Mathematical Biosciences and Engineering

MBE, 20(11), 19086–19132.
DOI: 10.3934/mbe.2023844
Received: 03 August 2023
Revised: 23 September 2023
Accepted: 06 October 2023
Published: 13 October 2023

http://www.aimspress.com/journal/MBE

Research article

Modified prairie dog optimization algorithm for global optimization and constrained engineering problems

Huangjing Yu[1], Yuhao Wang[1], Heming Jia[1,*] and Laith Abualigah[2,3,4,5,6,7,8]

[1] School of Information Engineering, Sanming University, Sanming 365004, China
[2] Computer Science Department, Prince Hussein Bin Abdullah Faculty for Information Technology, Al al-Bayt University, Mafraq 25113, Jordan
[3] Department of Electrical and Computer Engineering, Lebanese American University, Byblos 13-5053, Lebanon
[4] Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan
[5] MEU Research Unit, Middle East University, Amman 11831, Jordan
[6] Applied science research center, Applied science private university, Amman 11931, Jordan
[7] School of Computer Sciences, Universiti Sains Malaysia, Pulau Pinang 11800, Malaysia
[8] School of Engineering and Technology, Sunway University Malaysia, Petaling Jaya 27500, Malaysia

* **Correspondence:** Email: jiaheming@fjsmu.edu.cn.

Abstract: The prairie dog optimization (PDO) algorithm is a metaheuristic optimization algorithm that simulates the daily behavior of prairie dogs. The prairie dog groups have a unique mode of information exchange. They divide into several small groups to search for food based on special signals and build caves around the food sources. When encountering natural enemies, they emit different sound signals to remind their companions of the dangers. According to this unique information exchange mode, we propose a randomized audio signal factor to simulate the specific sounds of prairie dogs when encountering different foods or natural enemies. This strategy restores the prairie dog habitat and improves the algorithm's merit-seeking ability. In the initial stage of the algorithm, chaotic tent mapping is also added to initialize the population of prairie dogs and increase population diversity, even use lens opposition-based learning strategy to enhance the algorithm's global exploration ability. To verify the optimization performance of the modified prairie dog optimization algorithm, we applied it to 23 benchmark test functions, IEEE CEC2014 test functions, and six engineering design problems for testing. The experimental results illustrated that the modified prairie dog optimization algorithm has good optimization performance.

Keywords: prairie dog optimization algorithm; audio signal factor, merit-seeking ability, lens opposition-based learning strategy, engineering design problems

1. Introduction

With the rapid development of contemporary science and technology, many practical engineering application problems have become increasingly complex, and the complexity required for their calculations has also gradually increased. When solving engineering application problems, people often do not have a suitable solution to execute. To simplify the complexity of practical problems and reduce energy consumption, metaheuristic algorithms with an optimal solution have attracted more and more attention. Metaheuristic algorithms are heuristic algorithms that simulate the process of a certain natural phenomenon or observe the survival behavior of natural organisms. Due to their high efficiency, strong timeliness, and global convergence, they can often quickly find a feasible solution from unknown spaces when solving nonlinear practical problems. However, due to the constraints and complexity of real-life practical problems, we cannot obtain the optimal solutions for all problems using only one algorithm. Therefore, metaheuristic algorithms based on physics, humans, biological populations, and evolution have been continuously proposed by scholars to solve practical engineering problems.

The inspiration for physics-based metaheuristic algorithms mostly comes from the laws of physics and chemical energy reactions in nature. For example, the gravitational search algorithm (GSA) [1], the rime optimization algorithm (RIME) [2], the simulated annealing (SA) [3], the black hole (BH) [4], the Kepler optimization algorithm (KOA) [5]. Human-based metaheuristic algorithms mainly simulate a series of human behaviors. For example, the teaching learning based optimization (TLBO) [6], the mother optimization algorithm (MOA) [7], the harmony search (HS) [8], the group teaching optimization algorithm (GTOA) [9], the brain storming optimization (BSO) [10]. The metaheuristic algorithm based on biological populations is currently one of the two popular branches, which mainly simulates the social behavior of natural biological populations, including foraging, nesting, and avoiding natural enemies. For example, particle swarm optimization (PSO) [11], the monarch butterfly optimization (MBO) [12], the QoS-based dissemination of content in grids [13], the reorganization and discovery of grid information with epidemic tuning [14], the bio-inspired algorithm for outlier's detection [15], the colony predation algorithm (CPA) [16], the ant colony optimization (ACO) [17], the crayfish optimization algorithm (COA) [18], the Siberian tiger optimization (STO) [19]. The evolution-based metaheuristic algorithm is the other branch of the two popular branches. Its inspiration mainly comes from gene mutation, cross-inheritance, natural selection, and other phenomena in evolutionary biology. For example, genetic algorithm (GA) [20], evolutionary programming (EP) [21], differential evolution (DE) [22], virulence optimization algorithm (VOA) [23], The bio-geography based optimizer (BBO) [24]. Metaheuristic optimization algorithms based on physics, humans, biological populations, and evolution and their inspirations are listed in Table 1.

Table 1. Metaheuristic optimization algorithms.

Classes	Metaheuristic	Inspiration	Date
Physics-based	GSA	The laws of universal gravitation and Newton's second law, which seeks the optimal solution through the interaction between gravity and mass	2009
	RIME	Based on the physical phenomenon of rime-ice, the algorithm is exploration and exploitation by simulating the growth process of rime-ice	2023
	SA	Originate from the principle of solid-state annealing, which raises an object to a very high temperature and then slowly cools it down	1983
	BH	Determine the spin of a black hole by determining the physical size of its innermost stable circular orbit, however, the space-time differences is the main factor between non-spin Schwarzschild black holes and Kerr black holes of the same mass	2013
	KOA	Kepler laws of planetary motion, which predict the velocity and position of planets at any time to find the closest solution to the optimal solution	2023
Human-based	TLBO	Guidance of teachers to students and mutual learning among students	2012
	MOA	The mother's leadership of the child's growth process is simulated, and the algorithm is divided into three stages: education, advice, and upbringing	2023
	HS	Simulate the principle of music performance	2001
	GTOA	Simulate the mechanism of group teaching	2020
	BSO	Simulate the process of humans using creative thinking to solve problems during meetings	2016
Based on biological populations	PSO	Simulate the behavior of birds searching for food in nature	1995
	MBO	Simulated the migration of monarch butterflies thousands of miles to Mexico	2019
	QoS-based dissemination of content in Grids	Inspired by the way of information transmission by ants and termites, a new grid information system is built to reorganize and disseminate information	2008
	Reorganization and discovery of grid information with epidemic tuning	Inspired by the information exchange behavior of ant colonies, resources are discovered by sharing information among groups	2008

Continued on next page

Classes	Metaheuristic	Inspiration	Date
	Bio-inspired algorithm for outliers detection	The similarity between mobile agent and data object is used to detect abnormal data in distributed system	2017
	CPA	Simulate the cooperative predation behavior of animals in nature, dispersing prey and rounding them up	2021
	ACO	Based on the behavior of ants discovering paths while searching for food	2006
	COA	The algorithm was divided into three stages based on the crayfish's summer resort behavior, competition and foraging behavior in response to temperature changes	2023
	STO	The hunting behavior of Siberian tigers in battle was simulated	2022
Evolution-based	GA	Borrow Mendel's genetic theory and Darwin's evolution theory and achieves the selection process of natural selection and survival of the fittest by simulating natural evolution	1992
	EP	Simulate the adaptive behavior of organisms to evolution	2003
	DE	Based on evolutionary ideas such as genetic algorithms and is based on distinct differences within the population	1997
	VOA	Inspired by the virus invasion of human body, the algorithm is exploration and exploitation by simulating the special invasion mechanism of virus	2016
	BBO	Simulate the change of species migration in a habitat	2008

The PDO algorithm [25] is a metaheuristic algorithm based on biological populations proposed in 2022. This algorithm simulates each cluster of prairie dogs' behavior in searching for food, building caves, and preventing natural enemies. Each cluster of prairie dogs has its information exchange mode. During the exploration phase, they continuously search for the best food source to build each family cave. However, during the exploitation phase, due to the influence of natural enemies and food sources, the algorithm easily falls into local optima, reducing its optimization performance. According to the no free lunch (NFL) [26] theorem, no algorithm can solve all optimization problems. Regardless of the algorithm used, at least one objective function enables the algorithm to find the optimal value. Therefore, Liu et al. applied the improved prairie dog optimization (IPDO) algorithm [27] to test its performance in gate recursive unit networks; Nguyen et al. [28] used the PDO algorithm to solve the problem of damage identification in engineering structures; Gürses et al. [29] combined Gaussian mutation and chaos search with PDO to enhance the optimization ability of the algorithm; Abualigah et al. [30] combined the opposition-based Laplacian distribution with the PDO algorithm and applied it to industrial engineering design problems and photovoltaic solar problems.

Whether prairie dogs are searching for food or avoiding natural enemies, they generate an audio signal to find better food resources or evade the pursuit of natural enemies in response to the slow convergence speed of the PDO algorithm and prairie dog's habitual nature. This paper proposes a modified prairie dog optimization (MPDO) algorithm. Adding an audio signal factor to represent the distance between the prairie dog and the food (natural enemy), the prairie dogs adjust their position based on the audio signal's strength and speed to ensure sufficient food and safety. This method is called the frequency wave strategy, effectively improving the algorithm's performance, and the global optimization ability of the algorithm is enhanced. The frequency wave strategy balances the exploration and exploitation of the algorithm, and controls the prairie dog's position by controlling the search range for food and the effective escape range for natural enemies. In addition, the MPDO algorithm also adds chaotic tent mapping and lens opposition-based learning strategy. The tent chaotic mapping [31] is added in population initialization, making the initialization distribution of the prairie dogs population more uniform and providing the possibility of finding the optimal solution. At the same time, the lens opposition-based learning strategy [32] enhances the algorithm's global exploration ability.

Through the above strategies, the MPDO algorithm has better global exploration ability. During the experimental phase, 23 standard benchmark and IEEE CEC2014 functions were used to test the MPDO algorithm. Then experimental data, convergence curves, and Wilcoxon rank sum test were analyzed. Finally, to test the MPDO algorithm's practicality in practical engineering problems, this article selected six engineering application problems to test the optimization performance of the MPDO algorithm. These experimental results indicate that the MPDO algorithm has good optimization performance.

The major contributions of this article are as follows:

• A frequency wave strategy was proposed according to prairie dogs' habitual nature. Then, chaotic tent mapping and lens opposition-based learning strategies are added to enhance the global exploration ability of the algorithm;

• Apply the MPDO algorithm to 23 benchmark functions to test its performance;

• The optimization performance of the MPDO algorithm was tested in IEEE CEC2014;

• Eight different algorithms were compared in 23 benchmark functions and IEEE CEC2014 testing.

The rest framework of this article is as follows: The second part briefly introduces the PDO algorithm. The third part introduces the modified methods of the MPDO algorithm and the idea of proposing strategies. The fourth part applies the MPDO algorithm to 23 benchmark and IEEE CEC2014 test functions and analyzes the experimental results. The fifth part provides the experimental results of the MPDO algorithm in six engineering design problems. Finally, the sixth part summarizes the entire article.

2. PDO

The PDO algorithm is a metaheuristic algorithm that simulates the foraging activities of prairie dogs. Prairie dogs engage in social activities such as foraging, building caves, maintaining caves and guarding against predators daily. Therefore, based on the daily activities of prairie dogs, the PDO algorithm is divided into four time periods. Then, we divide exploration and exploitation based on a fixed mirror lifestyle.

2.1. Population initialization

The foraging activity of each prairie dog is represented by 1 × dim in the spatial dimension. In order to prevent prairie dogs from deviating from their trajectory when foraging, upper-bound *UB* and lower-bound *LB* are specified to limit the movement range of prairie dogs. The set of each prairie dog in different locations is a solution to a problem. Figure 1 shows the solution of *N* prairie dogs in the dim dimension.

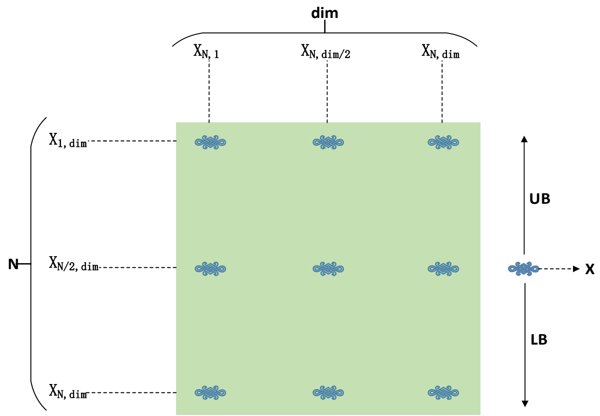


Figure 1. Population initialization.

2.2. Exploration stage

During the first time period, the position of prairie dogs in foraging activities was related to the food sources *ρ*, the current quality of food, and the location of randomly synthesized prairie dogs. *ρ* is a fixed food source alarm at 0.1 Khz. In the mathematical model, the quality of the current food is defined as the effectiveness of the evaluation currently obtained best solution $eCBest_{i,j}$. The position of the randomly synthesized prairie dog is defined as the random cumulative effect $CPD_{i,j}$. The calculation formula is as follows:

$$eCBest_{i,j} = GBest_{i,j} \times \Delta + \frac{PD_{i,j} \times mean(PD_i)}{GBest_{i,j} \times (UB_j - LB_j) + \Delta} \tag{1}$$

$$CPD_{i,j} = \frac{GBest_{i,j} - rPD_{i,j}}{GBest_{i,j} + \Delta} \tag{2}$$

where $GBest_{i,j}$ is the global optimal solution obtained so far, *Δ* is a very small number indicating the differences between prairie dog, and $rPD_{i,j}$ are the positions of the random solutions of prairie dog.

Therefore, the formula for updating the location of prairie dogs searching for food is as follows:

$$PD_{i+1,j+1} = GBest_{i,j} - eCBest_{i,j} \times \rho - CPD_{i,j} \times Levy(n) \tag{3}$$

In formula (3), Levy is a Levy distribution with discontinuous jumps.

After finding new food sources, prairie dogs excavate and build new caves around them. During this time period, the location of prairie dogs is related to their excavation intensity *DS* of the caves. The update formula for *DS* is as follows:

$$DS = 1.5 \times r \times (1 - \frac{t}{T})^{(2\frac{t}{T})} \tag{4}$$

r is transformed between -1 and 1 according to the parity of the current iteration number, t is the current iteration number, and T is the maximum iteration number.

The formula (5) shows an update in the position of prairie dogs during the second time period:

$$PD_{i+1,j+1} = GBest_{i,j} \times rPD \times DS \times Levy(n) \tag{5}$$

2.3. Exploitation stage

During the third time period, prairie dogs will refer to the quality of the current food source ε and the cumulative effect of all prairie dogs to randomly update their positions. In the mathematical model, the quality of the current food source ε is a small number designated as representing the quality of food source. The formula for updating the position of prairie dogs is as follows:

$$PD_{i+1,j+1} = GBest_{i,j} - eCBest_{i,j} \times \varepsilon - CPD_{i,j} \times rand \tag{6}$$

where, *rand* is a random number between 0 and 1.

During the foraging process of prairie dogs, predators often attack them. Therefore, the predator attack is defined as the predatory effect *PE*. The calculation formula for *PE* is as follows:

$$PE = 1.5 \times (1 - \frac{t}{T})^{(2\frac{t}{T})} \tag{7}$$

Update the position of prairie dogs during the fourth period by formula (8).

$$PD_{i+1,j+1} = GBest_{i,j} \times PE \times rand \tag{8}$$

2.4. Implementation of PDO algorithm

The original PDO algorithm simulated the behavior of prairie dogs in foraging, burrowing, and avoiding natural enemies, dividing the behavior of prairie dogs into four time periods. During these four time periods, prairie dogs, according to food sources alarm ρ, the cumulative effect of $CPD_{i,j}$ on all prairie dogs, the intensity of burrowing *DS*, the quality of food sources ε, and the predatory effect *PE* of the predator constantly updates position to find better food sources. Formula (9) summarizes the updated positions of prairie dogs at four time periods.

$$\begin{cases} PD_{i+1,j+1} = GBest_{i,j} - eCBest_{i,j} \times \rho - CPD_{i,j} \times Levy(n) & \forall t < \frac{T}{4} \\ PD_{i+1,j+1} = GBest_{i,j} \times rPD \times DS \times Levy(n) & \forall \frac{T}{4} \le t < \frac{T}{2} \\ PD_{i+1,j+1} = GBest_{i,j} - eCBest_{i,j} \times \varepsilon - CPD_{i,j} \times rand & \forall \frac{T}{2} \le t < 3\frac{T}{4} \\ PD_{i+1,j+1} = GBest_{i,j} \times PE \times rand & \forall 3\frac{T}{4} \le t < T \end{cases} \tag{9}$$

The pseudo-code of the PDO algorithm is shown in Algorithm 1.

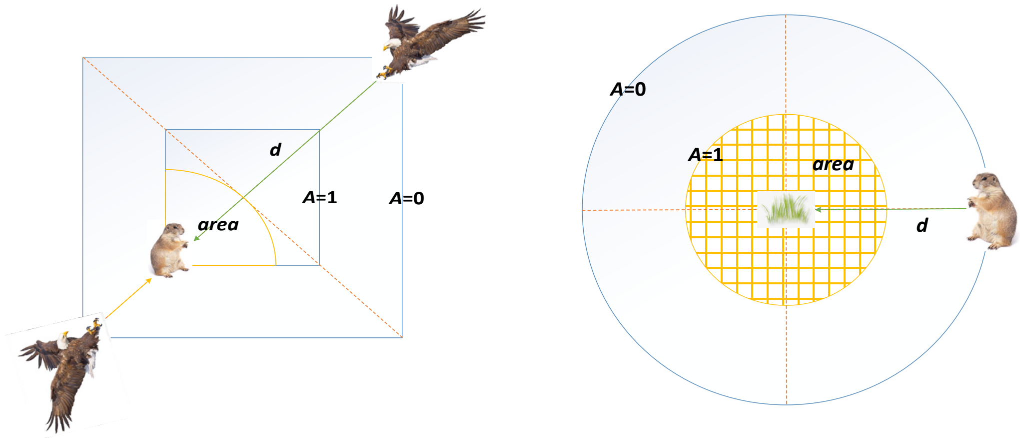
Algorithm 1. Prairie Dog Optimization Algorithm Pseudo-Code.

```
Population initialization
Initialization parameters
Calculate fitness value
while t ≤ T
        Update DS and PE using formulas (4) and (7)
        Update CPD_{i,j} using formula (2)
        If t < T / 4
              Update position using formula (3)
        Else if T / 4 ≤ t < T / 2
              Update position using formula (5)
        Else if T / 2 ≤ t < (3 × T) / 4
              Update position using formula (6)
        Else if (3 × T) / 4 ≤ t
              Update position using formula (8)
        End
    t = t + 1
End
```

3. MPDO

3.1. Frequency wave strategy

The prairie dog population has a perfect speech coordination system, where thought recognizes various foods and natural enemies and emits different frequencies of audio frequency fluctuation signals to provide feedback on the position of food and natural enemies. Therefore, a frequency wave strategy is proposed to improve the optimization performance of the algorithm. During the food search (Figure 2(b)), prairie dogs respond to the distance between food resource and prairie dogs with the audio signal's strength. When the audio signal is weak, prairie dogs tend to sound sources to search for better food; when the audio signal is strong, the prairie dog approaches the sound source around their current position. The distance between the predator and the prairie dog is reflected by the speed of the audio signal (Figure 2(a)). When predators appear in the prairie dog's view at a relatively long distance, they will emit a slower audio signal to remind the prairie dog population to stay away from their natural enemy; when the location of the predator poses a threat to the prairie dog, a faster audio signal will cause the prairie dog population to flee to the effective avoidance area quickly. Figure 2 is a schematic diagram of an audio signal warning simulation, briefly showing the movement of prairie dog within different signal ranges.



(a) Prairie dog evade natural enemies. (b) Prairie dog search for food resource.

Figure 2. Audio signal warning simulation schematic diagram.

The audio signal factor of the frequency wave strategy is defined as *A*, which changes randomly due to changes in the position of food or natural enemies. The distance between the prairie dog and food (natural enemies) is defined as *d*, and the sound source area (avoidance area) is defined as *area*. They calculation formulas are as follows:

$$A = 2 \times rand \tag{10}$$

$$d = Pos - PD \tag{11}$$

$$area = (abs(Pos^2 - PD_{i,j}^2))^{0.2} \tag{12}$$

In the above formulas, *rand* is the sound frequency fluctuation between 0 and 1 caused by random changes in the position of food or natural enemies. *Pos* is the location of food (natural enemies).

The specific update formula for frequency wave strategy is as follows:

$$\begin{cases} x_{new} = PD_{i,j} - A \times d \times Levy & A < 1 \\ x_{new} = PD_{i,j} + r \times A \times area & else \end{cases} \tag{13}$$

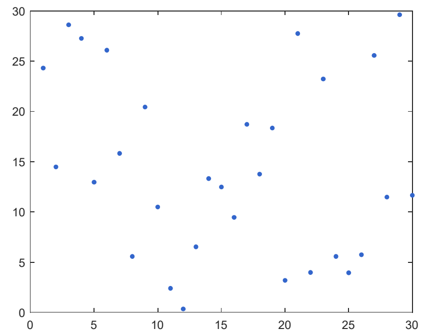
where, x_{new} represents the new location of the prairie dog, and *r* is a random number between –1 and 1.

3.2. Tent chaotic initialization

The original PDO algorithm uses the traditional population initialization method, which cannot effectively guarantee the randomness and diversity of the generated initial population position. However, the tent chaotic initialization population has ergodicity and orderliness, which can make the initial position distribution of the prairie dog population more uniform, thus expanding the scope of individual search space and maintaining population diversity. The mathematical model of tent chaos is as follows:

$$x_{i+1} = \begin{cases} \dfrac{x_i}{0.7} & x_i < 0.7 \\ \dfrac{10}{3}(1 - x_i) & x_i \geq 0.7 \end{cases} \tag{14}$$

In order to compare the differences between the original initialization and tent chaotic maps more intuitively, assuming the dimension is two-dimensional and the population size is 30, two initialization population distribution maps are shown in Figure 3.


(a) Original initialization

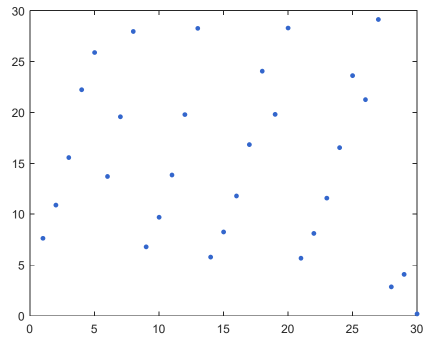

(b) Tent chaotic mapping

Figure 3. Distribution map of different initialized population.

From Figure 3, it can be observed that the population distribution generated by tent chaotic initialization is more orderly and uniform.

3.3. Lens opposition-based learning strategy

Traditional opposition-based learning is a strategy to expand the search range by generating the current solution in the opposite direction. The generated opposite solution is generally fixed, which is not conducive to the algorithm finding a better position. Based on the optical principle of convex lens imaging, taking one-dimensional space as an example, the coordinate axis [*lb*, *ub*] represents the search range, and the y-axis represents the convex lens. Assuming that there is an m individual with a height of *H*, the projection on the coordinate axis is *X*. The refraction of the lens *y* generates an image *m'* with a height of *H'*. The projection of *m'* on the coordinate axis is *X'*. The opposite individual *X'* generated by the convex lens imaging principle is shown in Figure 4.

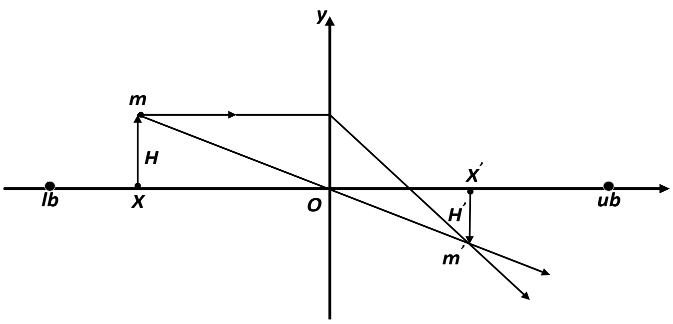


Figure 4. Schematic diagram of lens opposition-based learning.

The formula is derived from the principle of convex lens imaging:

$$\frac{(ub+lb)/2-X}{X^{'}-(ub+lb)/2}=\frac{H}{H^{'}} \tag{15}$$

Let H / H' = k, k be called the scaling factor, and substitute k into formula (15) to obtain the calculation formula for the opposition solution X':

$$X^{'}=\frac{ub+lb}{2}+\frac{ub+lb}{2k}-\frac{X}{k} \tag{16}$$

When $k = 1$, formula (16) can be abbreviated as the traditional opposition-based learning strategy, by adjusting the value of the scaling factor k, the position of generating the opposition solution in the D-dimensional space is random, the spatial search scope is further expanded, and the population diversity is increased.

3.4. Implementation of the MPDO algorithm

Foraging and burrowing are essential activities for the survival of prairie dogs, during which natural enemies will pursue them. According to the laws of natural survival, animals will gradually evolve while being hunted by natural enemies. In order to avoid the pursuit of natural enemies, prairie dogs have evolved a complex language system that allows them to emit different sound frequencies to respond when facing different natural enemies. According to the strength of the audio signal, prairie dogs will selectively stay away from or close to the food during the foraging process. When attacked by natural enemies, the faster audio signal can help prairie dogs escape to the effective area, while the slower audio signal can effectively remind the prairie dogs to stay away from natural enemies. Therefore, chaotic tent initialization evenly distributes the prairie dog population and increases population diversity. Then opposition-based learning will be carried out to expand the search space. The audio signal factors will enable prairie dogs to find better food resources or avoid pursuing natural enemies.

MPDO algorithm pseudo-code is shown in Algorithm 2.

Algorithm 2. The Modified Prairie Dog Optimization Algorithm Pseudo-Code

```
Using the formula (14) for population initialization
Calculate fitness value
While t ≤ T
        Implement lens opposition-base learning strategy through formula (16)
        Calculate the audio signal factor A using formula (10)
        Calculate the distance d using formula (11)
        If t < T / 4
            Update position using formula (3)
        Else if T / 4 ≤ t < T / 2
              Update position using formula (5)
        Else if T / 2 ≤ t < (3 × T) / 4
              Update position using formula (6)
        Else if (3 × T) / 4 ≤ t
              Update position using formula (8)
```

```
        End
        Calculate the area using formula (12)
        If A<1
            Use formula (13) to update the position of prairie dogs
        Else
            Use formula (13) to update the position of prairie dogs
        End
    t = t + 1
End
```

The flowchart of the MPDO algorithm is shown in Figure 5.

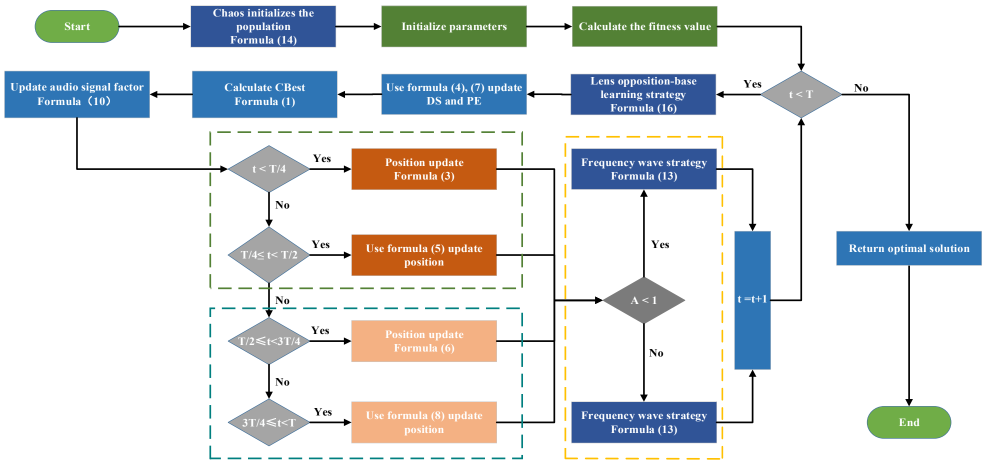


Figure 5. MPDO algorithm flowchart.

3.5. Time complexity analysis

Time complexity is an important indicator for evaluating an algorithm and directly reflects operational efficiency. Assuming the population size is N, the search space dimension is dim, the number of iterations is T, the time required for frequency wave strategy position update is f, the time required for lens opposition-based learning strategy position update is f, the evaluated time of the experimental function is t_2, the total running time of the algorithm is t. According to the calculation principle of time complexity, the following calculation formula (17) is given.

$$\begin{aligned} O(t) = O(population \quad initialization) + O(strategy \quad position \quad update) \\ + O(evaluate \quad experimental \quad function) \end{aligned} \tag{17}$$

During the algorithm operation, due to the short calculation time of the parameters, it can usually be ignored. The time complexity of calculating the parameters is not given in the above formula.

The time required for each stage of the PDO algorithm is defined as follows:

1) The time required for population initialization is $O\ (N \times dim \times T)$;
2) The time required to evaluate the experimental function is t_2.

Therefore, the time complexity of the PDO algorithm is expressed as the formula (18).

$$O(t) = O((N \times dim \times T) + t_2) \tag{18}$$

The time required for each stage of the MPDO algorithm is defined as:

1) The time required for tent chaos to initialize the population is O ($N \times dim \times T$);
2) The time required for the frequency wave strategy position update is O ($N \times dim \times T$) $\times f$);
3) The time required for the position update of the lens opposition-based learning strategy is O ($N \times dim \times T$) $\times f$);
4) The time required to evaluate the experimental function is t_2.

The time complexity of the MPDO algorithm is expressed as the formula (19).

$$O(t) = O((N \times \text{dim} \times T) \times (1 + 2f) + t_2) \tag{19}$$

Due to ($N \times dim \times T$) >> (1+2f), therefore the time complexity of the PDO algorithm and MPDO algorithm was replaced by formula (20). In summary, the time complexity of the MPDO algorithm is consistent with that of the PDO algorithm. The modifications made to this article's PDO algorithm do not increase time complexity.

$$O(t) = O((N \times \text{dim} \times T) \times C + t_2) \tag{20}$$

4. Experimental results and analysis

This experimental environment uses Windows 11 computer with a 64-bit operating system, an 11th Gen Inter (R) Core (TM) i7-11700 processor with a main frequency of 2.50 GHz, a memory of 16 GB, and a programming language implemented in MATLAB version R2021a.

Table 2. Parameter settings for comparison algorithms.

Algorithm	Parameters	Value
MPDO	A	[0,2]
	P	0.1
	Δ	0.005
PDO	P	0.1
	Δ	0.005
WOA	Coefficient Vector A	[–1,1]
	Coefficient Vector C	[0,2]
	Spiral parameters b	1
	Spiral parameters l	[–1,1]
SHO	U	0.05
	V	0.05
	L	0.05
ROA	C	0.1
SCA	a	2
SCSO	SM	2
	Roulette wheel selection	[0,360]
GWO	A	[–2,2]
	C	[0,2]

In order to verify the optimization performance of the MPDO algorithm, it was applied to 23 benchmark functions and IEEE CEC2014 benchmark functions for testing. The PDO algorithm [25], the WOA [33], the sea-horse optimizer (SHO) [34], the remora optimization algorithm (ROA) [35], the sine cosine algorithm (SCA) [36], the sand cat swarm optimization (SCSO) [37] and the grey wolf optimizer (GWO) [38] are selected to compare with MPDO algorithm. According to the best value (Min), mean value (Mean) and standard deviation (Std) obtained by each algorithm, the superiority of MPDO algorithm is analyzed. Then the advantages and disadvantages of MPDO algorithm in convergence graph are analyzed. Finally, the differences between MPDO algorithm and other algorithms are analyzed by Wilcoxon rank sum test. Table 2 provides the parameter settings for these eight algorithms.

4.1. 23 benchmark function testing experiments

The 23 benchmark test functions include seven Uni-modal benchmark functions, six multi-modal benchmark functions, and ten fixed-dimensional multi-modal benchmark functions. In this experiment, the population size is set to 30, the dimension is set to 30 and 500, and the maximum number of iterations is set to 500. The MPDO and the other seven algorithms were run 30 times each to obtain the best fitness value, average fitness value, and standard deviation.

4.1.1. Statistical analysis of 23 function experimental data and convergence curve

Tables 3–5 show the optimal values, mean values, and standard deviations of 8 different algorithms in 23 functions at 30 and 500 dimensions. From the data in Table 3, both MPDO and PDO obtained the theoretical optimal values of F1–F4, while ROA obtained the theoretical optimal values of F1, F3–F5. In F6, MPDO obtained a relatively stable theoretical optimal value. MPDO obtains the theoretical optimal value of F7. The relative optimal values of the F8 are WOA and ROA. Although MPDO did not obtain the theoretical optimal value, optimization results are significantly better than PDO optimization results. MPDO obtained stable relative optimal values in F9–F13, PDO, ROA, and SCSO obtained stable optimal values in F9–F11 WOA, and SHO obtained stable optimal values in F9 and F11. In addition, WOA also obtained optimal relative values in F10, and GWO obtained optimal relative values in F11. Due to the relatively simple fixed dimensional multi-modal benchmark functions F14–F23, eight algorithms in F14, F16–F19 obtained theoretical optimal values, while MPDO and SCSO obtained the optimal relative values in F15. Only SCA did not obtain theoretical optimal values for F20 and F23; SHO and SCA did not obtain optimal values for F22. F21 and F22 were similar, but PDO did not obtain theoretical optimal values in F21. Overall, the optimization performance of MPDO in uni-modal benchmark functions, multi-modal benchmark functions, and fixed dimensional multi-modal benchmark functions are superior to that of PDO and the other seven algorithms, indicating that MPDO using frequency wave strategy has better optimization performance.

To fully illustrate the optimization effect of MPDO, Figures 6–8 show the convergence ability of 8 algorithms in 23 functions in 30 and 500 dimensions. From the convergence curve, MPDO has good convergence ability in F1–F4 and quickly finds the function's optimal value. In F5, the MPDO algorithm is very similar to the optimal values found by other algorithms. In F10, both MPDO and ROA obtained good relative optimal values. In F6, F7, F12 and F13, the MPDO algorithm can effectively find better convergence values. In F9 and F11, the MPDO algorithm has good convergence speed and quickly finds the optimal value. Due to the relatively simple F14–F23 function, eight algorithms have good optimization results. In F16–F19, each algorithm quickly finds the optimal value. In F14, F15, F20–F23, the MPDO algorithm found the optimal value. Based on the above analysis, the

MPDO algorithm has better optimization ability than the PDO algorithm and has good results compared to the other seven algorithms.

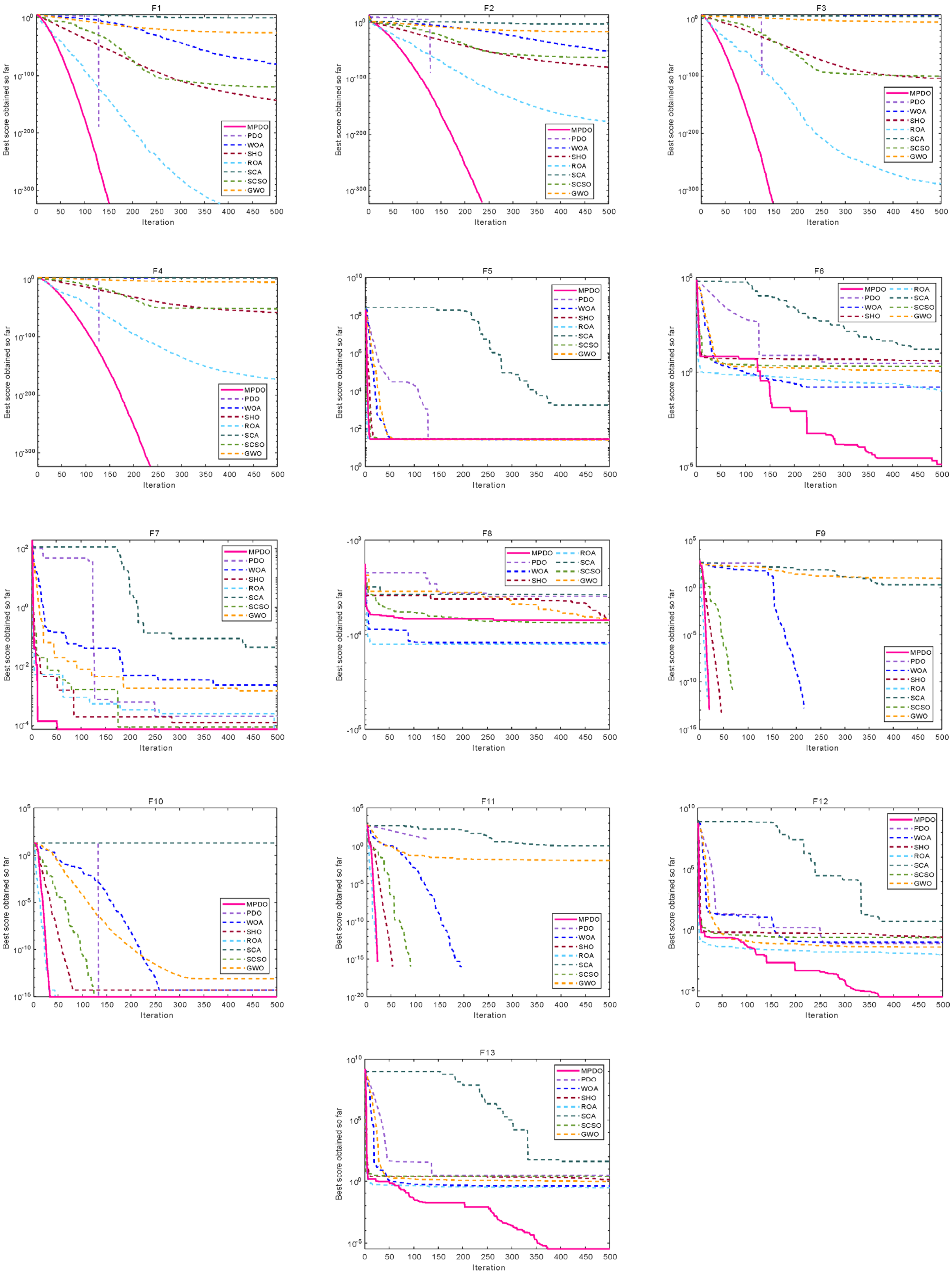


Figure 6. Convergence curves of various algorithms in the F1-F13 function (dim=30).

Table 3. Statistical results of F1–F13 standard Benchmark functions (dim = 30).

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
F1	Min	**0**	**0**	4.44×10^{-83}	4.68×10^{-147}	**0**	8.60×10^{-2}	4.00×10^{-132}	4.22×10^{-29}
	Mean	**0**	**0**	3.80×10^{-74}	1.45×10^{-140}	2.50×10^{-313}	13.5	6.54×10^{-111}	2.37×10^{-27}
	Std	**0**	**0**	1.50×10^{-73}	6.29×10^{-140}	**0**	28.5	3.43×10^{-110}	2.89×10^{-27}
F2	Min	**0**	**0**	1.93×10^{-57}	2.42×10^{-81}	6.01×10^{-184}	4.91×10^{-4}	1.29×10^{-65}	2.54×10^{-17}
	Mean	**0**	**0**	1.76×10^{-51}	8.83×10^{-78}	1.28×10^{-159}	2.45×10^{-2}	1.42×10^{-60}	1.21×10^{-16}
	Std	**0**	**0**	4.39×10^{-51}	3.50×10^{-77}	6.95×10^{-159}	6.35×10^{-2}	5.20×10^{-60}	1.06×10^{-16}
F3	Min	**0**	**0**	1.28×10^{4}	5.35×10^{-106}	**0**	1.56×10^{3}	3.47×10^{-110}	5.80×10^{-9}
	Mean	**0**	**0**	4.41×10^{4}	1.81×10^{-99}	2.31×10^{-284}	8.64×10^{3}	1.92×10^{-99}	3.15×10^{-5}
	Std	**0**	**0**	1.56×10^{4}	6.44×10^{-99}	**0**	5.60×10^{3}	8.14×10^{-99}	1.29×10^{-4}
F4	Min	**0**	**0**	1.44×10^{-1}	9.99×10^{-60}	4.17×10^{-181}	16.0	1.67×10^{-57}	7.48×10^{-8}
	Mean	**0**	**0**	45.6	2.36×10^{-56}	1.09×10^{-160}	3.68×10^{1}	3.44×10^{-51}	9.85×10^{-7}
	Std	**0**	**0**	26.7	1.00×10^{-55}	3.23×10^{-160}	13.0	1.21×10^{-50}	1.32×10^{-6}
F5	Min	28.3	2.97×10^{-1}	27.1	27.2	$\mathbf{2.66\times10^{-1}}$	92.0	26.2	26.2
	Mean	28.7	**16.6**	27.9	28.1	25.3	5.72×10^{4}	27.9	27.1
	Std	$\mathbf{8.54\times10^{-2}}$	13.6	4.13×10^{-1}	4.70×10^{-1}	6.66	1.29×10^{5}	9.37×10^{-1}	7.13×10^{-1}
F6	Min	$\mathbf{4.23\times10^{-7}}$	6.25×10^{-1}	8.06×10^{-2}	1.92	2.39×10^{-2}	4.32	7.23×10^{-1}	6.25×10^{-5}
	Mean	$\mathbf{1.48\times10^{-4}}$	2.98	3.16×10^{-1}	3.16	9.12×10^{-2}	21.6	1.94×10	7.96×10^{-1}
	Std	$\mathbf{1.72\times10^{-4}}$	1.58	2.09×10^{-1}	6.08×10^{-1}	5.79×10^{-2}	26.2	6.22×10^{-1}	4.19×10^{-1}
F7	Min	$\mathbf{1.54\times10^{-7}}$	2.05×10^{-6}	1.67×10^{-4}	7.27×10^{-6}	3.63×10^{-6}	9.11×10^{-3}	1.67×10^{-6}	5.56×10^{-4}
	Mean	5.42×10^{-5}	$\mathbf{9.78\times10^{-5}}$	5.77×10^{-3}	1.02×10^{-4}	1.92×10^{-4}	1.59×10^{-1}	1.67×10^{-4}	1.88×10^{-3}
	Std	5.96×10^{-5}	$\mathbf{9.75\times10^{-5}}$	5.64×10^{-3}	1.22×10^{-4}	2.50×10^{-4}	1.60×10^{-1}	2.09×10^{-4}	1.28×10^{-3}
F8	Min	-6.92×10^{3}	-4.36×10^{3}	$\mathbf{-1.26\times10^{4}}$	-7.30×10^{3}	$\mathbf{-1.26\times10^{4}}$	-4.27×10^{3}	-8.57×10^{3}	-7.30×10^{3}
	Mean	-5.88×10^{3}	-3.73×10^{3}	-1.05×10^{4}	-6.08×10^{3}	$\mathbf{-1.24\times10^{4}}$	-3.71×10^{3}	-6.83×10^{3}	-6.07×10^{3}
	Std	4.27×10^{2}	2.91×10^{2}	1.77×10^{3}	6.89×10^{2}	4.33×10^{2}	$\mathbf{2.39\times10^{2}}$	9.75×10^{2}	7.89×10^{2}
F9	Min	**0**	**0**	**0**	**0**	**0**	8.99×10^{-2}	**0**	5.68×10^{-14}
	Mean	**0**	**0**	**0**	**0**	**0**	41.7	**0**	2.41
	Std	**0**	**0**	**0**	**0**	**0**	31.6	**0**	3.62

Continued on next page

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
F10	Min	**8.88×10^{-16}**	**8.88×10^{-16}**	**8.88×10^{-16}**	4.44×10^{-15}	**8.88×10^{-16}**	6.29×10^{-2}	**8.88×10^{-16}**	7.19×10^{-14}
	Mean	**8.88×10^{-16}**	**8.88×10^{-16}**	4.32×10^{-15}	4.44×10^{-15}	**8.88×10^{-16}**	13.1	**8.88×10^{-16}**	1.04×10^{-13}
	Std	**0**	**0**	2.38×10^{-15}	**0**	**0**	9.35	**0**	1.84×10^{-14}
F11	Min	**0**	**0**	**0**	**0**	**0**	2.59×10^{-1}	**0**	**0**
	Mean	**0**	**0**	**0**	**0**	**0**	1.12	**0**	4.21×10^{-3}
	Std	**0**	**0**	**0**	**0**	**0**	8.24×10^{-1}	**0**	8.06×10^{-3}
F12	Min	**5.45×10^{-8}**	4.92×10^{-2}	5.16×10^{-3}	9.71×10^{-2}	2.07×10^{-3}	7.42×10^{-1}	3.15×10^{-2}	6.52×10^{-3}
	Mean	**1.67×10^{-6}**	5.24×10^{-1}	2.33×10^{-2}	2.87×10^{-1}	7.63×10^{-3}	3.96×10^{3}	1.10×10^{-1}	4.33×10^{-2}
	Std	**2.34×10^{-6}**	5.62×10^{-1}	1.85×10^{-2}	1.06×10^{-1}	4.37×10^{-3}	1.52×10^{4}	6.40×10^{-2}	2.19×10^{-2}
F13	Min	**3.55×10^{-7}**	1.95	1.03×10^{-1}	1.27	2.55×10^{-2}	3.07	1.45	1.67×10^{-1}
	Mean	**7.44×10^{-4}**	2.96	4.77×10^{-1}	2.10	2.46×10^{-1}	1.88×10^{4}	2.34	6.82×10^{-1}
	Std	**2.79×10^{-3}**	1.93×10^{-1}	1.94×10^{-1}	3.69×10^{-1}	1.44×10^{-1}	5.74×10^{4}	4.19×10^{-1}	2.55×10^{-1}

Table 4. Statistical results of F1–F13 standard Benchmark functions (dim = 500).

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
F1	Min	**0**	**0**	3.02×10^{-82}	1.75×10^{-113}	**0**	5.37×10^{4}	1.23×10^{-108}	9.73×10^{-4}
	Mean	**0**	**0**	1.12×10^{-71}	5.46×10^{-109}	9.82×10^{-314}	1.81×10^{5}	6.66×10^{-97}	1.39×10^{-3}
	Std	**0**	**0**	5.31×10^{-71}	2.31×10^{-108}	**0**	7.16×10^{4}	3.36×10^{-96}	3.33×10^{-4}
F2	Min	**0**	**0**	5.69×10^{-57}	1.26×10^{-61}	2.23×10^{-184}	23.7	5.97×10^{-56}	7.84×10^{-3}
	Mean	**0**	**0**	9.25×10^{-48}	2.33×10^{-59}	5.25×10^{-158}	1.17×10^{2}	3.17×10^{-51}	1.07×10^{-2}
	Std	**0**	**0**	2.99×10^{-47}	9.79×10^{-59}	2.86×10^{-157}	72.6	1.44×10^{-50}	1.73×10^{-3}
F3	Min	**0**	**0**	1.48×10^{7}	2.19×10^{-82}	3.83×10^{-299}	4.35×10^{6}	8.28×10^{-96}	1.39×10^{5}
	Mean	**0**	**0**	2.93×10^{7}	3.90×10^{-73}	8.99×10^{-267}	6.85×10^{6}	1.91×10^{-82}	3.17×10^{5}
	Std	**0**	**0**	1.12×10^{7}	1.71×10^{-72}	**0**	1.50×10^{6}	9.97×10^{-82}	7.70×10^{4}
F4	Min	**0**	**0**	52.9	1.05×10^{-48}	1.88×10^{-177}	98.6	3.15×10^{-53}	57.4
	Mean	**0**	**0**	80.5	2.83×10^{-46}	9.95×10^{-159}	99.0	1.30×10^{-45}	66.7
	Std	**0**	**0**	13.6	4.36×10^{-46}	4.22×10^{-158}	2.19×10^{-1}	4.80×10^{-45}	4.48

Continued on next page

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
F5	Min	**4.94×10^{2}**	4.99×10^{2}	4.95×10^{2}	4.98×10^{2}	**4.94×10^{2}**	7.89×10^{8}	4.98×10^{2}	4.98×10^{2}
	Mean	**4.94×10^{2}**	4.99×10^{2}	4.96×10^{2}	4.99×10^{2}	4.95×10^{2}	1.84×10^{9}	4.98×10^{2}	4.98×10^{2}
	Std	**5.05×10^{-3}**	6.02×10^{-3}	4.85×10^{-1}	1.50×10^{-1}	2.34×10^{-1}	4.91×10^{8}	1.62×10^{-1}	2.69×10^{-1}
F6	Min	**1.09×10^{-5}**	46.0	10.1	1.16×10^{2}	6.86×10^{-1}	8.36×10^{4}	93.2	87.4
	Mean	**2.87×10^{-3}**	1.02×10^{2}	30.9	1.17×10^{2}	14.9	1.75×10^{5}	1.05×10^{2}	91.2
	Std	**4.00×10^{-3}**	30.2	8.95	6.92×10^{-1}	6.69	6.55×10^{4}	4.36	2.16
F7	Min	4.17×10^{-7}	4.25×10^{-6}	7.73×10^{-5}	3.56×10^{-5}	**4.15×10^{-7}**	7.61×10^{3}	2.00×10^{-5}	3.06×10^{-2}
	Mean	**6.15×10^{-5}**	9.80×10^{-5}	3.62×10^{-3}	1.25×10^{-4}	2.11×10^{-4}	1.50×10^{4}	1.98×10^{-4}	4.46×10^{-2}
	Std	**5.85×10^{-5}**	8.39×10^{-5}	2.63×10^{-3}	6.97×10^{-5}	2.13×10^{-4}	4.28×10^{3}	1.91×10^{-4}	1.05×10^{-2}
F8	Min	**-8.89×10^{4}**	-2.50×10^{4}	-2.09×10^{5}	-2.58×10^{4}	-2.09×10^{5}	-1.75×10^{4}	-6.82×10^{4}	-7.11×10^{4}
	Mean	**-8.58×10^{4}**	-2.22×10^{4}	-1.67×10^{5}	-2.21×10^{4}	-2.03×10^{5}	-1.54×10^{4}	-6.01×10^{4}	-5.64×10^{4}
	Std	1.68×10^{3}	1.86×10^{3}	2.86×10^{4}	2.06×10^{3}	1.34×10^{4}	**1.00×10^{3}**	4.83×10^{3}	9.02×10^{3}
F9	Min	**0**	**0**	**0**	**0**	**0**	4.78×10^{2}	**0**	45.6
	Mean	**0**	**0**	**0**	**0**	**0**	1.32×10^{3}	**0**	77.8
	Std	**0**	**0**	**0**	**0**	**0**	6.93×10^{2}	**0**	29.1
F10	Min	**8.88×10^{-16}**	**8.88×10^{-16}**	**8.88×10^{-16}**	4.44×10^{-15}	**8.88×10^{-16}**	12.4	**8.88×10^{-16}**	1.46×10^{-3}
	Mean	**8.88×10^{-16}**	**8.88×10^{-16}**	4.80×10^{-15}	4.68×10^{-15}	**8.88×10^{-16}**	20.3	**8.88×10^{-16}**	1.89×10^{-3}
	Std	**0**	**0**	2.35×10^{-15}	9.01×10^{-16}	**0**	1.91	**0**	2.89×10^{-4}
F11	Min	**0**	**0**	**0**	**0**	**0**	4.20×10^{2}	**0**	1.05×10^{-4}
	Mean	**0**	**0**	**0**	**0**	**0**	1.84×10^{3}	**0**	1.86×10^{-2}
	Std	**0**	**0**	**0**	**0**	**0**	7.25×10^{2}	**0**	3.96×10^{-2}
F12	Min	**8.88×10^{-8}**	5.91×10^{-2}	2.60×10^{-2}	1.01	7.21×10^{-3}	3.18×10^{9}	6.31×10^{-1}	6.54×10^{-1}
	Mean	**3.03×10^{-6}**	7.24×10^{-1}	8.70×10^{-2}	1.05	3.80×10^{-2}	5.63×10^{9}	7.59×10^{-1}	7.36×10^{-1}
	Std	**4.45×10^{-6}**	4.67×10^{-1}	4.27×10^{-2}	1.77×10^{-2}	2.43×10^{-2}	1.18×10^{9}	6.88×10^{-2}	3.33×10^{-2}
F13	Min	**1.55×10^{-5}**	50.0	6.77	49.4	1.59	4.65×10^{9}	49.6	48.1
	Mean	**5.43×10^{-4}**	50.0	18.4	49.6	7.55	1.03×10^{10}	49.8	50.8
	Std	4.75×10^{-4}	**4.47×10^{-4}**	6.93	1.07×10^{-1}	3.19	2.07×10^{9}	8.74×10^{-2}	1.52

Table 5. Statistical results of F14–F23 standard Benchmark functions.

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
F14	Min	**9.98×10^{-1}**	**9.98×10^{-1}**	**9.98×10^{-1}**	**9.98×10^{-1}**	**9.98×10^{-1}**	**9.98×10^{-1}**	**9.98×10^{-1}**	**9.98×10^{-1}**
	Mean	3.65	4.72	3.16	6.20	4.39	**1.73**	3.81	4.32
	Std	3.28	3.62	**3.23**	5.07	4.60	9.69×10^{-1}	3.81	4.34
F15	Min	**3.07×10^{-4}**	5.65×10^{-4}	3.08×10^{-4}	3.08×10^{-4}	3.08×10^{-4}	4.11×10^{-4}	**3.07×10^{-4}**	3.08×10^{-4}
	Mean	6.17×10^{-4}	1.70×10^{-3}	7.75×10^{-4}	1.06×10^{-3}	4.77×10^{-4}	1.05×10^{-3}	**4.14×10^{-4}**	3.14×10^{-3}
	Std	2.09×10^{-4}	1.05×10^{-3}	5.27×10^{-4}	3.73×10^{-3}	**1.98×10^{-4}**	3.47×10^{-4}	2.61×10^{-4}	6.87×10^{-3}
F16	Min	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**
	Mean	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**	**-1.03**
	Std	**4.55×10^{-16}**	3.35×10^{-3}	3.38×10^{-10}	6.26×10^{-7}	2.37×10^{-6}	3.53×10^{-5}	1.04×10^{-9}	3.04×10^{-8}
F17	Min	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**
	Mean	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.99×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**
	Std	**3.24×10^{-16}**	1.97×10^{-4}	5.85×10^{-6}	3.18×10^{-5}	8.95×10^{-5}	1.07×10^{-3}	3.27×10^{-8}	8.47×10^{-7}
F18	Min	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**
	Mean	**3.00**	**3.00**	**3.00**	**5.70**	**3.00**	**3.00**	**3.00**	**3.00**
	Std	1.51×10^{-5}	**6.42×10^{-13}**	1.41×10^{-3}	14.8	5.14×10^{-4}	5.60×10^{-5}	7.41×10^{-6}	4.85×10^{-5}
F19	Min	**–3.86**	**–3.86**	**–3.86**	**–3.86**	**–3.86**	**–3.86**	**–3.86**	**–3.86**
	Mean	**–3.86**	**–3.86**	**–3.86**	**–3.86**	**–3.86**	**–3.85**	**–3.86**	**–3.86**
	Std	**2.39×10^{-15}**	6.69×10^{-3}	5.01×10^{-3}	3.05×10^{-3}	3.02×10^{-3}	1.15×10^{-2}	3.31×10^{-3}	4.66×10^{-3}
F20	Min	**–3.32**	**–3.32**	**–3.32**	**–3.32**	**–3.32**	–3.14	**–3.32**	**–3.32**
	Mean	–3.11	–3.01	–3.20	–2.95	–3.23	–2.86	–3.17	**–3.24**
	Std	3.86×10^{-1}	4.09×10^{-1}	2.19×10^{-1}	4.27×10^{-1}	9.94×10^{-2}	3.66×10^{-1}	1.80×10^{-1}	**8.85×10^{-2}**
F21	Min	**–10.2**	–10.1	**–10.2**	–10.1	**–10.2**	–4.91	**–10.2**	**–10.2**
	Mean	–6.42	–4.88	–7.27	–5.78	**–10.1**	–1.97	–5.19	–8.97
	Std	2.29	2.90	3.01	2.88	**3.35×10^{-2}**	1.46	1.97	2.17
F22	Min	**–10.4**	**–10.4**	**–10.4**	–10.3	**–10.4**	–6.55	**–10.4**	**–10.4**
	Mean	–8.63	–5.02	–7.42	–5.46	**–10.4**	–3.38	–6.48	**–10.4**
	Std	2.55	3.44	3.10	1.89	1.98×10^{-2}	2.03	2.68	**1.13×10^{-3}**

Continued on next page

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
F23	Min	**–10.5**	**–10.5**	**–10.5**	**–10.5**	**–10.5**	–8.77	**–10.5**	**–10.5**
	Mean	–7.29	–3.99	–6.92	–6.01	**–10.5**	–4.11	–6.22	**–10.5**
	Std	2.70	2.59	3.32	2.61	1.94×10^{-2}	2.07	2.82	**1.29×10^{-3}**

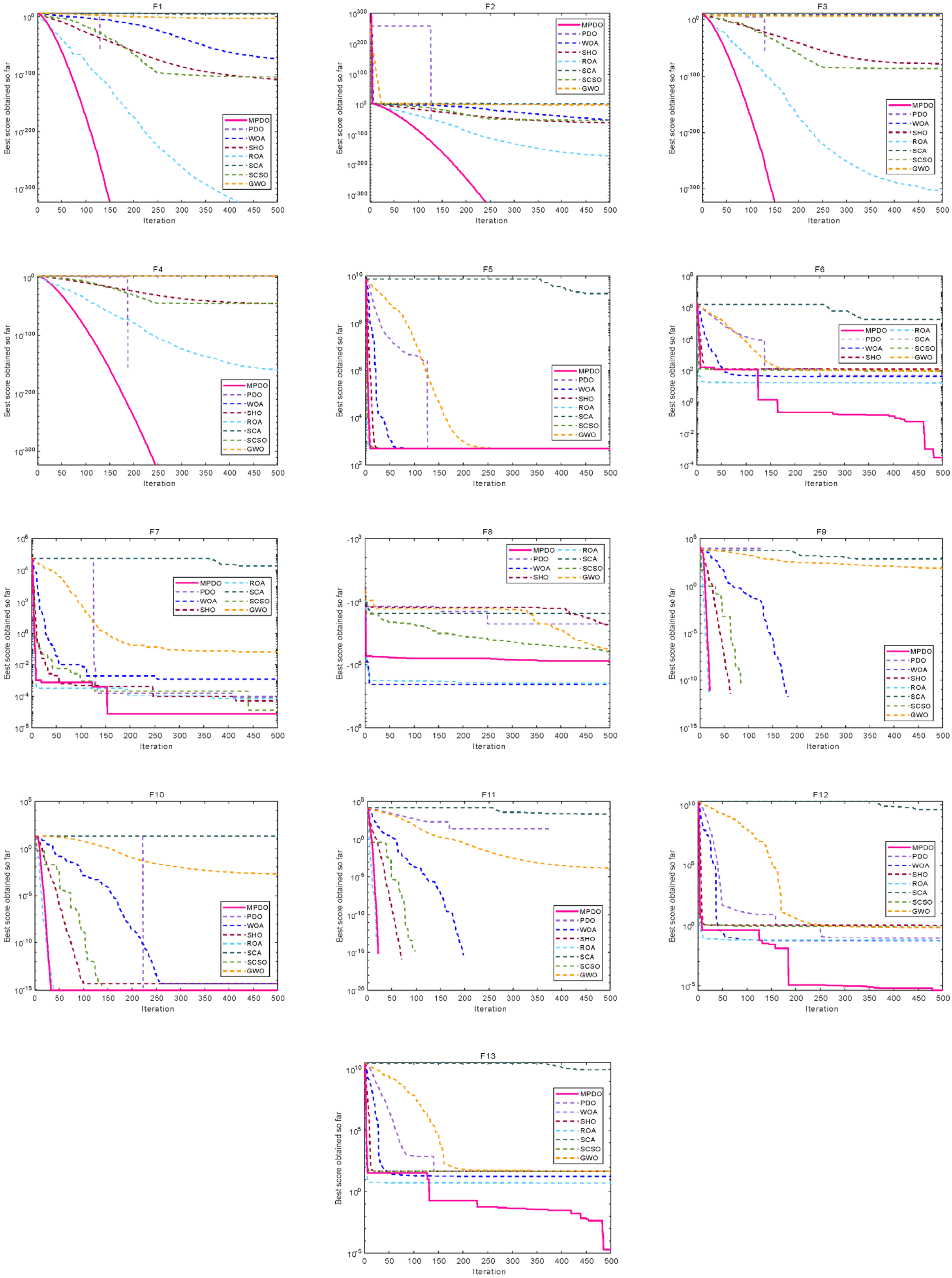


Figure 7. Convergence curves of various algorithms in the F1–F13 function (dim = 500).

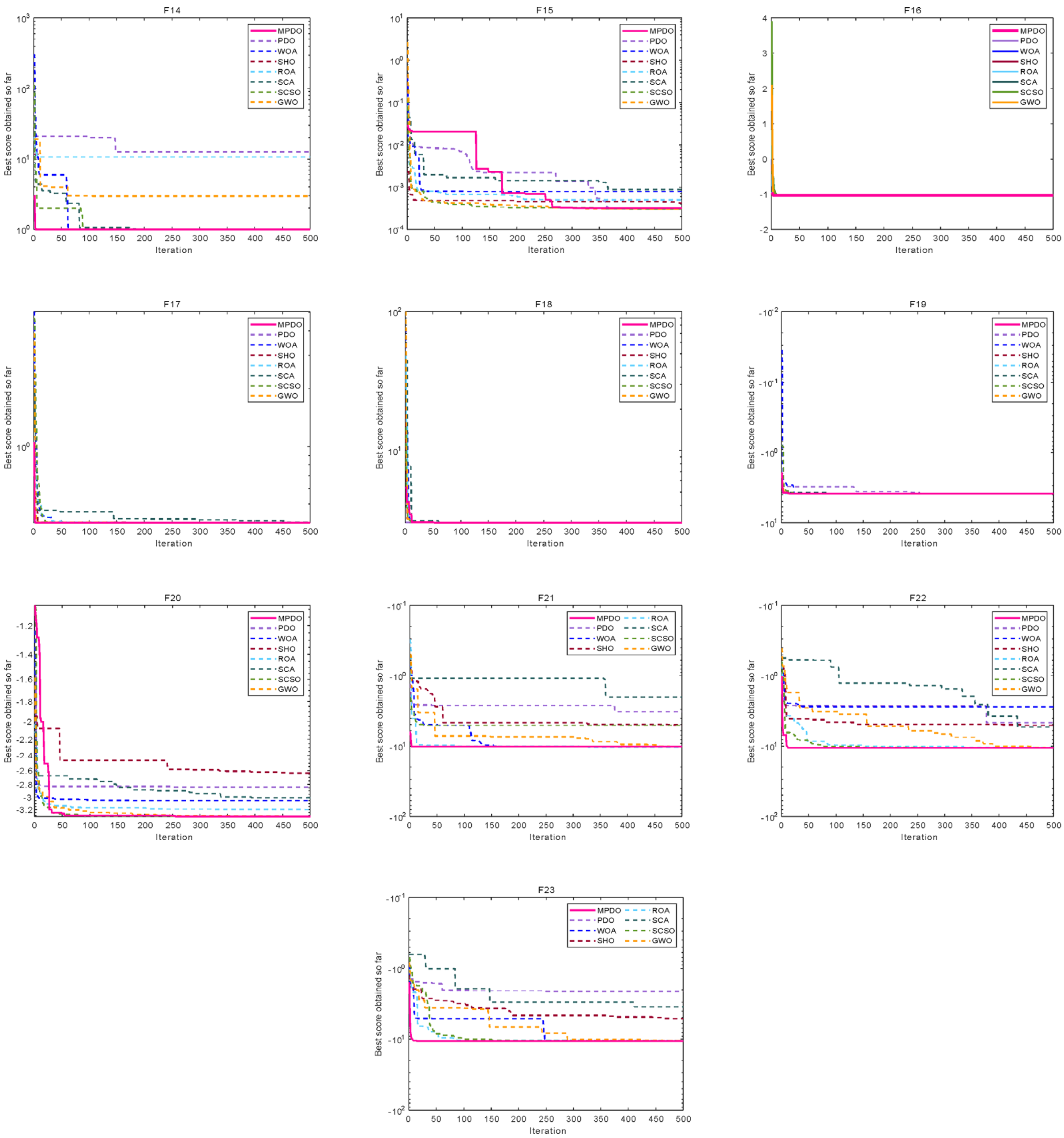


Figure 8. Convergence curves of various algorithms in F14–F23 functions.

4.1.2. Analysis of Wilcoxon rank sum test results

Wilcoxon rank sum test is a non-parametric test method that uses means to test whether there are differences between algorithms. After analysis of the 23 function data and convergence curve, we can only estimate that the MPDO algorithm has good optimization ability preliminarily. The Wilcoxon rank sum test compares the MPDO algorithm with seven different algorithms to test the differences between the MPDO algorithm and other algorithms. Table 6 shows that the results of the MPDO algorithm and the PDO algorithm in the F1–F4 function are 1, indicating that the values obtained by the two algorithms are consistent. The results of the ROA in F1 are greater than 5%, indicating that the difference between the ROA and the MPDO algorithm is small, and they have relatively close values. In the F9–F11 function, many results with 1 indicate that these algorithms are consistent with the MPDO algorithm and obtain the same value. The F14–F23 function is relatively simple, so many algorithms have smaller differences than the MPDO algorithm. In addition, the results of most of the data in the table are less than 5%, indicating significant differences between the MPDO algorithm and other algorithms in most cases.

Based on the analysis of the comprehensive data table, convergence curve, and Wilcoxon rank sum test results, the MPDO algorithm has good optimization performance among 23 benchmark test functions. Compared with the PDO algorithm, the optimization ability of the MPDO algorithm is significantly improved. Compared with other algorithms, the MPDO algorithm also has good advantages.

Table 6. Experimental results of the Wilcoxon rank-sum test on the 23 standard benchmark functions.

F	dim	MPDO VS PDO	MPDO VS WOA	MPDO VS SHO	MPDO VS ROA	MPDO VS SCA	MPDO VS SCSO	MPDO VS GWO
F1	30	**1.00**	1.73×10^{-6}	1.73×10^{-6}	$\mathbf{5.00\times10^{-1}}$	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	**1.00**	1.73×10^{-6}	1.73×10^{-6}	$\mathbf{2.50\times10^{-1}}$	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F2	30	**1.00**	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	**1.00**	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F3	30	**1.00**	1.73×10^{-6}	1.73×10^{-6}	2.56×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	**1.00**	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F4	30	**1.00**	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	**1.00**	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F5	30	4.72×10^{-2}	3.18×10^{-6}	1.80×10^{-5}	1.73×10^{-6}	1.73×10^{-6}	3.38×10^{-3}	1.92×10^{-6}
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F6	30	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	2.13×10^{-6}
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F7	30	$\mathbf{6.87\times10^{-2}}$	1.73×10^{-6}	$\mathbf{6.56\times10^{-2}}$	3.16×10^{-3}	1.73×10^{-6}	1.40×10^{-2}	1.73×10^{-6}
	500	$\mathbf{1.47\times10^{-1}}$	1.92×10^{-6}	1.83×10^{-3}	3.88×10^{-4}	1.73×10^{-6}	3.59×10^{-4}	1.73×10^{-6}
F8	30	1.73×10^{-6}	1.73×10^{-6}	$\mathbf{2.21\times10^{-1}}$	1.73×10^{-6}	1.73×10^{-6}	7.71×10^{-4}	$\mathbf{1.92\times10^{-1}}$
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F9	30	**1.00**	**1.00**	**1.00**	**1.00**	1.73×10^{-6}	**1.00**	1.68×10^{-6}
	500	**1.00**	**1.00**	**1.00**	**1.00**	1.73×10^{-6}	**1.00**	1.73×10^{-6}
F10	30	**1.00**	9.85×10^{-6}	4.32×10^{-8}	**1.00**	1.73×10^{-6}	**1.00**	1.67×10^{-6}
	500	**1.00**	5.06×10^{-6}	1.01×10^{-7}	**1.00**	1.73×10^{-6}	**1.00**	1.73×10^{-6}
F11	30	**1.00**	**1.00**	**1.00**	**1.00**	1.73×10^{-6}	**1.00**	1.56×10^{-2}
	500	**1.00**	**1.00**	**1.00**	**1.00**	1.73×10^{-6}	**1.00**	1.73×10^{-6}
F12	30	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F13	30	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}

Continued on next page

F	dim	MPDO VS PDO	MPDO VS WOA	MPDO VS SHO	MPDO VS ROA	MPDO VS SCA	MPDO VS SCSO	MPDO VS GWO
F14	30	**2.06×10^{-1}**	**4.78×10^{-1}**	2.85×10^{-2}	**8.88×10^{-1}**	5.67×10^{-3}	**8.77×10^{-1}**	**6.58×10^{-1}**
	500	2.07×10^{-2}	**5.86×10^{-1}**	3.16×10^{-3}	**3.71×10^{-1}**	**6.88×10^{-1}**	**6.73×10^{-1}**	**1.71×10^{-1}**
F15	30	4.29×10^{-6}	**5.17×10^{-1}**	9.63×10^{-4}	4.39×10^{-3}	6.34×10^{-6}	8.31×10^{-4}	**2.80×10^{-1}**
	500	1.24×10^{-5}	**9.59×10^{-1}**	3.88×10^{-4}	8.73×10^{-3}	2.61×10^{-4}	4.07×10^{-5}	**7.81×10^{-1}**
F16	30	8.86×10^{-5}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	2.93×10^{-4}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F17	30	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F18	30	1.73×10^{-6}	**1.53×10^{-1}**	8.92×10^{-5}	2.60×10^{-5}	**8.97×10^{-2}**	8.22×10^{-3}	4.07×10^{-2}
	500	1.73×10^{-6}	**4.41×10^{-1}**	1.64×10^{-5}	3.68×10^{-2}	1.04×10^{-2}	4.68×10^{-3}	**2.06×10^{-1}**
F19	30	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
	500	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
F20	30	**5.98×10^{-2}**	**5.17×10^{-1}**	1.40×10^{-2}	**2.62×10^{-1}**	1.20×10^{-3}	**7.50×10^{-1}**	**5.71×10^{-2}**
	500	**9.43×10^{-1}**	1.57×10^{-2}	**6.44×10^{-1}**	1.48×10^{-4}	**8.97×10^{-2}**	4.68×10^{-3}	1.25×10^{-4}
F21	30	1.74×10^{-4}	**8.61×10^{-1}**	3.33×10^{-2}	5.31×10^{-5}	1.73×10^{-6}	**3.93×10^{-1}**	1.48×10^{-4}
	500	1.29×10^{-3}	**7.50×10^{-1}**	1.85×10^{-2}	3.61×10^{-3}	2.88×10^{-6}	**2.37×10^{-1}**	**7.86×10^{-2}**
F22	30	9.32×10^{-6}	6.04×10^{-3}	1.64×10^{-5}	**6.44×10^{-1}**	5.22×10^{-6}	1.59×10^{-3}	**6.44×10^{-1}**
	500	1.36×10^{-5}	**2.80×10^{-1}**	1.24×10^{-5}	1.48×10^{-3}	6.98×10^{-6}	**1.71×10^{-1}**	3.61×10^{-3}
F23	30	6.89×10^{-5}	**1.85×10^{-1}**	4.39×10^{-3}	1.48×10^{-3}	2.22×10^{-4}	**3.60×10^{-1}**	1.48×10^{-3}
	500	4.68×10^{-3}	**2.80×10^{-1}**	1.13×10^{-5}	1.48×10^{-3}	5.31×10^{-5}	3.50×10^{-2}	1.75×10^{-2}

4.2. IEEE CEC2014 test function experiment

The IEEE CEC2014 test set has a total of 30 single objective test functions and is one of the most widely used. Therefore, we selected IEEE CEC2014 to verify the optimization performance of the MPDO algorithm, set the population size $N = 30$, and the maximum number of iterations $T = 500$.

Table 7 shows the optimal values, mean values, and standard deviations obtained by the MPDO algorithm in the IEEE CEC2014 test function. From the data in the Table 7, which eight algorithms have found the optimal solution in CEC12, CEC13, CEC14, CEC16 and CEC26. Only the PDO algorithm has not found the optimal value in CEC19. Although the MPDO algorithm did not obtain the optimal value in some CEC test functions, there is a small gap compared to the GWO algorithm that obtained the optimal solution. From the overall data, the optimization ability of the MPDO algorithm is stronger than the other seven algorithms and has good performance compared to the PDO algorithm.

Figures 9 and 10 shows the convergence curves of 8 algorithms in the IEEE CEC2014 test function. In the uni-modal function, it can be seen that the MPDO algorithm can find the optimal value better. In simple multi-modal functions, the MPDO algorithm did not find the optimal value in CEC6. In CEC8, CEC9 and CEC10, the MPDO and GWO algorithms are in a state of stagnation. In CEC16, the relative optimal obtained by MPDO is only weaker than that obtained by SHO. While in other functions, the MPDO algorithm has good convergence performance. Mixed and composite functions test the overall performance of algorithms. The MPDO algorithm showed good optimization performance in mixed functions. Although the MPDO algorithm showed weak convergence in the

CEC24 function, it showed good optimization performance in other composite functions. Overall, the MPDO algorithm has good optimization performance compared to other algorithms.

Through 30 independent runs, Table 8 obtains the Wilcoxon rank sum test data of the MPDO algorithm and the other seven algorithms in the IEEE CEC2014 test function. From Table 8, due to the simplicity of mixed and composite functions, CEC17-CEC28 has some results greater than 5%, but most are still less than 5%. The data from CEC8-CEC15 shows that only two results of each function are greater than 5%, while the rest are less than 5%. This indicates a significant difference between the MPDO and the other seven algorithms in these functions. In addition, in CEC5, only one data result is greater than 5%. In CEC6, two data results are greater than 5%. These data indicate that the MPDO algorithm differs significantly from other Wilcoxon rank sum test algorithms.

Table 7. Statistical results of IEEE CEC2014 test functions.

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
CEC1	Min	**1.17×10^{6}**	6.76×10^{8}	1.04×10^{8}	1.31×10^{8}	9.28×10^{7}	2.99×10^{8}	7.40×10^{7}	2.99×10^{7}
	Mean	**4.21×10^{6}**	1.21×10^{9}	2.33×10^{8}	3.96×10^{8}	3.02×10^{8}	5.01×10^{8}	2.05×10^{8}	1.03×10^{8}
	Std	**2.71×10^{6}**	3.41×10^{8}	8.41×10^{7}	1.40×10^{8}	9.83×10^{7}	1.40×10^{8}	9.43×10^{7}	6.28×10^{7}
CEC2	Min	**1.73×10^{3}**	5.39×10^{10}	2.40×10^{9}	1.76×10^{10}	9.59×10^{9}	1.90×10^{10}	1.34×10^{9}	3.02×10^{8}
	Mean	**1.55×10^{4}**	6.81×10^{10}	7.54×10^{9}	3.34×10^{10}	3.10×10^{10}	2.92×10^{10}	9.79×10^{9}	2.81×10^{9}
	Std	**9.73×10^{3}**	8.84×10^{9}	3.54×10^{9}	8.87×10^{9}	1.17×10^{10}	4.29×10^{9}	5.63×10^{9}	2.83×10^{9}
CEC3	Min	**5.85×10^{2}**	6.54×10^{4}	6.75×10^{4}	2.92×10^{4}	4.90×10^{4}	4.91×10^{4}	4.21×10^{4}	3.44×10^{4}
	Mean	**1.33×10^{4}**	1.25×10^{5}	1.43×10^{5}	4.91×10^{4}	6.80×10^{4}	7.71×10^{4}	5.50×10^{4}	5.26×10^{4}
	Std	9.34×10^{3}	4.54×10^{4}	7.63×10^{4}	1.03×10^{4}	**8.10×10^{3}**	1.64×10^{4}	8.85×10^{3}	1.21×10^{4}
CEC4	Min	**4.68×10^{2}**	4.56×10^{3}	7.72×10^{2}	1.07×10^{3}	9.64×10^{2}	1.55×10^{3}	6.53×10^{2}	5.57×10^{2}
	Mean	**5.34×10^{2}**	1.02×10^{4}	1.35×10^{3}	2.99×10^{3}	2.68×10^{3}	2.84×10^{3}	1.09×10^{3}	7.28×10^{2}
	Std	**3.76×10^{1}**	3.41×10^{3}	3.65×10^{2}	1.27×10^{3}	1.40×10^{3}	9.19×10^{2}	3.86×10^{2}	1.81×10^{2}
CEC5	Min	**5.20×10^{2}**	5.21×10^{2}	5.21×10^{2}	**5.20×10^{2}**	5.21×10^{2}	5.21×10^{2}	**5.20×10^{2}**	5.21×10^{2}
	Mean	**5.21×10^{2}**	**5.21×10^{2}**	**5.21×10^{2}**	**5.21×10^{2}**	**5.21×10^{2}**	**5.21×10^{2}**	**5.21×10^{2}**	**5.21×10^{2}**
	Std	4.59×10^{-1}	8.74×10^{-2}	9.63×10^{-2}	1.14×10^{-1}	**8.98×10^{-2}**	4.68×10^{-2}	1.43×10^{-1}	6.08×10^{-2}
CEC6	Min	6.30×10^{2}	6.39×10^{2}	6.32×10^{2}	6.26×10^{2}	6.27×10^{2}	6.32×10^{2}	6.26×10^{2}	**6.10×10^{2}**
	Mean	6.38×10^{2}	6.43×10^{2}	6.39×10^{2}	6.31×10^{2}	6.35×10^{2}	6.38×10^{2}	6.31×10^{2}	**6.17×10^{2}**
	Std	3.99	**2.04**	3.10	2.25	3.14	2.32	3.52	3.01
CEC7	Min	**7.00×10^{2}**	1.21×10^{3}	7.24×10^{2}	8.19×10^{2}	7.48×10^{2}	8.71×10^{2}	7.15×10^{2}	7.04×10^{2}
	Mean	**7.00×10^{2}**	1.38×10^{3}	7.49×10^{2}	9.96×10^{2}	9.13×10	9.50×10^{2}	7.92×10^{2}	7.27×10^{2}
	Std	**4.21×10^{-2}**	78.8	20.5	77.7	88.0	48.0	51.8	20.9
CEC8	Min	8.93×10^{2}	1.10×10^{3}	9.88×10^{2}	9.39×10^{2}	1.01×10^{3}	1.05×10^{3}	8.93×10^{2}	**8.54×10^{2}**
	Mean	9.85×10^{2}	1.19×10^{3}	1.04×10^{3}	9.78×10^{2}	1.04×10^{3}	1.09×10^{3}	9.96×10^{2}	**9.05×10^{2}**
	Std	56.7	47.4	46.3	**17.0**	20.7	24.1	37.4	21.9
CEC9	Min	1.03×10^{3}	1.18×10^{3}	1.12×10^{3}	1.08×10^{3}	1.09×10^{3}	1.16×10^{3}	1.05×10^{3}	**9.77×10^{2}**
	Mean	1.12×10^{3}	1.26×10^{3}	1.20×10^{3}	1.13×10^{3}	1.16×10^{3}	1.21×10^{3}	1.12×10^{3}	**1.03×10^{3}**

Continued on next page

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
	Std	37.3	67.8	68.8	25.5	28.7	**24.6**	30.2	42.3
CEC10	Min	3.29×10^3	7.42×10^3	4.64×10^3	3.19×10^3	4.60×10^3	6.88×10^3	3.93×10^3	$\mathbf{2.47\times10^3}$
	Mean	4.62×10^3	8.75×10^3	6.35×10^3	4.58×10^3	6.21×10^3	7.95×10^3	5.60×10^3	$\mathbf{3.55\times10^3}$
	Std	8.80×10^2	5.62×10^2	6.56×10^2	5.54×10^2	5.82×10^2	$\mathbf{4.76\times10^2}$	7.82×10^2	5.47×10^2
CEC11	Min	1.66×10^3	2.29×10^3	1.47×10^3	1.37×10^3	1.48×10^3	2.18×10^3	1.46×10^3	$\mathbf{1.16\times10^3}$
	Mean	2.37×10^3	2.84×10^3	2.31×10^3	1.81×10^3	2.20×10^3	2.61×10^3	2.04×10^3	$\mathbf{1.73\times10^3}$
	Std	3.92×10^2	2.76×10^2	3.31×10^2	2.27×10^2	4.02×10^2	$\mathbf{2.26\times10^2}$	2.97×10^2	2.78×10^2
CEC12	Min	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$
	Mean	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$	$\mathbf{1.20\times10^3}$
	Std	3.32×10^{-1}	5.65×10^{-1}	3.23×10^{-1}	2.16×10^{-1}	3.25×10^{-1}	2.96×10^{-1}	2.73×10^{-1}	$\mathbf{7.02\times10^{-1}}$
CEC13	Min	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$
	Mean	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$	$\mathbf{1.30\times10^3}$
	Std	2.47×10^{-1}	8.76×10^{-1}	2.01×10^{-1}	1.49×10^{-1}	1.83×10^{-1}	1.37×10^{-1}	1.88×10^{-1}	$\mathbf{7.31\times10^{-2}}$
CEC14	Min	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$
	Mean	$\mathbf{1.40\times10^3}$	$\mathbf{1.41\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$	$\mathbf{1.40\times10^3}$
	Std	2.09×10^{-1}	8.16	2.03×10^{-1}	2.04	4.14	$\mathbf{3.77\times10^{-1}}$	2.15×10^{-1}	1.72×10^{-1}
CEC15	Min	$\mathbf{1.50\times10^3}$	1.57×10^3	$\mathbf{1.50\times10^3}$	$\mathbf{1.50\times10^3}$	$\mathbf{1.50\times10^3}$	1.51×10^3	$\mathbf{1.50\times10^3}$	$\mathbf{1.50\times10^3}$
	Mean	1.51×10^3	4.01×10^3	1.51×10^3	1.51×10^3	1.60×10^3	1.51×10^3	$\mathbf{1.50\times10^3}$	$\mathbf{1.50\times10^3}$
	Std	5.58	3.61×10^3	6.54	8.19	4.03×10^2	2.76	1.91	**1.11**
CEC16	Min	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$
	Mean	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$	$\mathbf{1.60\times10^3}$
	Std	3.40×10^{-1}	2.22×10^{-1}	2.73×10^{-1}	3.01×10^{-1}	3.25×10^{-1}	1.97×10^{-1}	$\mathbf{4.11\times10^{-1}}$	3.85×10^{-1}
CEC17	Min	4.75×10^3	2.95×10^4	6.57×10^3	6.65×10^3	3.00×10^3	7.68×10^3	$\mathbf{2.37\times10^3}$	3.06×10^3
	Mean	1.00×10^5	5.36×10^5	2.43×10^5	1.96×10^5	6.06×10^4	$\mathbf{5.58\times10^4}$	7.22×10^4	1.44×10^5
	Std	1.05×10^5	1.99×10^5	4.88×10^5	1.51×10^5	1.01×10^5	$\mathbf{9.28\times10^4}$	1.51×10^5	2.22×10^5
CEC18	Min	$\mathbf{1.86\times10^3}$	7.73×10^3	2.47×10^3	6.11×10^3	2.16×10^3	5.21×10^3	2.01×10^3	1.99×10^3
	Mean	1.21×10^4	2.14×10^5	1.71×10^4	1.07×10^4	8.84×10^3	3.18×10^4	9.93×10^3	$\mathbf{1.05\times10^4}$
	Std	1.25×10^4	4.18×10^5	1.36×10^4	$\mathbf{2.89\times10^3}$	5.03×10^3	3.19×10^4	4.24×10^3	7.92×10^3

Continued on next page

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
CEC19	Min	**1.90×10^3**	1.91×10^3	**1.90×10^3**	**1.90×10^3**	**1.90×10^3**	**1.90×10^3**	**1.90×10^3**	**1.90×10^3**
	Mean	1.91×10^3	1.92×10^3	1.91×10^3	**1.90×10^3**	1.91×10^3	1.91×10^3	**1.90×10^3**	**1.90×10^3**
	Std	1.39	15.9	1.53	1.26	1.63	**8.96×10^{-1}**	1.22	9.38×10^{-1}
CEC20	Min	**2.06×10^3**	2.47×10^3	2.35×10^3	2.41×10^3	2.09×10^3	2.32×10^3	2.69×10^3	2.21×10^3
	Mean	8.85×10^3	4.67×10^5	1.23×10^4	**7.98×10^3**	1.17×10^4	9.49×10^3	7.89×10^3	9.14×10^3
	Std	7.55×10^3	1.44×10^6	7.00×10^3	**3.13×10^3**	4.68×10^3	7.35×10^3	3.35×10^3	5.25×10^3
CEC21	Min	**2.50×10^3**	3.08×10^4	6.10×10^3	3.62×10^3	3.23×10^3	3.81×10^3	3.66×10^3	2.83×10^3
	Mean	1.61×10^4	1.17×10^6	2.49×10^5	**1.05×10^4**	1.30×10^4	1.90×10^4	1.11×10^4	8.97×10^3
	Std	2.46×10^4	1.34×10^6	4.83×10^5	**3.55×10^3**	1.01×10^4	1.39×10^4	6.25×10^3	6.20×10^3
CEC22	Min	**2.22×10^3**	2.28×10^3	2.23×10^3	**2.22×10^3**	2.23×10^3	2.25×10^3	2.23×10^3	**2.22×10^3**
	Mean	2.46×10^3	2.48×10^3	2.30×10^3	2.29×10^3	2.29×10^3	**2.28×10^3**	2.31×10^3	2.30×10^3
	Std	1.13×10^2	1.09×10^2	78.2	60.3	71.5	**15.5**	63.5	59.9
CEC23	Min	**2.50×10^3**	**2.50×10^3**	**2.50×10^3**	**2.50×10^3**	**2.50×10^3**	2.64×10^3	**2.50×10^3**	2.63×10^3
	Mean	**2.50×10^3**	**2.50×10^3**	2.64×10^3	2.63×10^3	**2.50×10^3**	2.65×10^3	**2.50×10^3**	2.63×10^3
	Std	**0**	**0**	27.0	25.4	**0**	8.22	**0**	2.91
CEC24	Min	2.54×10^3	2.57×10^3	2.53×10^3	2.52×10^3	2.54×10^3	2.54×10^3	2.52×10^3	**2.51×10^3**
	Mean	2.59×10^3	2.60×10^3	2.58×10^3	2.57×10^3	2.59×10^3	2.56×10^3	2.59×10^3	**2.55×10^3**
	Std	15.8	7.76	25.1	30.7	18.4	**11.8**	22.4	36.3
CEC25	Min	**2.66×10^3**	2.69×10^3	2.70×10^3	2.68×10^3	2.68×10^3	2.68×10^3	2.67×10^3	2.67×10^3
	Mean	**2.69×10^3**	2.70×10^3	2.70×10^3	2.70×10^3	2.70×10^3	2.70×10^3	2.70×10^3	2.70×10^3
	Std	11.9	**1.81**	2.02	4.88	4.54	6.02	5.81	5.20
CEC26	Min	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**
	Mean	2.71×10^3	2.71×10^3	2.71×10^3	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**	**2.70×10^3**
	Std	25.2	17.6	25.3	1.04×10^{-1}	6.51×10^{-1}	1.59×10^{-1}	**9.05×10^{-2}**	18.2
CEC27	Min	2.90×10^3	2.90×10^3	2.71×10^3	2.71×10^3	2.71×10^3	2.72×10^3	2.71×10^3	**2.70×10^3**
	Mean	2.90×10^3	2.90×10^3	3.10×10^3	3.04×10^3	2.89×10^3	3.07×10^3	**2.88×10^3**	3.02×10^3
	Std	**0**	**0**	1.44×10^2	1.49×10^2	47.2	1.17×10^2	58.5	1.33×10^2
CEC28	Min	**3.00×10^3**	**3.00×10^3**	**3.00×10^3**	3.29×10^3	**3.00×10^3**	3.24×10^3	**3.00×10^3**	3.16×10^3

Continued on next page

F	Metric	MPDO	PDO	WOA	SHO	ROA	SCA	SCSO	GWO
	Mean	**3.00×10^{3}**	**3.00×10^{3}**	3.40×10^{3}	3.45×10^{3}	**3.00×10^{3}**	3.29×10^{3}	**3.00×10^{3}**	3.26×10^{3}
	Std	**0**	**0**	1.58×10^{2}	1.25×10^{2}	**0**	59.5	**0**	1.00×10^{2}
CEC29	Min	**3.10×10^{3}**	**3.10×10^{3}**	3.26×10^{3}	3.17×10^{3}	**3.10×10^{3}**	5.01×10^{3}	**3.10×10^{3}**	3.16×10^{3}
	Mean	3.43×10^{3}	**3.10×10^{3}**	3.57×10^{5}	7.19×10^{5}	1.77×10^{5}	2.41×10^{4}	7.13×10^{4}	5.15×10^{5}
	Std	3.48×10^{2}	**0**	8.62×10^{5}	1.52×10^{6}	5.26×10^{5}	1.95×10^{4}	3.58×10^{5}	1.15×10^{6}
CEC30	Min	**3.20×10^{3}**	**3.20×10^{3}**	4.14×10^{3}	4.54×10^{3}	3.56×10^{3}	4.27×10^{3}	3.66×10^{3}	3.51×10^{3}
	Mean	3.88×10^{3}	**3.20×10^{3}**	5.77×10^{3}	5.51×10^{3}	5.22×10^{3}	4.93×10^{3}	4.75×10^{3}	4.45×10^{3}
	Std	6.85×10^{2}	**0**	1.15×10^{3}	1.50×10^{3}	1.22×10^{3}	5.20×10^{2}	6.57×10^{2}	7.93×10^{2}

Table 8. Experimental results of the Wilcoxon rank−sum test on the IEEE CEC2014 test functions.

F	MPDO VS PDO	MPDO VS WOA	MPDO VS SHO	MPDO VS ROA	MPDO VS SCA	MPDO VS SCSO	MPDO VS GWO
CEC1	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
CEC2	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
CEC3	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
CEC4	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
CEC5	3.88×10^{-4}	1.29×10^{-3}	1.48×10^{-2}	4.53×10^{-4}	8.47×10^{-6}	$\mathbf{1.47\times10^{-1}}$	1.24×10^{-5}
CEC6	3.11×10^{-5}	$\mathbf{3.39\times10^{-1}}$	2.88×10^{-6}	5.32×10^{-3}	$\mathbf{8.45\times10^{-1}}$	1.80×10^{-5}	1.73×10^{-6}
CEC7	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}	1.73×10^{-6}
CEC8	1.73×10^{-6}	3.59×10^{-4}	$\mathbf{6.88\times10^{-1}}$	8.19×10^{-5}	2.35×10^{-6}	$\mathbf{4.17\times10^{-1}}$	6.98×10^{-6}
CEC9	1.73×10^{-6}	1.13×10^{-5}	$\mathbf{5.72\times10^{-1}}$	3.32×10^{-4}	1.92×10^{-6}	$\mathbf{6.29\times10^{-1}}$	2.60×10^{-6}
CEC10	1.73×10^{-6}	2.88×10^{-6}	$\mathbf{9.92\times10^{-1}}$	3.18×10^{-6}	1.73×10^{-6}	3.06×10^{-4}	4.45×10^{-5}
CEC11	2.16×10^{-5}	$\mathbf{8.45\times10^{-1}}$	1.36×10^{-5}	$\mathbf{1.85\times10^{-1}}$	8.73×10^{-3}	2.58×10^{-3}	3.88×10^{-6}
CEC12	2.88×10^{-6}	3.59×10^{-4}	$\mathbf{1.36\times10^{-1}}$	3.00×10^{-2}	2.88×10^{-6}	$\mathbf{4.05\times10^{-1}}$	3.00×10^{-2}
CEC13	1.73×10^{-6}	$\mathbf{2.29\times10^{-1}}$	$\mathbf{7.19\times10^{-1}}$	$\mathbf{6.58\times10^{-1}}$	1.29×10^{-3}	4.72×10^{-2}	6.34×10^{-6}
CEC14	1.73×10^{-6}	1.96×10^{-2}	8.94×10^{-4}	9.78×10^{-2}	1.92×10^{-6}	$\mathbf{3.82\times10^{-1}}$	$\mathbf{1.11\times10^{-1}}$
CEC15	1.73×10^{-6}	5.32×10^{-3}	$\mathbf{8.77\times10^{-1}}$	$\mathbf{5.45\times10^{-2}}$	2.58×10^{-3}	4.20×10^{-4}	4.29×10^{-6}
CEC16	5.29×10^{-4}	2.56×10^{-2}	1.02×10^{-5}	3.52×10^{-6}	1.57×10^{-2}	6.89×10^{-5}	1.73×10^{-6}
CEC17	2.88×10^{-6}	$\mathbf{2.62\times10^{-1}}$	8.73×10^{-3}	$\mathbf{8.59\times10^{-2}}$	3.16×10^{-2}	$\mathbf{6.27\times10^{-2}}$	$\mathbf{7.97\times10^{-1}}$
CEC18	3.38×10^{-3}	$\mathbf{1.41\times10^{-1}}$	$\mathbf{9.59\times10^{-1}}$	$\mathbf{5.17\times10^{-1}}$	1.96×10^{-3}	$\mathbf{6.14\times10^{-1}}$	$\mathbf{7.50\times10^{-1}}$
CEC19	1.73×10^{-6}	1.04×10^{-2}	1.89×10^{-4}	$\mathbf{7.19\times10^{-1}}$	1.59×10^{-3}	6.89×10^{-5}	2.88×10^{-6}
CEC20	3.41×10^{-5}	$\mathbf{8.22\times10^{-2}}$	$\mathbf{7.97\times10^{-1}}$	$\mathbf{1.06\times10^{-1}}$	$\mathbf{7.04\times10^{-1}}$	$\mathbf{5.72\times10^{-1}}$	$\mathbf{8.13\times10^{-1}}$
CEC21	2.60×10^{-6}	1.80×10^{-5}	$\mathbf{6.73\times10^{-1}}$	$\mathbf{7.34\times10^{-1}}$	$\mathbf{7.19\times10^{-2}}$	$\mathbf{8.29\times10^{-1}}$	$\mathbf{1.36\times10^{-1}}$
CEC22	$\mathbf{3.49\times10^{-1}}$	3.11×10^{-5}	1.24×10^{-5}	3.52×10^{-6}	4.29×10^{-6}	2.37×10^{-5}	1.13×10^{-5}
CEC23	**1.00**	2.56×10^{-6}	2.56×10^{-6}	**1.00**	1.73×10^{-6}	**1.00**	1.73×10^{-6}
CEC24	4.79×10^{-2}	$\mathbf{5.45\times10^{-2}}$	7.73×10^{-3}	$\mathbf{5.42\times10^{-1}}$	4.29×10^{-6}	$\mathbf{8.14\times10^{-2}}$	5.31×10^{-5}
CEC25	5.36×10^{-4}	2.35×10^{-5}	6.36×10^{-3}	2.23×10^{-3}	1.36×10^{-4}	1.48×10^{-3}	4.53×10^{-4}
CEC26	3.06×10^{-4}	$\mathbf{2.45\times10^{-1}}$	1.11×10^{-2}	$\mathbf{5.30\times10^{-1}}$	1.32×10^{-2}	6.64×10^{-4}	6.89×10^{-5}
CEC27	**1.00**	3.18×10^{-6}	7.69×10^{-6}	$\mathbf{5.00\times10^{-1}}$	3.18×10^{-6}	$\mathbf{2.50\times10^{-1}}$	1.96×10^{-3}
CEC28	**1.00**	2.56×10^{-6}	1.73×10^{-6}	**1.00**	1.73×10^{-6}	**1.00**	1.73×10^{-6}
CEC29	1.96×10^{-4}	1.97×10^{-5}	2.07×10^{-2}	2.60×10^{-6}	1.73×10^{-6}	3.15×10^{-5}	2.22×10^{-4}
CEC30	5.96×10^{-5}	2.13×10^{-6}	4.73×10^{-6}	5.79×10^{-5}	6.34×10^{-6}	4.86×10^{-5}	3.61×10^{-3}

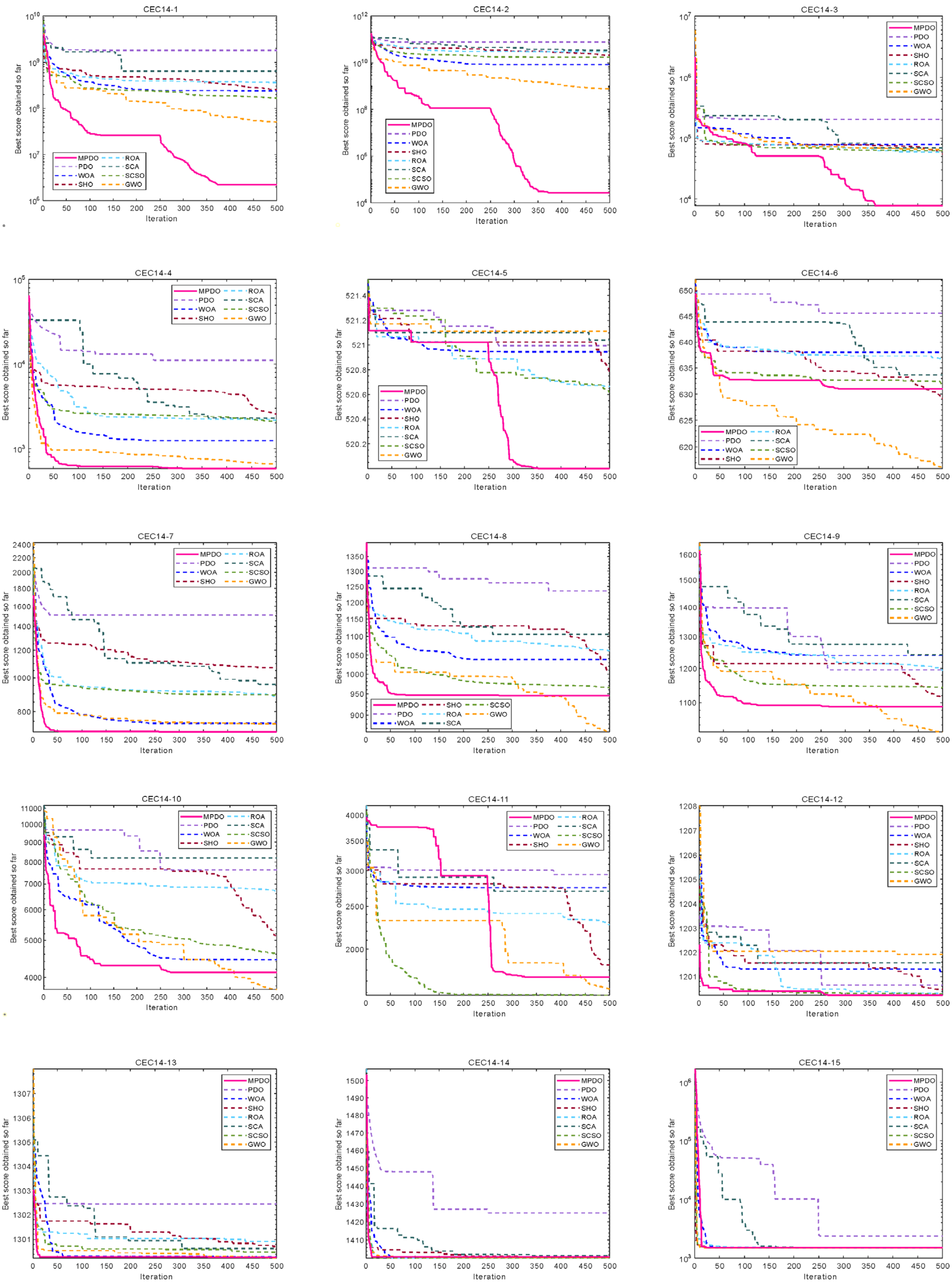


Figure 9. Convergence curves of various algorithms in the IEEE CEC2014 function (CEC 1–CEC 15).

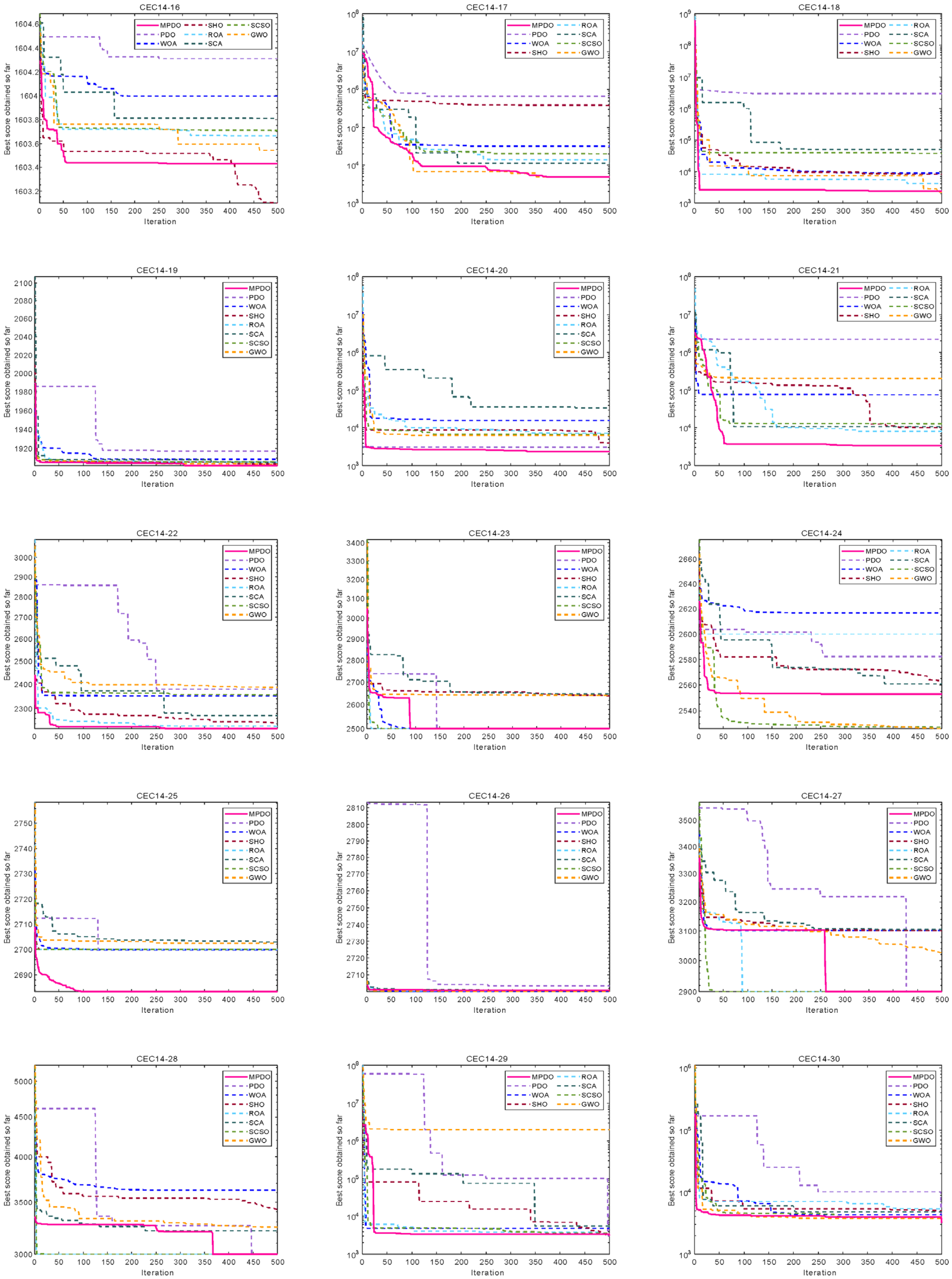


Figure 10. Convergence curves of various algorithms in the IEEE CEC2014 function (CEC 16–CEC 30).

5. Constrained engineering design problems

In Part 4, we tested the MPDO algorithm on 23 benchmark and IEEE CEC2014 functions. In order to test the practical effectiveness of the MPDO algorithm in engineering problems, in Part 5, we selected six engineering design problems: Car Crash-worthiness Design, Welded Beam Design, Speed Reducer Design, Cantilever Beam Design, Pressure Vessel Design, Multiple Disc Clutch Brake.

5.1. Car crash-worthiness design problem

The design of car crash-worthiness is a minimum value problem, which includes 11 variables and ten constraint conditions. Figure 11 shows the finite element model of the problem. The decision variables for this problem are the internal thickness of the B-pillar, the thickness of the B-pillar reinforcement, the thickness of the floor slab, the thickness of the crossbeam, the thickness of the door beam, the thickness of the door strip line reinforcement, the thickness of the longitudinal roof beam, the internal material of the B-pillar, the internal material of the floor slab, the height of the obstacle and the impact position of the obstacle. Abdominal load, upper viscosity standard, middle viscosity standard, low viscosity standard, upper rib deflection, middle rib deflection, lower rib deflection pubic symphysis force, B-pillar midpoint speed, and B-pillar front door speed are the constraints of this problem

Figure 11. Car crash-worthiness design model.

The mathematical formula for car crash-worthiness design is as follows:
Minimize:

$$f(\vec{x}) = \text{Weight}, \tag{21}$$

Subject to:

$$g_1(\vec{x}) = F_a \text{ (load in abdomen)} \le 1 \ kN, \tag{22}$$

$$g_2(\vec{x}) = V \times Cu \text{ (dummy upper chest)} \le 0.32 \ m/s, \tag{23}$$

$$g_3(\vec{x}) = V \times Cm \text{ (dummy middle chest)} \le 0.32 \ m/s, \tag{24}$$

$$g_4(\vec{x}) = V \times Cl \text{ (dummy lower chest)} \leq 0.32 \ m/s, \tag{25}$$

$$g_5(\vec{x}) = \Delta_{\text{ur}} \text{ (upper rib deflection)} \leq 32 \ mm, \tag{26}$$

$$g_6(\vec{x}) = \Delta_{\text{mr}} \text{ (middle rib deflection)} \leq 32 \ mm, \tag{27}$$

$$g_7(\vec{x}) = \Delta_{\text{lr}} \text{ (lower rib deflection)} \leq 32 \ mm, \tag{28}$$

$$g_8(\vec{x}) = F \text{ (Public force)}_p \leq 4 \ kN, \tag{29}$$

$$g_9(\vec{x}) = V_{\text{MBP}} \text{(Velocity of V}- \text{Pillar at middle point)} \leq 9.9 \text{ mm/ ms}, \tag{30}$$

$$g_{10}(\vec{x}) = V_{\text{FD}} \text{(Velocity of front door at V}- \text{Pillar)} \leq 15.7 \text{ mm/ ms}, \tag{31}$$

Variable range:

$$0.5 \leq x_1 - x_7 \leq 1.5, \ x_8, x_9 \in (0.192, 0.345), \ -30 \leq x_{10}, x_{11} \leq 30, \tag{32}$$

The experimental data for the car crash-worthiness design problem is shown in Table 9. The data in Table 9 shows that the optimal weight obtained by the MPDO algorithm is 23.19869131, which is the best solution for the optimal weights of six algorithms. This indicates that the MPDO algorithm can more efficiently solve the problem of car crash-worthiness design.

Table 9. Experimental results of the car crash-worthiness design problem.

Algorithm	**MPDO**	MALO[39]	MSROA[40]	SOA[41]	GTO[42]	MPA[43]
x1	**0.500000802**	0.5	0.5	0.50063	0.5	0.5
x2	**1.242709892**	1.2281	1.2284047	1.25921	1.2607	1.22823
x3	**0.5**	0.5	0.5	0.5	0.5	0.5
x4	**1.18070453**	1.2126	1.2125762	1.26308	1.1495	1.2049
x5	**0.500004599**	0.5	0.5	0.9377	0.6205	0.5
x6	**1.128124272**	1.308	0.9827072	1.11573	0.886	1.2393
x7	**0.500000896**	0.5	0.5	0.5	0.5	0.5
x8	**0.345**	0.3449	0.345	0.334889	0.34485	0.34498
x9	**0.193007238**	0.2804	0.345	0.252275	0.344608	0.192
x10	**3.036846819**	0.4242	0.2051698	4.3435	6.202292	0.44035
x11	**1.13771349**	4.6565	2.4627542	16.2208	7.3429	1.78504
Best Weight	**23.19869131**	23.2294	23.230900	24.42114	23.4084	23.19982

5.2. Welded beam design problem

The purpose of the welded beam design problem is to minimize the total cost of the welded beam, and the welded beam model is shown in Figure 12. The four decision variables for this problem are weld width *h*, connecting beam length *l*, beam height *t*, and connecting beam thickness *b*. In addition, there are seven constraint conditions.

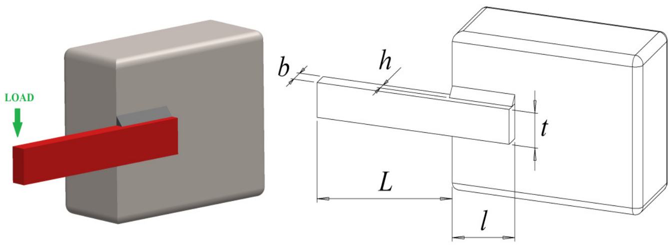


Figure 12. Welded beam model.

The mathematical formula for the design problem of welded beams is as follows:
Consider:

$$x = [x_1\ x_2\ x_3\ x_4] = [h\ l\ t\ b] \tag{33}$$

Objective function:

$$f(x) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) \tag{34}$$

Subject to:

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0 \tag{35}$$

$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0 \tag{36}$$

$$g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0 \tag{37}$$

$$g_4(\vec{x}) = x_1 - x_4 \le 0 \tag{38}$$

$$g_5(\vec{x}) = P - P_c(\vec{x}) \le 0 \tag{39}$$

$$g_6(\vec{x}) = 0.125 - x_1 \le 0 \tag{40}$$

$$g_7(\vec{x}) = 1.10471 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 0.5 \le 0 \tag{41}$$

Where:

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')}, \tau' = \frac{P}{\sqrt{2 x_1 x_2}}, \tau'' = \frac{MR}{J}, \tag{42}$$

$$M = P\left(L + \frac{x_2}{2}\right), R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \sigma(\vec{x}) = \frac{6PL}{x_4 x_3^2}, \tag{43}$$

$$J = 2\left\{\sqrt{2 x_1 x_2}\left[\frac{x_x^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \delta(\vec{x}) = \frac{6PL^3}{E x_4 x_3^2}, \tag{44}$$

$$P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{0}}}{L^2}, \left(1-\frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \left(1-\frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \tag{45}$$

$$P = 6000lb, L = 14\ in, \delta_{\max} = 0.25\ \text{in}, E = 30\times10^6\ psi, \tag{46}$$

$$\tau_{\max} = 13600\ psi, and\ \ \sigma_{\max} = 30000\ \ psi \tag{47}$$

Variable range:

$$0.1 \le x_i \le 2, i = 1,4; 0.1 \le x_i \le 10, i = 2.3 \tag{48}$$

The experimental results for the design issues of welded beams are shown in Table 10. The weight obtained by the MPDO algorithm is 1.708762277, the optimal solution compared to other algorithms. Under this weight, the thickness of the connecting beam *b* is 0.205418434, the height of the beam *t* is 9.099486427, the length of the connecting beam *l* is 3.315146754, and the weld width *h* is 0.201371958. From this, we have seen that the MPDO algorithm can effectively solve the problem of welded beam design.

Table 10. Experimental results of the welded beam design problem.

Algorithm	h	l	t	b	Best Weight
MPDO	**0.201371958**	**3.315146754**	**9.099486427**	**0.205418434**	**1.708762277**
TSA[44]	0.244157	6.223066	8.29555	0.244405	2.38241101
RO[45]	0.203687	3.528467	9.004233	0.207241	1.735344
IHS[46]	0.20573	3.47049	9.03662	0.2057	1.7248
CPSO[47]	0.202369	3.544214	9.04821	0.205723	1.73148
MFO[48]	0.2057	3.4703	9.0364	0.2057	1.72452
ROA[35]	0.200077	3.365754	9.011182	0.206893	1.706447

5.3. Speed reducer design problem

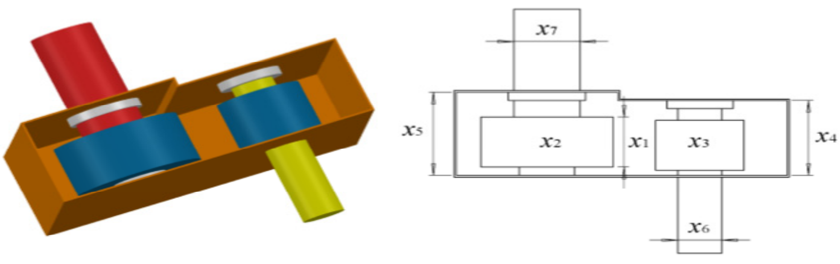


Figure 13. Speed reducer model.

The design model of the speed reducer is shown in Figure 13. The design of the speed reducer satisfies seven decision variables: the width of the tooth surface x_1, the number of teeth on the gear module x_2, the number of teeth on the pinion x_3, the length of the first shaft x_4 between bearings, the length of the second shaft x_5 between bearings, the diameter of the first shaft x_6 and the diameter of the second shaft x_7. It is a minimum value problem aimed at finding the minimum mass of the speed reducer, with four constraint conditions: the bending stress of the gear teeth, the covering stress, the lateral deflection of the shaft, and the stress inside the shaft.

The mathematical formula for the speed reducer design problem is as follows:

Consider:

$$x=[x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7] \tag{49}$$

Objective function:

$$\begin{aligned}f(\vec{x}) = 07854\times x_1\times x_2^2\times(3.3333\times x_3^2+14.9334\times x_3-\\43.0934)-1.508\times x_1\times(x_6^2+x_7^2)+7.4777\times x_6^3+x_7^3+\\0.7854\times x_4\times x_6^2+x_5\times x_7^2\end{aligned} \tag{50}$$

Subject to:

$$g_1(\vec{x})=\frac{27}{x_1\times x_2^2\times x_3}-1\le 0 \tag{51}$$

$$g_2(\vec{x})=\frac{397.5}{x_1\times x_2^2\times x_3^2}-1\le 0 \tag{52}$$

$$g_3(\vec{x})=\frac{1.93\times x_4^3}{x_2\times x_3\times x_6^4}-1\le 0 \tag{53}$$

$$g_4(\vec{x})=\frac{1.93\times x_5^3}{x_2\times x_3\times x_7^4}-1\le 0 \tag{54}$$

$$g_5(\vec{x})=\frac{1}{110\times x_6^3}\times\sqrt{(\frac{745\times x_4}{x_2\times x_3})^2+16.9\times 10^6}-1\le 0 \tag{55}$$

$$g_6(\vec{x})=\frac{1}{85\times x_7^3}\times\sqrt{(\frac{745\times x_5}{x_2\times x_3})^2+16.9\times 10^6}-1\le 0 \tag{56}$$

$$g_7(\vec{x})=\frac{x_2\times x_3}{40}-1\le 0 \tag{57}$$

$$g_8(\vec{x})=\frac{5\times x_2}{x_1}-1\le 0 \tag{58}$$

$$g_9(\vec{x})=\frac{x_1}{12\times x_2}-1\le 0 \tag{59}$$

$$g_{10}(\vec{x})=\frac{1.5\times x_6+1.9}{x_4}-1\le 0 \tag{60}$$

$$g_{11}(\vec{x})=\frac{1.1\times x_7+1.9}{x_5}-1\le 0 \tag{61}$$

Variable range:

$$2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3,$$
$$7.3 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5 \le x_7 \le 5.5 \tag{62}$$

Table 11 presents the test results of the reducer design problem. When x_1 is 3.497599089, x_2 is 0.7, x_3 is 17, x_4 is 7.3, x_5 is 7.8, x_6 is 3.350055813, and x_7 is 5.285531993, the optimal weight obtained by the MPDO algorithm is 2995.437365, which achieves better results compared to other algorithms. Therefore, MPDO is an effective algorithm for solving this problem.

Table 11. Experimental results of the speed reducer design problem.

Algorithm	**Optimal Values for Variables**							**Optimal Weight**
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
MPDO	**3.497599**	**0.7**	**17**	**7.3**	**7.8**	**3.35005581**	**5.28553199**	**2995.4374**
hHHO-SCA[49]	3.506119	0.7	17	7.3	7.99141	3.452569	5.286749	3029.8731
MSCSO[50]	3.497592	0.7	17	7.3	7.8	3.350043	5.285504	2995.438
AOA[51]	3.6	0.7	17	7.3	8.3	3.48321691	5.29818568	3089.0737
RSA[52]	3.50279	0.7	17	7.3	7.74715	3.35067	5.28675	2996.5157
MDA[53]	3.5	0.7	17	7.3	7.67039	3.54242	5.2481	3019.5833

5.4. Cantilever beam design problem

Cantilever beam design is a minimization problem aimed at reducing the weight of the cantilever beam. The decision variable for this problem includes five hollow block heights with constant thickness. Figure 14 shows the cantilever beam design model.

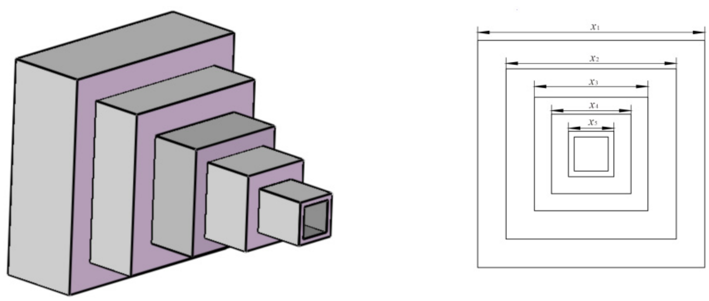


Figure 14. Cantilever beam model.

The mathematical formula for the design problem of cantilever beams is as follows:
Consider:

$$x = [x_1\ x_2\ x_3\ x_4\ x_5] \tag{63}$$

Objective function:

$$f(x) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5) \tag{64}$$

Subject to:

$$g(x)=\frac{61}{x_1^3}+\frac{37}{x_2^3}+\frac{19}{x_3^3}+\frac{7}{x_4^3}+\frac{1}{x_5^3}-1\leq 0 \tag{65}$$

Variable range:

$$0.01\leq x_i\leq 100(i=1,2,\cdots 5) \tag{66}$$

According to the cantilever beam model, the height of five hollow blocks with constant thickness continuously decreases. The results of the MPDO algorithm xi (I = 1, 2, ·····, 5) in Table 12 conform to the design of decreasing in sequence, and the optimal weight obtained is 1.340052195, which is an effective solution to this problem.

Table 12. Experimental results of the cantilever beam design problem.

Algorithm	Optimal Values for Variables					Optimum Weight
	x_1	x_2	x_3	x_4	x_5	
MPDO	**5.9909046**	**5.34666433**	**4.49228394**	**3.47344894**	**2.17189358**	**1.3400522**
ERHHO[54]	6.0509	5.2639	4.514	3.4605	2.1878	1.3402
BWO[55]	6.2094	6.2094	6.2094	6.2094	6.2094	1.9373625
OOA[56]	5.0000635	5.00006346	5.00006346	5.00006346	5.00006346	1.5600198
WOA[33]	5.1261	5.6188	5.0952	3.9329	2.3219	1.3787315
SCA[36]	5.1096	5.9911	5.015	3.7095	3.2744	1.4414387

5.5. Pressure vessel design problem

The design of pressure vessels minimizes the total cost of cylindrical pressure vessels to satisfy pressure requirements. There are four variables to address this issue: vascular wall thickness T_S, head wall thickness T_h, inner diameter R and body length L, and four constraint conditions. The specific mathematical model of the pressure vessel design problem is shown in Figure 15.

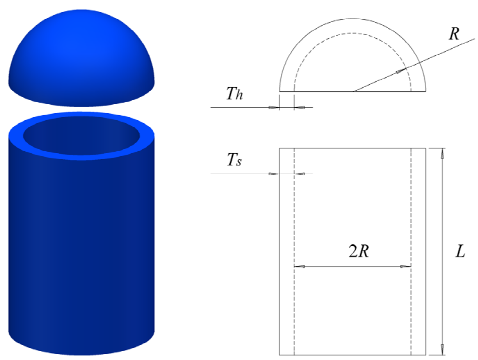


Figure 15. Pressure vessel model.

The mathematical formula for pressure vessel design problems is as follows:

Consider:

$$\vec{x} = [x_1\ x_2\ x_3\ x_4] = [T_s\ T_h\ R\ L] \tag{67}$$

Objective function:

$$f(\vec{x}) = 0.6224x_1x_2x_3 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{68}$$

Subject to:

$$g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0 \tag{69}$$

$$g_2(\vec{x}) = -x_3 + 0.00954x_3 \le 0 \tag{70}$$

$$g_3(\vec{x}) = -\pi x_3^2 x_4 + \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \tag{71}$$

$$g_4(\vec{x}) = -x_4 - 240 \le 0 \tag{72}$$

Variable range:

$$0 \le x_1 \le 99, 0 \le x_2 \le 99, 10 \le x_3 \le 200, 10 \le x_4 \le 200 \tag{73}$$

The experimental results of pressure vessel design issues are shown in Table 13. The MPDO algorithm yields a T_S of 0.747477958, T_h of 0.37238725, R of 40.56802084, and L of 196.5707208, resulting in a minimum cost of 5744.455052 for pressure vessel design. Five algorithms generated cost values greater than 6000 among the compared algorithms, while six generated cost values less than 6000.

Table 13. Experimental results of the pressure vessel design problem.

Algorithm	T_s	T_h	R	L	Best Cost
MPDO	**0.747477958**	**0.37238725**	**40.56802084**	**196.5707208**	**5744.455052**
EROA[57]	0.84343	0.400762	44.786	145.9578	5935.7301
HPSO[58]	0.8125	0.4375	42.0984	176.6366	6059.7143
AO[59]	1.054	0.182806	59.6219	39.805	5949.2258
MSROA[29]	0.773374321	0.374874166	41.83662957	180.1871401	5807.849903
MGTOA[60]	0.754364	0.366375	40.42809	198.5652	5752.402458
WOA[33]	0.8125	0.4375	42.09827	176.639	6059.741
GA[20]	0.8125	0.4375	42.0974	176.6541	6059.94634
CS[61]	0.8125	0.4375	42.09845	176.6366	6059.714335
SMA[62]	0.7931	0.3932	40.6711	196.2178	5994.1857
BA[63]	0.8125	0.4375	42.0984	176.6366	6059.7143
ES[64]	0.8125	0.4375	42.098087	176.640518	6059.7456

5.6. Multiple disc clutch brake problem

The main goal of a multi-disc clutch brake is to find the minimum mass of the multi-disc brake. This problem has five decision variables and eight constraint conditions. Five decision variables include inner radius r_i, outer radius r_o, brake disc thickness t, driving force F, and surface friction number Z. Figure 16 shows a specific model of a multi-disc clutch brake.

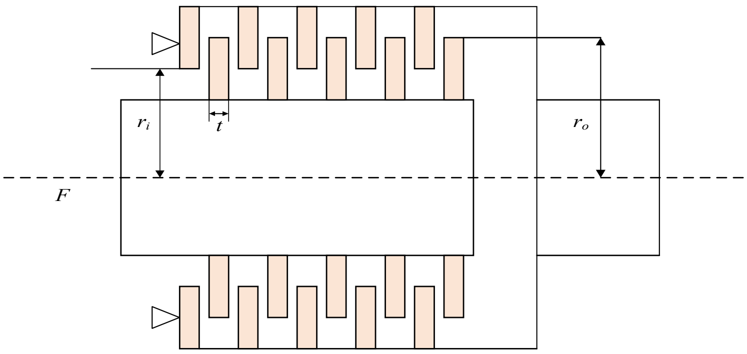


Figure 16. Multi-disc clutch brake model.

The mathematical formula for the multi-disc clutch brake problem is as follows:
Consider:

$$x = [x_1\ x_2\ x_3\ x_4\ x_5] = [r_i\ r_o\ t\ F\ Z] \tag{74}$$

Objective function:

$$f(x) = \text{II}(r_o^2 - r_i^2)t(Z+1)\rho \quad (\rho = 0.0000078) \tag{75}$$

Subject to:

$$g_1(x) = r_o - r_i - \Delta r \geq 0 \tag{76}$$

$$g_2(x) = l_{max} - (Z+1)(t+\delta) \geq 0 \tag{77}$$

$$g_3(x) = P_{max} - P_{rz} \geq 0 \tag{78}$$

$$g_4(x) = P_{max} v_{sr\ max} - P_{rz} \upsilon_{sr} \geq 0 \tag{79}$$

$$g_5(x) = v_{sr\ max} - \upsilon_{sr} \geq 0 \tag{80}$$

$$g_6(x) = T_{max} - T \geq 0 \tag{81}$$

$$g_7(x) = M_h - sM_s \geq 0 \tag{82}$$

$$g_8(x) = T \geq 0 \tag{83}$$

Variable range:

$$\begin{gathered} 60 \leq x_1 \leq 80, 90 \leq x_2 \leq 110, 1 \leq x_3 \leq 3, \\ 600 \leq x_4 \leq 1000, 2 \leq x_5 \leq 9 \end{gathered} \tag{84}$$

Other parameters:

$$M_h = \frac{2}{3}\mu F Z \frac{r_o^3 - r_i^2}{r_o^2 - r_i^3}, P_{rz} = \frac{F}{\text{II}(r_o^2 - r_i^2)}, \tag{85}$$

$$\upsilon_{rz} = \frac{2\Pi\left(r_o^3 - r_i^3\right)}{90\left(r_o^2 - r_i^2\right)}, T = \frac{I_z \Pi\, n}{30\left(M_h + M_f\right)} \tag{86}$$

$$\Delta r = 20mm, I_z = 55kgmm^2, P_{max} = 1MPa, F_{max} = 1000N, \tag{87}$$

$$T_{max} = 15s, \mu = 0.5, s = 1.5, M_s = 40Nm, M_f = 3Nm, \tag{88}$$

$$n = 250rpm, \upsilon_{sr\ max} = 10\,m/s, l_{max} = 30mm \tag{89}$$

Table 14 shows the test results of the multi-disc clutch brake problem. The MPDO algorithm obtains an inner radius r_i of 70, an outer radius r_o of 90, a brake disc thickness t of 1, a driving force F of 600, a surface friction number Z of 2, and an optimal weight of 0.235242458, which is the best solution compared to other algorithms.

Table 14. Experimental results of the multiple disc clutch brake problem.

Algorithm	Optimal Values for Variables					Optimum Weight
	x_1	x_2	x_3	x_4	x_5	
MPDO	**70**	**90**	**1**	**600**	**2**	**0.235242458**
WCA[65]	70	90	1	910	3	0.313656
CMVO[66]	70	90	1	910	3	0.313656
SCA[36]	69.516	90	1	1000	2	0.24019
MFO[48]	70	90	1	910	3	0.313656
RSA[52]	70.0347	90.0349	1	801.7285	2.974	0.31176
OOA[56]	60	90	1	600	2	0.330809706

6. Conclusions and future work

We propose a frequency wave strategy based on prairie dogs' special sound transmission mode. The position of prairie dogs changed by simulating different signals emitted when encountering different food sources and natural enemies. In order to balance the exploration and exploitation of the algorithm, the strong and weak audio signal received by prairie dogs in the foraging stage were used to expand or narrow the scope of searching for food, and the fast and slow audio signal received in the avoiding natural enemies stage were used to avoid or stay from nature enemies. This enables the algorithm to effectively find better optimization value in the later evaluation stage, enhancing the optimization ability of the algorithm. In order to enhance the global exploration ability of the algorithm, a chaotic tent map and lens opposition-based learning strategy are added to the evaluation process of the algorithm.

In order to verify the optimization performance of the MPDO algorithm, 23 benchmark test functions and IEEE CEC2014 test functions were used to evaluate the MPDO algorithm. Experimental data and convergence curves were analyzed by comparing them with seven algorithms. The final results showed that the MPDO algorithm has good optimization performance. In order to verify the practicality of the MPDO algorithm in engineering application problems, six constrained engineering design problems were tested at the end of the article. The comparison results with other algorithms proved that the MPDO algorithm is an effective strategy for solving practical application problems. Structural health monitoring (SHM) has been aiming at improving the damage detection capability of

SHM systems, reducing the load of large engineering structure, such as bridges, and improving structures' operation and service life. Therefore, in future work, we hope the MPDO algorithm is applied to the SHM field and achieves good results. In addition, we also hope to apply MPDO to the clustering problem, image segmentation and processing problem, and feature selection problem of machine learning.

Use of AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The authors would like to thank the support of Digital Fujian Research Institute for Industrial Energy Big Data, Fujian Province University Key Lab for Industry Big Data Analysis and Application, Fujian Key Lab of Agriculture IOT Application, IOT Application Engineering Research Center of Fujian Province Colleges and Universities, Sanming City 5G Innovation Laboratory, Fujian University students innovation and entrepreneurship training program (S202311311060), Ministry of Education supply and demand docking employment education project (20230104862), and also the anonymous reviewers and the editor for their careful reviews and constructive suggestions to help us improve the quality of this paper.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: A gravitational search algorithm, *Inf. Sci.*, **179** (2009), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004
2. H. Su, D. Zhao, A. A. Heidari, L. Liu, X. Zhang, M. Mafarja, et al., RIME: A physics-based optimization, *Neurocomputing*, **532** (2023), 183–214. https://doi.org/10.1016/j.neucom.2023.02.010
3. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, *Science*, **220** (1983), 671–680. https://www.science.org/doi/10.1126/science.220.4598.671
4. A. Hatamlou, Black hole: A new heuristic optimization approach for data clustering, *Inf. Sci.*, **222** (2013), 175–184. https://doi.org/10.1016/j.ins.2012.08.023
5. M. Abdel-Basset, R. Mohamed, S. A. A. Azeem, M. Jameel, M. Abouhawwash, Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion, *Knowl. Based Syst.*, **268** (2023), 110454. https://doi.org/10.1016/j.knosys.2023.110454
6. R. V. Rao, V. J. Savsani, D. P. Vakharia, Teaching-learning-based optimization: An optimization method for continuous nonlinear large scale problems, *Inf. Sci.*, **183** (2012), 1–15. https://doi.org/10.1016/j.ins.2011.08.006

7. I. Matoušová, P. Trojovský, M. Dehghani, E. Trojovská, J. Kostra, Mother optimization algorithm: A new human-based metaheuristic approach for solving engineering optimization, *Sci. Rep.*, **13** (2023), 10312. https://doi.org/10.1038/s41598-023-37537-8
8. Z. W. Geem, J. H. Kim, G. V. Loganathan, A new heuristic optimization algorithm: Harmony search, *Simulation*, **2** (2001), 60–68. https://doi.org/10.1177/003754970107600201
9. Y. Zhang, Z. Jin, Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems, *Expert Syst. Appl.*, **148** (2020), 113246. https://doi.org/10.1016/j.eswa.2020.113246
10. S. Cheng, Q. Qin, J. Chen, Y. Shi, Brain storm optimization algorithm, *Artif. Intell. Rev.*, **46** (2016), 445–458. https://doi.org/10.1007/s10462-016-9471-0
11. J. Kennedy, R. Eberhart, Particle swarm optimization, in *Proceedings of ICNN'95 - International Conference on Neural Networks*, **4** (1995), 1942–1948. https://doi.org/10.1109/ICNN.1995.488968
12. G. G. Wang, S. Deb, Z. Cui, Monarch butterfly optimization, *Neural Comput. Appl.*, **31** (2019), 1995–2014. https://doi.org/10.1007/s00521-015-1923-y
13. A. Forestiero, C. Mastroianni, G. Spezzano, QoS-based dissemination of content in grids, *Future Gener. Comput. Syst.*, **24.3** (2008), 235–244. https://doi.org/10.1016/j.future.2007.05.003
14. A. Forestiero, C. Mastroianni, G. Spezzano, Reorganization and discovery of grid information with epidemic tuning, *Future Gener. Comput. Syst.*, **24.8** (2008), 788–797. https://doi.org/10.1016/j.future.2008.04.001
15. A. Forestiero, Bio-inspired algorithm for outliers detection, *Multimed. Tools Appl.*, **76** (2017), 25659–25677. https://doi.org/10.1007/s11042-017-4443-1
16. J. Tu, H. Chen, M. Wang, A. H. Gandomi, The colony predation algorithm, *J. Bionic Eng.*, **18** (2021), 674–710. https://doi.org/10.1007/s42235-021-0050-y
17. M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, *IEEE Comput. Intell. Mag.*, **1** (2006), 28–39. https://doi.org/10.1109/MCI.2006.329691
18. H. Jia, H. Rao, C. Wen, S. Mirjalili, Crayfish optimization algorithm, *Artif. Intell. Rev.*, **2023** (2023), 1–61. https://doi.org/10.1007/s10462-023-10567-4
19. P. Trojovský, D. Mohammad, H. Pavel, Siberian tiger optimization: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems, *IEEE Access*, **10** (2022), 132396–132431. https://doi.org/10.1109/ACCESS.2022.3229964
20. J. H. Holland, Genetic algorithms, *Sci. Am.*, **267** (1992), 66–73. https://www.jstor.org/stable/24939139
21. N. Sinha, R. Chakrabarti, P. K. Chattopadhyay, Evolutionary programming techniques for economic load dispatch, *IEEE Trans. Evol. Comput.*, **7** (2003), 83–94.
22. R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, *J. Global. Optim.*, **11** (1997), 341–359. https://doi.org/10.1023/A:1008202821328
23. M. Jaderyan, K. Hassan, Virulence optimization algorithm, *Appl. Soft Comput.*, **43** (2016), 596–618. https://doi.org/10.1016/j.asoc.2016.02.038
24. D. Simon, Biogeography-based optimization, *IEEE Trans. Evol. Comput.*, **12** (2008), 702–713. https://doi.org/10.1109/TEVC.2008.919004

25. A. E. Ezugwu, J. O. Agushaka, L. Abualigah, S. Mirjalili, A. H. Gandomi, Prairie dog optimization algorithm, *Neural Comput. Appl.*, **34** (2022), 20017–20065. https://doi.org/10.1007/s00521-022-07530-9
26. D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, *IEEE Trans. Evol. Comput.*, **1** (1997), 67–82. https://doi.org/10.1109/4235.585893
27. J. Liu, S. Zhang, Z. Druzhinin, Performance prediction of the PEMFCs based on gate recurrent unit network optimized by improved version of prairie dog optimization algorithm, *Int. J. Hydrogen Energy*, **2023** (2023). https://doi.org/10.1016/j.ijhydene.2023.03.349
28. L. Ngoc-Nguyen, S. Khatir, H. Q. Nguyen, T. Bui-Tien, M. A. Wahab, A practical review of prairie dog optimization algorithm in solving damage identification problems in engineering structures, in *Proceedings of the International Conference of Steel and Composite for Engineering Structures: ICSCES 2022*, (2023), 296–306. https://doi.org/10.1007/978-3-031-24041-6_24
29. D. Gürses, P. Mehta, S. M. Sait, S. Kumar, A. R. Yildiz, A multi-strategy boosted prairie dog optimization algorithm for global optimization of heat exchangers, *Mater. Test.*, **65** (2023), 1396–1404. https://doi.org/10.1515/mt-2023-0082
30. L. Abualigah, A. Diabat, C. L. Thanh, S. Khatir, Opposition-based Laplacian distribution with Prairie Dog Optimization method for industrial engineering design problems, *Comput. Methods Appl. Mech. Eng.*, **414** (2023), 116097. https://doi.org/10.1016/j.cma.2023.116097
31. S. Mirjalili, A. H. Gandomi, Chaotic gravitational constants for the gravitational search algorithm, *Appl. Soft Comput.*, **53** (2017), 407–419. https://doi.org/10.1016/j.asoc.2017.01.008
32. Q. Liu, N. Li, H. Jia, Q. Qi, L. Abualigah, Modified remora optimization algorithm for global optimization and multilevel thresholding image segmentation, *Mathematics*, **10** (2022), 1014. https://doi.org/10.3390/MATH10071014
33. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Softw.*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
34. S. Zhao, T. Zhang, S. Ma, M. Wang, Sea-horse optimizer: A novel nature-inspired metaheuristic for global optimization problems, *Appl. Intell.*, **53** (2023), 11833–11860. https://doi.org/10.1007/s10489-022-03994-3
35. H. Jia, X. Peng, C. Lang, Remora optimization algorithm, *Expert Syst. Appl.*, **185** (2021), 115665. https://doi.org/10.1016/j.eswa.2021.115665
36. S. Mirjalili, SCA: A sine cosine algorithm for solving optimization problems, *Knowl. Based Syst.*, **96** (2016), 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
37. A. Seyyedabbasi, F. Kiani, Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems, *Eng. Comput.*, **2022** (2022), 1–25. https://doi.org/10.1007/s00366-022-01604-x
38. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Softw.*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
39. S. Wang, K. Sun, W. Zhang, H. Jia, Multilevel thresholding using a modified ant lion optimizer with opposition-based learning for color image segmentation, *Math. Biosci. Eng.*, **18** (2021), 3092–3143. https://doi.org/10.3934/mbe.2021155
40. H. Jia, Y. Li, D. Wu, H. Rao, C. Wen, L. Abualigah, Multi-strategy remora optimization algorithm for solving multi-extremum problems, *J. Comput. Design Eng.*, **2023** (2023), qwad044. https://doi.org/10.1093/jcde/qwad044

41. G. Dhiman, V. Kumar, Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems, *Knowl. Based Syst.*, **165** (2019), 169–196. https://doi.org/10.1016/j.knosys.2018.11.024
42. B. Abdollahzadeh, F. S. Gharehchopogh, S. Mirjalili, Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems, *Int. J Intell. Syst.*, **36** (2021), 5887–5958. https://doi.org/10.1002/INT.22535
43. A. Faramarzi, M. Heidarinejad, S. Mirjalili, A. H. Gandomi, Marine predators algorithm: A nature-inspired metaheuristic, *Expert Syst. Appl.*, **152** (2020), 113377. https://doi.org/10.1016/j.eswa.2020.113377
44. A. Babalik, A. C. Cinar, M. S. Kiran, A modification of tree-seed algorithm using Deb's rules for constrained optimization, *Appl. Soft. Comput.*, **63** (2018), 289–305. https://doi.org/10.1016/j.asoc.2017.10.013
45. A. Kaveh, M. Khayatazad, A new metaheuristic method: Ray optimization, *Comput. Struct.*, **112** (2012), 283–294. https://doi.org/10.1016/j.compstruc.2012.09.003
46. M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search algorithm for solving optimization problems, *Appl. Math. Comput.*, **188** (2007), 1567–1579. https://doi.org/10.1016/j.amc.2006.11.033
47. Q. He, L. Wang, An effective co-evolutionary particle swarm optimization for constrained engineering design problems, *Eng. Appl. Artif. Intell.*, **20** (2007), 89–99. https://doi.org/10.1016/j.engappai.2006.03.003
48. A. G. Hussien, M. Amin, M. A. E. Aziz, A comprehensive review of moth-flame optimisation: Variants, hybrids, and applications, *J. Exp. Theory Artif. Intell.*, **32** (2020), 705–725. https://doi.org/10.1080/0952813X.2020.1737246
49. V. K. Kamboj, A. Nandi, A. Bhadoria, S. Sehgal, An intensify harris hawks optimizer for numerical and engineering optimization problems, *Appl. Soft Comput.*, **89** (2020), 106018. https://doi.org/10.1016/j.asoc.2019.106018
50. D. Wu, H. Rao, C. Wen, H. Jia, Q. Liu, L. Abualigah, Modified sand cat swarm optimization algorithm for solving constrained engineering optimization problems, *Mathematics*, **10** (2022), 4350. https://doi.org/10.3390/MATH10224350
51. L. Abualigah, A. Diabat, S. Mirjalili, M. A. Elaziz, A. H. Gandomi, The arithmetic optimization algorithm, *Comput. Methods Appl. Mech. Eng.*, **376** (2021), 113609. https://doi.org/10.1016/J.CMA.2020.113609
52. L. Abualigah, M. A. Elaziz, P. Sumari, W. G. Zong, A. H. Gandomi, Reptile search algorithm (RSA): A nature-inspired metaheuristic optimizer, *Expert Syst. Appl.*, **191** (2021), 116158. https://doi.org/10.1016/J.ESWA.2021.116158
53. S. Lu, H. M. Kim, A regularized inexact penalty decomposition algorithm for multidisciplinary design optimization problemswith complementarity constraints, *J. Mech. Des.*, **132** (2010), 041005. https://doi.org/10.1115/1.4001206
54. M. Song, H. Jia, L. Abualigah, Q. Liu, Z. Lin, D. Wu, et al., Modified harris hawks optimization algorithm with exploration factor and random walk strategy, *Comput. Intell. Neurosci.*, **2022** (2022), 23. https://doi.org/10.1155/2022/4673665
55. V. Hayyolalam, A. A. P. Kazem, Black widow optimization algorithm: A novel metaheuristic approach for solving engineering optimization problems, *Eng. Appl. Artif. Intell.*, **87** (2020), 103249. https://doi.org/10.1016/j.engappai.2019.103249

56. P. Trojovsky, M. Dehghani, Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems, *Front. Mech. Eng.*, **8** (2023), 136. https://doi.org/10.3389/FMECH.2022.1126450
57. S. Wang, A. G. Hussien, H. Jia, L. Abualigah, R. Zheng, Enhanced remora optimization algorithm for solving constrained engineering optimization problems, *Mathematics*, **10** (2022), 1696. https://doi.org/10.3390/MATH10101696
58. Q. He, L. Wang, A hybrid particle swarm optimization with a feasibilitybased rule for constrained optimization, *Appl. Math. Comput.*, **186** (2007), 1407–1422. https://doi.org/10.1016/j.amc.2006.07.134
59. A. Laith, Y. Dalia, A. E. Mohamed, A. E. Ahmed, A. A. A. Mohammed, H. G. Amir, Aquila Optimizer: A novel metaheuristic optimization algorithm, *Comput. Ind. Eng.*, **157** (2021), 107250. https://doi.org/10.1016/J.CIE.2021.107250
60. J. You, H. Jia, D. Wu, H. Rao, C. Wen, Q. Liu, et al., Modified artificial gorilla troop optimization algorithm for solving constrained engineering optimization problems, *Mathematics*, **11** (2023), 1256. https://doi.org/10.3390/MATH11051256
61. A. H. Gandomi, X. S. Yang, A. H. Alavi, Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems, *Eng. Comput.*, **29** (2023), 17–35. https://doi.org/10.1007/s00366-011-0241-y
62. S. Li, H. Chen, M. Wangm, A. A. Heidari, S. Mirjalili, Slime mould algorithm: A new method for stochastic optimization, *Future Gener. Comput. Syst.*, **111** (2020), 300–323. https://doi.org/10.1016/j.future.2020.03.055
63. A. H. Gandomi, X. Yang, A. H. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, *Neural Comput. Appl.*, **22** (2023), 1239–1255. https://doi.org/10.1007/s00521-012-1028-9
64. M. Efren, C. Carlos, An empirical study about the usefulness of evolution strategies to solve constrained optimization problem, *Int. J. Gen. Syst.*, 37 (2008), 443–473. https://doi.org/10.1080/03081070701303470
65. H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm—A novel metaheuristic optimization method for solving constrained engineering optimization problems, *Comput. Struct.*, **110** (2012), 151–166. https://doi.org/10.1016/j.compstruc.2012.07.010
66. G. I. Sayed, A. Darwish, A. E. Hassanien, A new chaotic multi-verse optimization algorithm for solving engineering optimization problems, *J. Exp. Theor. Artif. Intell.*, **30** (2018), 293–317. https://doi.org/10.1080/0952813X.2018.1430858