with a balanceable fitness estimation for many-objective optimization problems,” IEEE Transactions on Evolutionary Computation, 2018, 22(1): 32-46.

[5] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. A. Coello Coello, F. Luna, and E. Alba, “SMPSO: A new PSO-based metaheuristic for multi-objective optimization,” Proceedings of the IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making, 2009, 66-73.

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

[7] M. May, "Simple mathematical models with very complicated dynamics". Nature, vol. 261, no. 5560, pp. 459–467. 1976.

[8] H.T. Fauf, J. Gao, A. Almadhor, A. Haider, Y.D. Zhang, F. Alturjman, “Multi population-based chaotic differential evolution for multi-modal and multi-objective optimization problems,” Applied Soft Computing vol. 132, pp. 109909, 2023.

[9] D.D.Cong, T.N.Thoi, “A chaos game Optimization-based model updating technique for structural damage identification under incomplete noisy measurements and temperature variations,” Structures, vol. 48, pp. 1271–1284, 2023.

[10] M.Rosic, M.Sedak, M. Simiz, P.Pejovic, “An Improved Chaos Driven Hybrid Differential Evolution and Butterfly Optimization Algorithm for Passive Target Localization Using TDOA Measurements,” Applied Sciences, vol. 13, no. 684, pp. 1-38, 2023.

[11] M.Rosic, M.Sedak, M. Simiz, P.Pejovic, “Chaos-Enhanced Adaptive Hybrid Butterfly Particle Swarm Optimization Algorithm for Passive Target Localization,” Sensors, vol. 22, no. 5739, pp. 1-36, 2022.

[12] Y.Su, X.Tong, M.Zhang, Z. Wang, “A new S-box three-layer optimization method and its application,” nonlinear Dynamics, vol. 111, pp. 2841-2867, 2023.