Yalvaç Akademi Dergisi

https://dergipark.org.tr/tr/pub/yalvac

IMPROVEMENT OF BELUGA WHALE OPTIMIZATION ALGORITHM BY DISTANCE BALANCE SELECTION METHOD

[1]*Serdar PAÇACI*

[1]*Isparta Uygulamalı Bilimler Üniversitesi, Teknoloji Fakültesi, Bilgisayar Mühendisliği Bölümü, Isparta*
serdarpacaci@isparta.edu.tr

DOI: 10.5712/yalvac.1257808

Abstract: In this study, an improved version of the Beluga whale optimization (BWO) algorithm, which is a meta-heuristic optimization algorithm recently presented in the literature, is developed to provide better solutions for the problems. The fitness-distance balance (FDB) selection method was applied in the search processes in the BWO algorithm, which was developed by modeling the swimming, preying and falling characteristics of beluga whales. CEC2020 benchmark functions were used to test the performance of the BWO algorithm and the algorithm named FDBBWO. The algorithms were tested on these test functions for 30, 50 and 100 dimensions. Friedman analysis was performed on the test results and the performance ranks of the algorithms were determined. In addition, Wilcoxon rank sum test was used to analyze whether there were significant differences in the results. As a result of the experimental study, it is observed that the BWO algorithm improves the early convergence problem that may arise due to the lack of diversity in the search process. In this way, the possibility of getting stuck at local optimum points is reduced. In addition, the developed algorithm is compared with 3 different algorithms that have been recently presented in the literature. According to the comparison results, FDBBWO has a superior performance compared to other meta-heuristic algorithms. Source code of the proposed FDBBWO algorithm: https://www.mathworks.com/matlabcentral/fileexchange/126400-fdb-bwo

Keywords: Beluga whale optimization, fitness-distance balance, meta-heuristic optimization

---

BEYAZ BALİNA OPTİMİZASYON ALGORİTMASININ UYGUNLUK UZAKLIK DENGESİ SEÇİM YÖNTEMİYLE İYİLEŞTİRİLMESİ

Özet: Bu çalışmada son zamanlarda literatüre sunulmuş bir meta-sezgisel optimizasyon algoritması olan Beyaz balina optimizasyon (Beluga whale optimization, BWO) algoritmasının problemlere daha uygun sonuçlar üretmesi amacıyla iyileştirilmiş bir versiyonu geliştirilmiştir. Beyaz balinaların yüzme, avlanma ve ölme özellikleri modellenerek geliştirilmiş olan BWO algoritmasında yer alan arama süreçlerinde uygunluk uzaklık dengesi (fitness-distance balance, FDB) seçim yöntemi uygulanmıştır. BWO algoritması ve FDBBWO ismi verilerek geliştirilen algoritmanın performanslarını test etmek için CEC2020 test fonksiyonları kullanılmıştır. Bu test fonksiyonları üzerinde 30, 50 ve 100 boyut için algoritmalar test edilmiştir. Elde edilen test sonuçlarına Friedman analizi yapılarak algoritmaların performans sıraları belirlenmiştir. Ayrıca Wilcoxon sıralı işaret testiyle de sonuçlar üzerinde anlamlı derecede farklılıklar oluşup oluşmadığı incelenmiştir. Deneysel çalışma sonucunda BWO algoritmasının arama sürecindeki çeşitlilik eksikliği sebebiyle ortaya çıkabilecek olan erken yakınsama probleminin iyileştiği gözlemlenmiştir. Bu sayede yerel optimum noktalara takılma olasılığı azaltılmıştır. Ayrıca geliştirilen algoritma literatüre son zamanlarda sunulmuş olan 3 farklı algoritmayla karşılaştırılmıştır. Karşılaştırma sonuçlarına göre FDBBWO, diğer meta-sezgisel algoritmalara göre daha üstün bir performans sergilemektedir.

Anahtar Kelimeler: Beyaz balina optimizasyon algoritması, uygunluk uzaklık dengesi seçimi yöntemi, meta-sezgisel

1. INTRODUCTION

Optimization can be defined as the process of finding the optimal solution from a set of solution to a problem. Mathematically, it is the process of finding the point that makes a function minimum or maximum. In the past, classical mathematical optimization algorithms that make use of derivatives have been used. However, in order to use these algorithms, the problem must be modeled mathematically, and they are not flexible. For this reason, meta-heuristic algorithms are used more and more today [1]. In addition, considering the increasing complexity of real-world optimization problems, the required features of optimization algorithms have also increased. A designed meta-heuristic optimization algorithm should be able to handle multi-modal, non-continuous and non-differentiable optimization problems [2].

Meta-heuristic algorithms can be grouped into 4 main categories according to their inspiration types: evolutionary algorithms [3-4], swarm-based algorithms [1, 4-7], human-based algorithms [8] and physics-based algorithms [9, 10]. Meta-heuristic algorithms try to find the optimum point by using a function called the fitness function, which is prepared according to the problem to be optimized. Two main elements are generally prominent in this search process. The first one is global search. The second is local search. The balanced operation of these two elements improves the performance of the meta-heuristic algorithm. In the global search phase, the variety of solutions produced by the algorithm is increased and the search space is better explored. In this way, the probability of getting stuck at the local optimum point is also reduced. Local search increases the probability of finding the optimum point. In this way, the solution generation quality of the algorithm also increases.

Hundreds of meta-heuristic algorithms have been presented in the literature. The genetic algorithm, inspired by Darwin's theory of evolution [3], the differential evolution algorithm, a simplification of the genetic algorithm [4, 29], particle swarm optimization, inspired by the foraging behavior of flocks of birds [6], the ant colony algorithm, inspired by ants finding the shortest path between anthill and the food source [7], the artificial bee colony algorithm, inspired by the foraging behavior of a swarm of bees [1], the gravitational search algorithm, inspired by Newton's laws of motion [9], The stochastic fractal search algorithm inspired by the diffusion process of fractals [11], the gray wolf algorithm inspired by the hierarchy and hunting behavior of wolves [5], the symbiotic organism search algorithm inspired by the relationships between organisms in the ecosystem [12], and the coronavirus herd immunity optimizer algorithm inspired by the logic of herd immunity as a way to combat the coronavirus pandemic are just a few of them.

Zhong et al. presented the beluga whale optimization algorithm (BWO) in 2022. They modeled the swimming, preying and whale fall processes of beluga whales. Therefore, an algorithm consisting of 3 main parts was presented. The swimming behavior represents the global search feature, and the preying behavior represents the local search feature. The algorithm's inability to increase the diversity of solutions increases the probability of getting stuck at local optimum points. This problem also manifests itself in the BWO algorithm.

When the equations used in most of the meta-heuristic algorithms are examined, it is seen that the algorithms generate new solutions by using the position information of a solution point randomly selected from the population. To improve the performance of the algorithm, Kahraman et al. proposed the fitness distance balance method by considering that the selection of a solution point within a certain logic framework instead of this random selection can improve the performance of the algorithm [15]. In summary, a selection process is performed using the locations of the solution points in the population and the fitness values obtained against these solution points. Using this method, it has been observed that the performance of many meta-heuristic algorithms in generating suitable solution points is improved [16-24].

In this study, the FDBBWO algorithm is proposed by using the FDB selection method in 3 different sections to be used in the equations in the beluga whale algorithm. In the original algorithm, the FDB selection method was used instead of the solution points selected from the population with the roulette wheel method. In this context, the first version was created by modifying the equations modeling swimming behavior, the second version by modifying the equations modeling preying behavior, and the third version by modifying the equation modeling whale fall. The algorithms were tested in 30, 50 and 100 dimensions using the CEC2020 test function set [25]. Friedman analysis was applied on the results obtained by the algorithms and the performance rankings of the algorithms were obtained. In addition, Wilcoxon rank sum test was used to examine whether there is a significant difference between the results obtained by the algorithms. According to the results obtained, it was concluded that the third method obtained better results in all dimensions. It is also observed that the premature convergence problem of the BWO algorithm is reduced and the diversity in solution point generation is increased.

In addition, the developed FDBBWO algorithm was compared with the mud ring algorithm [26], prairie dog optimization algorithm [27] and coati optimization algorithm [28] presented in the literature in 2022 and 2023. According to the results obtained, the FDBBWO algorithm ranks first again.

This paper is organized as follows. In the second section, the beluga whale optimization algorithm, the fitness distance balance selection method, and the types of FDB-based algorithms developed are mentioned. In the third section, the benchmark problem set and experimental study settings are described, and the analysis results are given. The last section, the fourth section, presents the conclusion of the study.

2. MATERIAL AND METHOD

In this section, the beluga whale optimization algorithm is introduced in detail, the fitness distance balance selection method and the developed FDBBWO algorithm are explained respectively.

2.1. Beluga Whale Optimization Algorithm

The beluga whale optimization (BWO) algorithm was created by modeling the swimming, preying and whale fall behavior of beluga whales [14]. The algorithm is divided into 3 parts: global search, local search, and whale fall. A variable called the balance factor is used to choose between global and local search. The balance factor is calculated as given in Equation 1. In the equation, T is the current iteration, T_{Max} is the maximum iteration and B_0 is a random number generated in the interval (0,1).

$$B_f = B_0(1 - T / 2T_{\max}) \tag{1}$$

If the calculated B_f value is greater than 0.5, a global search is performed, otherwise a local search is performed.

In the global search section, the swimming behavior of beluga whales is modeled. Two different equations are defined for the new position of the whale during the swimming behavior. If the number of parameters to be optimized is less than or equal to one fifth of the population size, new solution points are found according to the expression given in Equation 2, otherwise new solution points are found as given in Equation 3.

$$\begin{aligned} X_{i,p_1}^{T+1} &= X_{i,p_1}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_2}^{T}\right)\left(1+r_1\right)\sin\left(2\pi r_2\right) \\ X_{i,p_2}^{T+1} &= X_{i,p_2}^{T} + \left(X_{r,p_2}^{T} - X_{i,p_2}^{T}\right)\left(1+r_1\right)\sin\left(2\pi r_2\right) \end{aligned} \tag{2}$$

$$X_{i,j}^{T+1} = \begin{cases} X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)\left(1+r_1\right)\sin\left(2\pi r_2\right) & \text{if } j \text{ is even} \\ X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)\left(1+r_1\right)\cos\left(2\pi r_2\right) & \text{if } j \text{ is odd} \end{cases} \tag{3}$$

T in the equations denotes the current iteration, p_j is a randomly chosen dimension value according to the dimension information of the optimization problem, r_1 and r_2 are a random value in the range (0 1).

For modeling the local search partition, the preying behavior of beluga whales was used. Whales can move and hunt based on the location of nearby whales. In other words, they hunt by sharing location information with each other. Levy Flight, which is frequently used in the literature, is added to the algorithm and it is assumed that they can catch the prey in this way. Local search is performed using the expression given in Equation 4. In the equation, T is the current iteration, X_{best} is the position of the whale with the best fitness value so far, X_r is the position of a whale randomly selected from the population, r_3 and r_4 are randomly selected values in the range (0 1). In addition, L_f is the value found according to Levy flight and is calculated as given in Equation 5. C_1 is a parameter that adjusts the jump intensity of the Levy flight and is calculated by the equation $2r4\ (1-T/T_{max})$.

$$X_i^{T+1} = r_3 X_{best}^{T} - r_4 X_i^{T} + C_1 L_F \left(X_r^{T} - X_i^{T}\right) \tag{4}$$

In the Levy flight equation, u and v are random values and β is defined as a constant of 1.5. σ is calculated according to Equation 6.

$$L_F = 0.05 \frac{u\,\sigma}{|v|^{1/\beta}} \tag{5}$$

$$\sigma = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma((1+\beta)/2)\,\beta\, 2^{(\beta-1)/2}} \right)^{1/\beta} \tag{6}$$

In their search for food, beluga whales are threatened by animals at the top of the food pyramid. Whales can escape threats by sharing information with each other. But this is not always possible. When they die for any reason, they actually become a source of food for many creatures living in the sea and an ecosystem is formed in the environment. To model this phenomenon, firstly find the probability of this happening. If the probability is realized, the whale is removed from the population. However, a new whale is added to the population to keep the population size constant. This is done as given in Equation 7. In the equation, r_5, r_6 and r_7 are random values in the range (0 1) and X_{step} is the step size.

$$X_i^{T+1} = r_5 X_i^T - r_6 X_r^T + r_7 X_{step} \tag{7}$$

X_{step} is calculated as given in Equation 8. *ub* and *lb* are the upper and lower bounds of the optimized parameters respectively, *T* is the current iteration and T_{max} is the maximum iteration, respectively.

$$X_{step} = \left(u_b - l_b\right)\exp\left(-(2W_f\ n)T / T_{\max}\right) \tag{8}$$

W_f in Equation 8 is defined as a function decrease linearly from 0.1 to 0.05 as given in Equation 9.

$$W_f = 0.1 - 0.05T / T_{\max} \tag{9}$$

2.2. Fitness Distance Balance Selection Method

When the equations used in meta-heuristic algorithms in the literature are examined, it is seen that in many algorithms, the process of selecting the solution candidate from the population randomly or with the best fitness value is encountered. The performance of the algorithm will be improved if this candidate solution is selected in such a way that it can contribute to the solution candidate to approach the optimum point. With this motivation, the fitness distance balance (FDB) method is a selection method developed by Kahraman et al. [15]. FDB performs the selection process by considering two feature values: the value obtained from the fitness function of each solution candidate and the Euclidean distance to the solution point with the best fitness value in the current population.

To use it in the algorithm, first the positions of the solution points in the population and the fitness values of each solution point must be calculated. Then the position of the solution point with the best fitness value is found. Then, the distance values of each solution point to the solution point with the best fitness value are found as given in Equation 10. In the equation, P_{best} is the solution candidate with the best fitness value, P_i is the solution point whose distance is to be calculated, and d is the number of parameters to be optimized.

$$D_{p_i} = \sqrt{\sum_{j=1}^{d}\left(P_{i,j} - P_{best,j}\right)^2} \tag{10}$$

When the expression given in Equation 10 is calculated for each individual, the vector D_p, which expresses the Euclidean distances of each solution point in the population to the solution P_{best}, is formed as shown in Equation 11. The expression *n* in the equation refers to the population size.

$$D_p = \begin{bmatrix} D_{P_1} \\ . \\ . \\ D_{P_n} \end{bmatrix} \tag{11}$$

While calculating the FDB scores for each solution point, the conformity values obtained by the solution points with the created D_p vector is used. However, these values must be normalized before they are used. For this reason, the values in the D_p vector is normalized using Equation 12.

$$normD_p = \frac{D_p - \min(D_p)}{\max(D_p)} \tag{12}$$

Similarly, the values obtained by the solution points using the fitness function are normalized according to the expression given in Equation 13. The expression F in the equation is the vector containing the fitness values.

$$normF = \frac{F - \min(F)}{\max(F)} \tag{13}$$

The expression given in Equation 14 is used to calculate the FDB scores. In the equation, w is a coefficient that adjusts the effects of the relevance value and distance value on the FDB score. When the studies conducted with FDB in the literature are examined, it is seen that the coefficient w is generally taken as 0.5.

$$S_{P_i} = w\,normF_i + (1-w)\,normD_{P_i} \tag{14}$$

After calculating the vector S_p containing the FDB scores, the selection process is completed by selecting the solution point with the highest FDB score. In this last step, probabilistic methods can also be used instead of selecting the one with the highest FDB score. In this study, after obtaining the vector S_p, the roulette wheel method was used to make the selection.

2.3. FDBBWO Algorithm

Three different cases were evaluated for the application of the selection method with FDB instead of the selection methods in the BWO algorithm. In the first case, X_{FDB} was used instead of X_r in Equations 2 and 3 in the original BWO algorithm, where the swimming behavior of the beluga whale was modeled. In the second case, X_{FDB} was used instead of X_r in Equation 4, where the preying behavior of beluga whales is modeled. In the third and final case, where the whale falls and a new whale is created, X_{FDB} is used instead of X_r in Equation 7. The cases created for the FDBBWO algorithm are given in Table 1. In addition, in the global search part of the original algorithm, the rule "if the number of parameters to be optimized is less than or equal to one-fifth of the population size, new solution points are found according to the expression given in Equation 2, otherwise new solution points are found as given in Equation 3" is changed. randomly generated values in the interval (0 1) called r_8 and r_9 are generated. If r_8 is greater than r_9, new solution points are calculated according to Equation 3, otherwise new solution points are calculated according to according to Equation 2.

Tablo 1. Equations used for original BWO and FDBBWO cases.

BWO with FDB cases		
BWO	Swimming	$X_{i,j}^{T+1} = \begin{cases} X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\sin(2\pi r_2) & if\ j\ is\ even \\ X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\cos(2\pi r_2) & if\ j\ is\ odd \end{cases}$ *or* $X_{i,p_1}^{T+1} = X_{i,p_1}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ $X_{i,p_2}^{T+1} = X_{i,p_2}^{T} + \left(X_{r,p_2}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$
	Preying	$X_{i}^{T+1} = r_3 X_{best}^{T} - r_4 X_{i}^{T} + C_1 L_F \left(X_{r}^{T} - X_{i}^{T}\right)$
	Fall	$X_{i}^{T+1} = r_5 X_{i}^{T} - r_6 X_{r}^{T} + r_7 X_{step}$
Case 1 FDB	Swimming	$X_{i,j}^{T+1} = \begin{cases} X_{i,p_j}^{T} + \left(X_{FDB,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\sin(2\pi r_2) & if\ j\ is\ even \\ X_{i,p_j}^{T} + \left(X_{FDB,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\cos(2\pi r_2) & if\ j\ is\ odd \end{cases}$ $r_8 > r_9$ $X_{i,p_1}^{T+1} = X_{i,p_1}^{T} + \left(X_{FDB,p_1}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ $X_{i,p_2}^{T+1} = X_{i,p_2}^{T} + \left(X_{FDB,p_2}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ *otherwise*
	Preying	$X_{i}^{T+1} = r_3 X_{best}^{T} - r_4 X_{i}^{T} + C_1 L_F \left(X_{r}^{T} - X_{i}^{T}\right)$
	Fall	$X_{i}^{T+1} = r_5 X_{i}^{T} - r_6 X_{r}^{T} + r_7 X_{step}$
Case 2 FDB	Swimming	$X_{i,j}^{T+1} = \begin{cases} X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\sin(2\pi r_2) & if\ j\ is\ even \\ X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\cos(2\pi r_2) & if\ j\ is\ odd \end{cases}$ $r_8 > r_9$ $X_{i,p_1}^{T+1} = X_{i,p_1}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ $X_{i,p_2}^{T+1} = X_{i,p_2}^{T} + \left(X_{r,p_2}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ *otherwise*
	Preying	$X_{i}^{T+1} = r_3 X_{best}^{T} - r_4 X_{i}^{T} + C_1 L_F \left(X_{FDB}^{T} - X_{i}^{T}\right)$
	Fall	$X_{i}^{T+1} = r_5 X_{i}^{T} - r_6 X_{r}^{T} + r_7 X_{step}$
Case 3 FDB	Swimming	$X_{i,j}^{T+1} = \begin{cases} X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\sin(2\pi r_2) & if\ j\ is\ even \\ X_{i,p_j}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_j}^{T}\right)(1+r_1)\cos(2\pi r_2) & if\ j\ is\ odd \end{cases}$ $r_8 > r_9$ $X_{i,p_1}^{T+1} = X_{i,p_1}^{T} + \left(X_{r,p_1}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ $X_{i,p_2}^{T+1} = X_{i,p_2}^{T} + \left(X_{r,p_2}^{T} - X_{i,p_2}^{T}\right)(1+r_1)\sin(2\pi r_2)$ *otherwise*
	Preying	$X_{i}^{T+1} = r_3 X_{best}^{T} - r_4 X_{i}^{T} + C_1 L_F \left(X_{r}^{T} - X_{i}^{T}\right)$
	Fall	$X_{i}^{T+1} = r_5 X_{i}^{T} - r_6 X_{FDB}^{T} + r_7 X_{step}$

The pseudocode of the FDBBWO algorithm is given in Figure 1.

```
Initialize the population
Evalute fitness values and find the best solution
While Fes<MaxFes Do
    Calculate the $W_f$ and $B_f$ values
    For each whale ($X_i$) Do
        If $B_f$ > 0.5
            Generate the random indexes ($P_j$)
            If FDBCase1 then generate $X_r$ using FDB Else generate $X_r$ using Roulette wheel selection
            Choice $r_8$ and $r_9$ randomly
            If $r_8 < r_9$
                Generate new solution point using Equation 2
            Else
                Generate new solution point using Equation 3
            End
        Else
            Calculate $C_1$ and $L_F$
            If FDBCase2 then generate $X_r$ using FDB Else generate $X_r$ using Roulette wheel selection
            Generate new solution point using Equation 4
        End
        Check the boundaries of the new solution point
        Evalute the fitness value
        Fes = Fes +1
    End For
    For each whale ($X_i$) Do
        If $B_f <= W_f$
            Calculate $C_2$
            Calculate $X_{step}$
            If FDBCase3 then generate $X_r$ using FDB Else generate $X_r$ using Roulette wheel selection
            Generate new solution point using Equation 7
            Check the boundaries of the new solution point
            Evalute the fitness value
            Fes = Fes +1
        End
    End
    Find the best solution
End While
```

Figure 1. The pseudocode of the FDBBWO algorithm

3. EXPERIMENTAL STUDY And DISCUSSIONS

In this section first describes the benchmark functions and experimental settings. Then, the results of the analysis performed on the data obtained because of the test are given. Finally, the results obtained by the proposed algorithm are compared with three meta-heuristic algorithms presented in the literature in 2022 and 2023.

3.1. Benchmark Functions and Experimental Study Settings

The designed FDBBWO algorithm was tested using the benchmark functions frequently used in the literature and defined for the CEC2020 conference [25]. This benchmark function set includes 10 different unconstrained optimization problems. The search space is chosen between -100 and 100. The 1st function is designed to detect the local search properties of the tested algorithms. The 2nd, 3rd and 4th functions are used to detect the global search properties of the algorithms. Functions 5, 6 and 7 are designed to determine whether the local and global search properties of the algorithms are in balance and functions 8, 9 and 10 are designed to determine the algorithm performance for problems with high complexity.

The population size of the BWO algorithm is taken as 50 as the algorithm is presented. Each benchmark function was tested for 30, 50 and 100 dimensions. In addition, each algorithm was run independently 51 times for each function. Since meta-heuristic algorithms involve stochastic processes, the same result may not be obtained every time the algorithm is run. For this reason, it is not possible to run the algorithm once and evaluate the algorithm. For this reason, as stated in the CEC2020 benchmark function definition document, the algorithm was run 51 times for each test function. In order to fairly compare the compared algorithms, each algorithm was allowed to use the fitness function 10000 x d times, where d is the problem dimension. The experimental studies were carried out in MATLAB 2020a on a computer with Intel Core i5-CPU @ 2.90 GHz, 8 GB RAM and Windows 10 operating system.

In order to compare the results obtained by the developed algorithm with other algorithms, the mud ring algorithm (MRA), inspired by the hunting strategies of dolphins with mud rings presented in the literature in 2022 [26], the prairie dog optimization algorithm developed by imitating the behavior of prairie dogs (PDO) presented in the literature in 2022 [27], and the kayoti optimization algorithm (COA), inspired by the behavior of kayoti, a raccoon-like animal presented in the literature in 2023 [28], were selected. The default parameters of the algorithms were used as algorithm parameters. The population size was taken as 30 for MRA, 100 for PDO and 30 for COA.

3.2. Analysis Results of FDBBWO Algorithm

In this section, the results obtained by the original BWO algorithm and 3 FDBBWO versions on benchmark functions are analyzed. Friedman analysis method is used to compare the performance of the algorithms and rank them according to their performance. In addition, Wilcoxon rank sum test was used to determine whether there is a significant difference between the results obtained by the algorithms. The significant difference rate was taken as 5%.

The mean and standard deviation values of the error values obtained by the algorithms in 30, 50 and 100 dimensions for 10 benchmark functions are given in Table 2. When the table is analyzed, it is seen that FDBBWO algorithms obtained lower error values in average error values. Only in the 4th function, all algorithms found the optimum value and the error values were 0.

Table 2. The mean and standard deviation values of the error values of the algorithms on the CEC2020 test functions for 30, 50 and 100 dimensions

		30D		**50D**		**100D**	
Benchmark Function	**Algorithm**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**
F1	BWO	4,48E+10	4,35E+09	9,62E+10	3,68E+09	2,43E+11	4,95E+09
	FDBBWO Case 1	3,48E+08	7,37E+07	8,72E+08	1,22E+08	2,36E+09	2,50E+08
	FDBBWO Case 2	3,33E+08	7,16E+07	8,99E+08	1,37E+08	2,37E+09	2,60E+08
	FDBBWO Case 3	3,23E+08	6,75E+07	8,80E+08	1,68E+08	2,34E+09	3,00E+08

Table 2. The mean and standard deviation values of the error values of the algorithms on the CEC2020 test functions for 30, 50 and 100 dimensions (Continued)

F2	BWO	6,90E+03	2,93E+02	1,33E+04	3,91E+02	3,00E+04	7,06E+02
	FDBBWO Case 1	3,24E+03	3,43E+02	6,52E+03	4,92E+02	1,46E+04	9,22E+02
	FDBBWO Case 2	3,17E+03	3,69E+02	6,50E+03	5,81E+02	1,43E+04	7,90E+02
	FDBBWO Case 3	3,20E+03	4,26E+02	6,55E+03	5,61E+02	1,42E+04	9,99E+02
F3	BWO	6,36E+02	2,58E+01	1,18E+03	4,02E+01	3,05E+03	5,78E+01
	FDBBWO Case 1	1,96E+02	1,34E+01	3,99E+02	2,45E+01	1,06E+03	8,63E+01
	FDBBWO Case 2	2,00E+02	1,66E+01	3,95E+02	2,57E+01	1,08E+03	9,63E+01
	FDBBWO Case 3	2,00E+02	1,64E+01	3,94E+02	3,10E+01	1,07E+03	1,03E+02
F4	BWO	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00
	FDBBWO Case 1	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00
	FDBBWO Case 2	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00
	FDBBWO Case 3	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00	0,00E+00
F5	BWO	3,90E+07	1,28E+07	2,32E+08	6,51E+07	9,73E+08	1,38E+08
	FDBBWO Case 1	4,23E+06	1,79E+06	8,19E+06	3,18E+06	3,51E+07	6,10E+06
	FDBBWO Case 2	4,45E+06	2,07E+06	7,23E+06	2,56E+06	3,52E+07	6,40E+06
	FDBBWO Case 3	4,12E+06	1,93E+06	7,78E+06	3,60E+06	3,55E+07	6,21E+06
F6	BWO	2,61E+03	2,64E+02	5,81E+03	4,49E+02	2,07E+04	1,34E+03
	FDBBWO Case 1	3,21E+02	9,12E+01	7,84E+02	1,55E+02	3,10E+03	4,61E+02
	FDBBWO Case 2	2,96E+02	9,31E+01	8,27E+02	1,42E+02	3,06E+03	4,11E+02
	FDBBWO Case 3	3,17E+02	8,77E+01	7,75E+02	1,39E+02	3,11E+03	3,89E+02
F7	BWO	8,98E+06	4,18E+06	3,05E+07	9,93E+06	2,42E+08	3,39E+07
	FDBBWO Case 1	4,76E+05	2,48E+05	3,96E+06	1,51E+06	1,55E+07	2,96E+06
	FDBBWO Case 2	4,88E+05	2,71E+05	4,44E+06	1,82E+06	1,59E+07	2,77E+06
	FDBBWO Case 3	5,46E+05	3,63E+05	4,05E+06	1,67E+06	1,61E+07	2,86E+06
F8	BWO	5,68E+03	4,28E+02	1,38E+04	4,28E+02	3,11E+04	4,64E+02
	FDBBWO Case 1	1,91E+02	1,36E+01	3,66E+03	3,33E+03	1,65E+04	2,44E+03
	FDBBWO Case 2	1,91E+02	1,26E+01	3,26E+03	3,27E+03	1,66E+04	2,52E+03
	FDBBWO Case 3	1,94E+02	1,32E+01	3,38E+03	3,30E+03	1,65E+04	2,55E+03

Table 2. The mean and standard deviation values of the error values of the algorithms on the CEC2020 test functions for 30, 50 and 100 dimensions (Continued)

F9	BWO	1,05E+03	5,96E+01	1,84E+03	8,90E+01	5,83E+03	2,69E+02
	FDBBWO Case 1	5,82E+02	2,07E+01	8,65E+02	3,92E+01	1,55E+03	5,14E+01
	FDBBWO Case 2	5,77E+02	2,08E+01	8,78E+02	4,17E+01	1,54E+03	4,25E+01
	FDBBWO Case 3	5,82E+02	2,41E+01	8,75E+02	4,25E+01	1,53E+03	4,84E+01
F10	BWO	1,67E+03	1,11E+02	1,02E+04	5,67E+02	2,20E+04	9,22E+02
	FDBBWO Case 1	4,41E+02	1,85E+01	7,04E+02	4,48E+01	1,29E+03	6,53E+01
	FDBBWO Case 2	4,45E+02	2,32E+01	7,00E+02	3,02E+01	1,25E+03	5,96E+01
	FDBBWO Case 3	4,43E+02	2,39E+01	7,02E+02	3,95E+01	1,26E+03	7,93E+01

The Friedman analysis results of 3 different FDBBWO models developed in this study and the original BWO algorithm are given in Table 3.

Table 3. Friedman analysis ranking results of BWO and FDBBWO algorithms

Algorithm	30D	50D	100D	Mean Rank
FDBBWO CASE 3	2,0422	2,0147	1,9735	2,0101
FDBBWO CASE 2	2,0441	2,0853	2,0637	2,0644
FDBBWO CASE 1	2,0637	2,0500	2,1127	2,0755
BWO	3,8500	3,8500	3,8500	3,8500

According to the Friedman analysis ranking shown in Table 3, the BWOCase3 version ranked first in all dimensions. Therefore, the BWOCase3 algorithm also ranks first in the average rank value. It is also observed that all three FDBBWO versions presented in this study find more suitable results than the original algorithm. This shows that the selection method with FDB reduces the probability of the original BWO algorithm getting stuck at local optimum points. In other words, FDB has a positive effect on the algorithm.

Since the Wilcoxon rank sum test is a pairwise comparison, each model is compared with the original BWO algorithm. According to the significant difference rate chosen as 5%, it is examined whether there is a difference between the algorithms compared according to the data obtained as a result of 51 studies. When there was a significant difference, it was decided which algorithm gave more favorable results on average and whether they obtained good, similar or bad results. The results obtained are given in Table 4.

Table 4. Comparison results of BWO and FDBBWO algorithms according to Wilcoxon rank sum test

	30D			**50D**			**100D**		
	Better	**Similar**	**Worse**	**Better**	**Similar**	**Worse**	**Better**	**Similar**	**Worse**
FDBBWO CASE 1	9	1	0	9	1	0	9	1	0
FDBBWO CASE 2	9	1	0	9	1	0	9	1	0
FDBBWO CASE 3	9	1	0	9	1	0	9	1	0

When Table 4 is analyzed, it is seen that the algorithms produce similar results in 1 test function for all dimensions. In the other 9 benchmark functions, there was a significant difference between the results produced by the original BWO algorithm and FDB models and FDB models produced better results.

Box-plot plots are used to analyze the global and local search capabilities of the algorithms. The results obtained by the original BWO algorithm and FDB models for 30, 50 and 100 dimensions on 10 benchmark functions are graphicalized. Figure 2 shows box-plot plots for bechmark functions 1 to 5. When the graph is analyzed, it is seen that the algorithms produce similar results in the fourth test function. In the other four functions, the FDB versions produced better results than the original algorithm. When the FDB versions are evaluated within themselves, it is observed that they produce close results to each other.

Figure 3 shows box-plot plots for benchmark functions 6 to 10. In all five test functions, the FDB versions produced more favorable results than the original algorithm. When the FDB versions are evaluated within themselves, it is observed that they produce similar results.

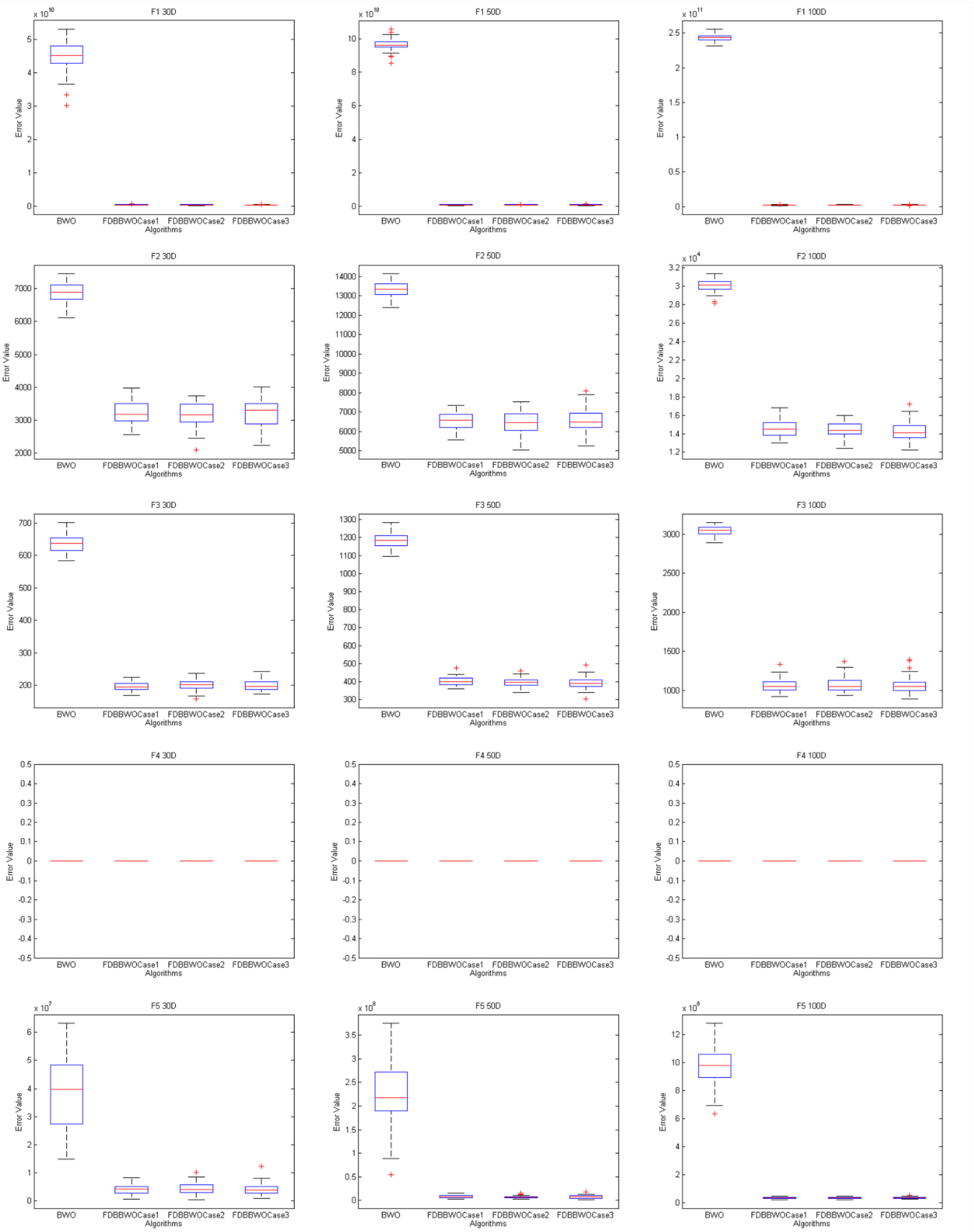

Figure 2. Box-plot characteristics of BWO and FDBBWO algorithms for test functions F1-F5

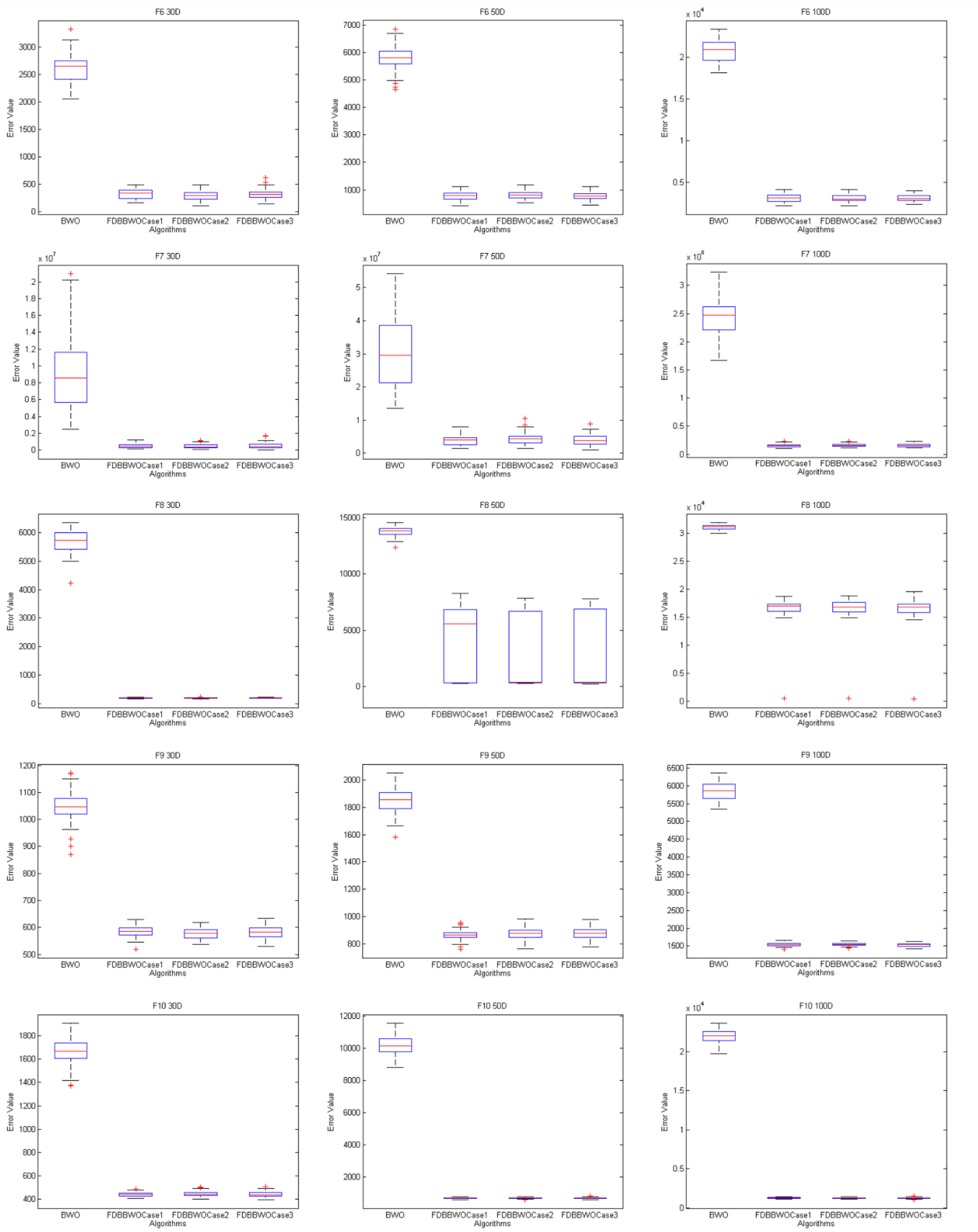

Figure 3. Box-plot characteristics of BWO and FDBBWO algorithms for test functions F6-F10

3.3. Analysis Results of FDBBWO and Other Algorithms

In this section, the results obtained by FDBBWO, COA, MRA and PDO algorithms on CEC2020 benchmark functions are analyzed. The mean and standard deviation values of the error values obtained by the algorithms in 30, 50 and 100 dimensions for 10 benchmark functions are given in Table 5. When the table is analyzed, it is observed that the FDBBWO algorithm obtains lower values as average error values in all dimensions. Only in the 4th function, all algorithms found the optimum value and the error values were 0.

Table 5. The mean and standard deviation values of the error values of the algorithms on the CEC2020 test functions for 30, 50 and 100 dimensions of FDBBWO and other algorithms

		30D		**50D**		**100D**	
Benchmark Function	**Algorithm**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**
F1	FDBBWO	3.23E+08	6.75E+07	8.80E+08	1.68E+08	2.34E+09	3.00E+08
	COA	6.14E+10	9.16E+09	1.19E+11	8.87E+09	2.81E+11	9.24E+09
	MRA	7.97E+10	8.10E+08	1.34E+11	3.03E+08	2.94E+11	4.01E+08
	PDO	3.56E+10	7.88E+09	7.30E+10	9.44E+09	1.60E+11	1.07E+10
F2	FDBBWO	3.20E+03	4.26E+02	6.55E+03	5.61E+02	1.42E+04	9.99E+02
	COA	7.87E+03	2.85E+02	1.51E+04	4.09E+02	3.22E+04	6.37E+02
	MRA	8.88E+03	1.26E+02	1.64E+04	1.64E+02	3.50E+04	5.58E+02
	PDO	6.45E+03	6.59E+02	1.25E+04	8.68E+02	2.79E+04	1.25E+03
F3	FDBBWO	2.00E+02	1.64E+01	3.94E+02	3.10E+01	1.07E+03	1.03E+02
	COA	7.82E+02	4.34E+01	1.43E+03	2.35E+01	3.40E+03	5.60E+01
	MRA	8.33E+02	8.09E+00	1.43E+03	1.52E+01	3.44E+03	1.77E+01
	PDO	6.15E+02	1.38E+02	1.19E+03	1.86E+02	2.92E+03	2.55E+02
F4	FDBBWO	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	COA	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	MRA	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	PDO	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F5	FDBBWO	4.12E+06	1.93E+06	7.78E+06	3.60E+06	3.55E+07	6.21E+06
	COA	1.24E+08	6.41E+07	8.10E+08	3.18E+08	2.16E+09	4.94E+08
	MRA	3.40E+08	8.04E+07	1.70E+09	6.52E+08	3.22E+09	2.43E+08
	PDO	3.13E+07	1.54E+07	1.99E+08	8.26E+07	6.86E+08	2.58E+08
F6	FDBBWO	3.17E+02	8.77E+01	7.75E+02	1.39E+02	3.11E+03	3.89E+02
	COA	4.55E+03	1.30E+03	9.88E+03	2.12E+03	3.43E+04	5.42E+03
	MRA	9.57E+03	3.99E+03	1.69E+04	3.69E+03	4.46E+04	1.73E+03
	PDO	2.40E+03	4.88E+02	5.67E+03	1.20E+03	1.94E+04	2.58E+03
F7	FDBBWO	5.46E+05	3.63E+05	4.05E+06	1.67E+06	1.61E+07	2.86E+06
	COA	4.43E+07	2.83E+07	1.42E+08	8.60E+07	4.72E+08	7.33E+07
	MRA	2.74E+08	1.77E+08	6.43E+08	2.54E+08	6.47E+08	4.28E+06
	PDO	8.50E+06	6.57E+06	3.28E+07	1.76E+07	2.24E+08	6.69E+07
F8	FDBBWO	1.94E+02	1.32E+01	3.38E+03	3.30E+03	1.65E+04	2.55E+03
	COA	7.88E+03	6.78E+02	1.53E+04	4.80E+02	3.35E+04	7.36E+02
	MRA	9.24E+03	1.35E+02	1.69E+04	2.22E+02	3.55E+04	1.85E+02
	PDO	5.39E+03	1.57E+03	1.28E+04	7.17E+02	2.92E+04	1.33E+03
F9	FDBBWO	5.82E+02	2.41E+01	8.75E+02	4.25E+01	1.53E+03	4.84E+01
	COA	1.43E+03	2.35E+02	2.55E+03	5.12E+02	8.61E+03	1.30E+03
	MRA	2.01E+03	2.80E+02	3.59E+03	3.67E+02	1.15E+04	4.78E+02
	PDO	9.73E+02	7.84E+01	1.72E+03	8.12E+01	5.39E+03	2.01E+02

Table 5. The mean and standard deviation values of the error values of the algorithms on the CEC2020 test functions for 30, 50 and 100 dimensions of FDBBWO and other algorithms (Continued)

F10	FDBBWO	4.43E+02	2.39E+01	7.02E+02	3.95E+01	1.26E+03	7.93E+01
	COA	3.26E+03	5.07E+02	1.49E+04	9.82E+02	2.90E+04	1.65E+03
	MRA	4.61E+03	3.34E+01	1.66E+04	5.55E+01	3.21E+04	1.00E+02
	PDO	1.73E+03	3.56E+02	7.71E+03	9.30E+02	1.39E+04	1.50E+03

The Friedman analysis results of the FDBBWO algorithm and the other algorithms compared are given in Table 6.

Table 6. Friedman analysis ranking results of FDBBWO and other algorithms

Algorithm	30D	50D	100D	Mean Rank
FDBBWO	1.1539	1.1500	1.1500	1.1513
PDO	2.0833	2.0951	2.0578	2.0788
COA	2.9382	2.9971	2.9814	2.9722
MRA	3.8245	3.7578	3.8108	3.7977

According to the Friedman analysis ranking shown in Table 6, the FDBBWO version ranked first in all dimensions. Therefore, the FDBBWO algorithm also ranks first in the average rank value. The PDO algorithm ranked second in all dimensions, while COA and MRA ranked third and fourth, respectively.

The results of the Wilcoxon ranked sign test for FDBBWO and the other compared algorithms are given in Table 7.

Table 7. Comparison results of FDBBWO and other algorithms according to Wilcoxon rank sum test

	30D			50D			100D		
	Better	**Similar**	**Worse**	**Better**	**Similar**	**Worse**	**Better**	**Similar**	**Worse**
COA	0	1	9	0	1	9	0	1	9
MRA	0	1	9	0	1	9	0	1	9
PDO	0	1	9	0	1	9	0	1	9

When Table 7 is analyzed, it is seen that the algorithms produce similar results in only 1 test function for all dimensions. In the other 9 test functions, there was a significant difference between the results produced by the FDBBWO algorithm and the other algorithms and all algorithms produced worse results than the FDBBWO model. This confirms that the developed algorithm produces better results.

Figure 4 shows box-plot plots for benchmark functions 1 to 5. When the graph is analyzed, it is seen that the algorithms produce similar results in the fourth test function. In the other four functions, the FDBBWO algorithm produced better results than the original algorithm. Figure 5 shows box-plot plots for benchmark functions 6 to 10. In all five test functions, the FDBBWO algorithm produced more favorable results than the original algorithm.

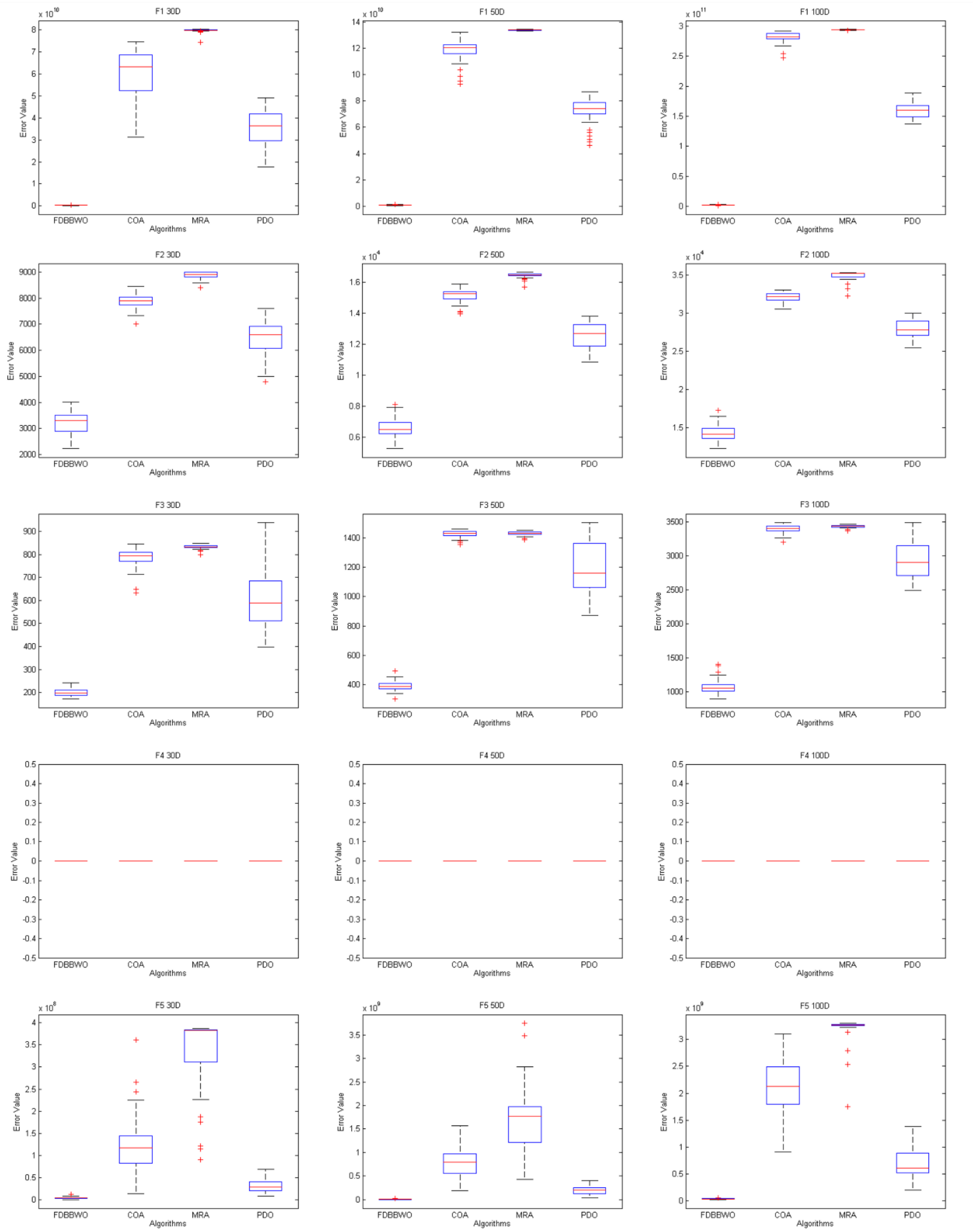

Figure 4. Box-plot characteristics of FDBBWO and other algorithms for test functions F1-F5

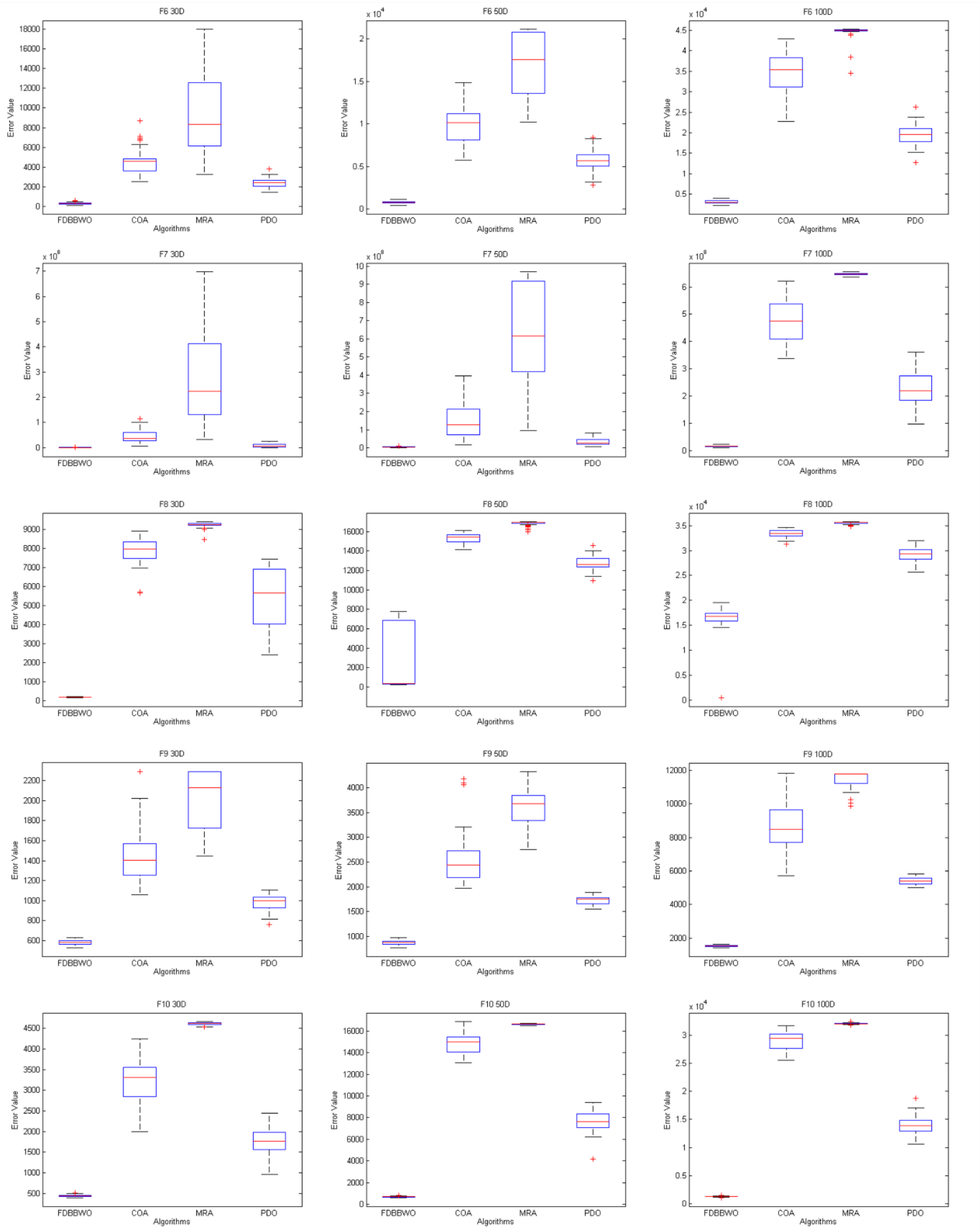

Figure 5. Box-plot characteristics of FDBBWO and other algorithms for test functions F6-F10

4. CONCLUSIONS

In this study, in order to improve the performance of the BWO algorithm, a fitness distance balance selection method is applied to three different parts of the algorithm. In this way, the diversity of solutions produced by the BWO algorithm is increased and the probability of premature convergence problem is reduced. The proposed algorithm is named FDBBWO. CEC2020 benchmark functions were used to test the performance of the developed algorithm.

In simulations for 30, 50 and 100 dimensions, each algorithm was run 51 times independently and the results were recorded. Friedman analysis was performed on the results obtained by the algorithms and the performance ranks of the algorithms were determined. In addition, Wilcoxon ranked sign test was applied for each algorithm in pairs and it was determined whether there was a significant difference. While the absence of significant difference indicates that the solutions produced are similar, if there is a difference, it is concluded that it produces good or bad results. As a result of the experimental analysis, it was observed that the FDBBWO algorithm produced better results in 9 of the 10 test functions and produced similar results in 1 function. In addition, the developed algorithm was compared with the COA, MRA and PDO algorithms presented in the literature in 2022 and 2023 using the same test functions. According to the results obtained, it is observed that the FDBBWO algorithm produces better results than these three algorithms.

In future studies, the developed FDBBWO algorithm will be applied to engineering problems and the results obtained will be analyzed.

REFERENCES

[1] Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization* (Vol. 200, pp. 1-10). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

[2] Del Ser, J., Osaba, E., Molina, D., Yang, X. S., Salcedo-Sanz, S., Camacho, D., Das, S., Suganthan P.N., Coello, C. A. C., and Herrera, F. (2019). Bio-inspired computation: Where we stand and what's next. Swarm and Evolutionary Computation, 48, 220-250.

[3] Golberg D.E., (1989). Genetic Algorithms in Search, Optimization and Machine Learning, Addion Wesley.

[4] Storn, R., and Price, K. (1997). Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), 341.

[5] Mirjalili, S., Mirjalili, S. M., and Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.

[6] Kennedy, J., and Eberhart, R. (1995). Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). IEEE.

[7] Dorigo, M., and Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. In Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406) (Vol. 2, pp. 1470-1477). IEEE.

[8] Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Computer-aided design, 43(3), 303-315.

[9] Rashedi, E., Nezamabadi-Pour, H., and Saryazdi, S. (2009). GSA: a gravitational search algorithm. Information sciences, 179(13), 2232-2248.

[10] Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics-based algorithm. Future Generation Computer Systems, 101, 646-667.

[11] Salimi, H. (2015). Stochastic fractal search: a powerful metaheuristic algorithm. Knowledge-Based Systems, 75, 1-18.

[12] Cheng, M. Y., and Prayogo, D. (2014). Symbiotic organisms search: a new metaheuristic optimization algorithm. Computers & Structures, 139, 98-112.

[13] Al-Betar, M. A., Alyasseri, Z. A. A., Awadallah, M. A., and Abu Doush, I. (2021). Coronavirus herd immunity optimizer (CHIO). Neural Computing and Applications, 33, 5011-5042.

[14] Zhong, C., Li, G., and Meng, Z. (2022). Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. Knowledge-Based Systems, 251, 109215.

[15] Kahraman, H. T., Aras, S., ve Gedikli, E. (2020). Fitness-distance balance (FDB): A new selection method for meta-heuristic search algorithms. Knowledge-Based Systems, 190, 105169.

[16] Guvenc, U., Duman, S., Kahraman, H. T., Aras, S., & Katı, M. (2021). Fitness–Distance Balance based adaptive guided differential evolution algorithm for security-constrained optimal power flow problem incorporating renewable energy sources. *Applied Soft Computing*, *108*, 107421.

[17] Aras, S., Gedikli, E., and Kahraman, H. T. (2021). A novel stochastic fractal search algorithm with fitness-distance balance for global numerical optimization. Swarm and Evolutionary Computation, 61, 100821.

[18] Duman, S., Kahraman, H. T., Guvenc, U., and Aras, S. (2021). Development of a Lévy flight and FDB-based coyote optimization algorithm for global optimization and real-world ACOPF problems. Soft Computing, 25, 6577-6617.

[19] Sharifi, M. R., Akbarifard, S., Qaderi, K., and Madadi, M. R. (2021). Developing MSA algorithm by new fitness-distance-balance selection method to optimize cascade hydropower reservoirs operation. Water Resources Management, 35, 385-406.

[20] Zheng, K., Yuan, X., Xu, Q., Dong, L., Yan, B., and Chen, K. (2022). Hybrid particle swarm optimizer with fitness-distance balance and individual self-exploitation strategies for numerical optimization problems. Information Sciences, 608, 424-452.

[21] Bakir, H., Guvenc, U., Kahraman, H. T., & Duman, S. (2022). Improved Lévy flight distribution algorithm with FDB-based guiding mechanism for AVR system optimal design. Computers & Industrial Engineering, 168, 108032.

[22] Duman, S., Kahraman, H. T., Sonmez, Y., Guvenc, U., Kati, M., and Aras, S. (2022). A powerful meta-heuristic search algorithm for solving global optimization and real-world solar photovoltaic parameter estimation problems. Engineering Applications of Artificial Intelligence, 111, 104763.

[23] Ozkaya, B., Guvenc, U., & Bingol, O. (2022). Fitness Distance Balance based LSHADE algorithm for energy hub economic dispatch problem. IEEE Access, 10, 66770-66796.

[24] Duman, S., Kahraman, H. T., and Kati, M. (2023). Economical operation of modern power grids incorporating uncertainties of renewable energy sources and load demand using the adaptive fitness-distance balance-based stochastic fractal search algorithm. Engineering Applications of Artificial Intelligence, 117, 105501.

[25] C. T. Yue, K. V. Price, P. N. Suganthan, J. J. Liang, M. Z. Ali, B. Y. Qu, N. H. Awad, and Partha P Biswas , Problem Definitions and Evaluation Criteria for the CEC 2020 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization, Technical Report 2019 11, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China And Technical Report, Nanyang Technological University, Singapore

[26] Desuky, A. S., Cifci, M. A., Kausar, S., Hussain, S., and El Bakrawy, L. M. (2022). Mud Ring Algorithm: A new meta-heuristic optimization algorithm for solving mathematical and engineering challenges. IEEE Access, 10, 50448-50466.

[27] Ezugwu, A. E., Agushaka, J. O., Abualigah, L., Mirjalili, S., and Gandomi, A. H. (2022). Prairie dog optimization algorithm. Neural Computing and Applications, 34(22), 20017-20065.

[28] Dehghani, M., Montazeri, Z., Trojovská, E., and Trojovský, P. (2023). Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. Knowledge-Based Systems, 259, 110011.

[29] GÜRFİDAN, R., & ERSOY, M. (2020). A New Hybrid Encryption Approach for Secure Communication: GenComPass. *International Journal of Computer Network & Information Security*, *12*(4).