Enhancing Optimization Efficiency: Walrus Optimiser with Early Exploration and Late Convergent Exploitation

Divam,Vedanshi,Prabhnoor
Netaji Subhas University of Technology, Dwarka, Delhi, 110078, India

Abstract

The Walrus Optimizer (WO) is a recently developed nature-inspired optimization algorithm that draws inspiration from the foraging behavior and social interactions of walruses. The algorithm mimics the way walruses search for food, communicate with each other, and coordinate their movements to maximize their collective success. By introducing adaptive strategies in the exploitation and exploration phases, our modified algorithm demonstrates improved performance in navigating rugged landscapes and escaping local minima. When tested on the benchmark functions the modified WOA shows enhanced convergence speed and better solution quality as compared to the original WOA at the expense of computation power. The results of the modified WOA either outperforms or show competitive results when compared with other meta-heuristic based algorithms.

Keywords: Walrus Optimization Algorithm, Metaheuristic Optimization, Exploration, Exploitation, Adaptive Strategies, Global Optimization, Nature-Inspired Algorithms, Local Optima, Search Space Exploration, Optimization Efficiency.

1. Introduction

Metaheuristics, renowned for their adaptability and versatility, are widely applied across various fields [1, 2, 3]. Among these, nature-inspired optimization algorithms stand out, drawing inspiration from natural phenomena such as evolution, swarm behavior, and genetic mechanisms [4, 5]. Algorithms like genetic algorithms [6], simulated annealing [7], and particle swarm optimization [8] offer efficient solutions to intricate problems, where conventional methods may falter, making them indispensable in addressing real-world challenges [9].There are several other categories of meta heursitic algorithms like those based on physical methods which typically occur in physics and chemistry[10] like Gravitational Search Algorithm (GSA)[11],Archimedes Optimization Algorithm[12],other category is human behaviour algorithms like teaching learning based optimisation (TLBO)[13],rich/poor optimization(RPO)[14], another category contains algorithms based on swarm intelligence[15,16] which draws inspiration from swarm behaviours in nature like Swarm Optimization (PSO)[17], Fire Hawk Optimizer(FHO)[18], Grey Wolf Optimization (GWO)[19],], Bat Flower pollination (BFP)[21], Ant Colony Optimization (ACO)[20] etc. The Walrus Optimization Algorithm (WOA)[21], a recent addition to the realm of nature-inspired optimization methods, brings forth innovative techniques inspired by the nimble and adaptive behaviour of walrus in quest of resources. WOA employs adaptive step to swiftly converge towards optimal solutions in continuous search spaces. The is still the provision to find a better solution than the existing ones by developing a more robust and efficient algorithm. This motivated our work to develop a novel population-based algorithm inspired by the activities of the walrus in the wild.

Our main contribution of this research can be expressed as follows:

- the introduction of an adaptive step size in the exploration phase enhances the Walrus Optimizer Algorithm's ability to explore broadly in early stages and exploit efficiently in later stages, ultimately contributing to better optimization performance.

These phases are repeated subject to termination criteria to find optimal solutions to optimization problems. • The various stages of WOA are described and mathematically modeled. • The efficacy and robustness of the WOA are evaluated by solving benchmark functions of CEC 2014 and CEC 2017. • Also, the performance of WOA in solving real-world problems is evaluated using engineering design problems. • The optimization results obtained from WOA are compared with nine state-of-the-art algorithms to analyze the proposed algorithm's capability

2. Original Walrus Optimization Algorithms

Walrus is a big fippered marine mammal with a discontinuous distribution in the Arctic Ocean and subarctic waters of the Northern Hemisphere around the North Pole52. Adult walruses are easily identifable with their large whiskers and tusks. Walruses are social animals who spend most of their time on the sea ice, seeking benthic bivalve mollusks to eat. The most prominent feature of walruses is the long tusks of this animal. These are elongated canines seen in both male and female species that may weigh up to 5.4 kg and measure up to 1 m in length. Males' tusks are slightly thicker and longer and are used for dominance, fighting, and display. The most muscular male with the longest tusks dominates the other group members and leads them. As the weather warms and the ice melts in late summer, walruses prefer to migrate to outcrops or rocky beaches. These migrations are very dramatic and involve massive aggregations of walruses54. The walrus has just two natural predators due to its large size and tusks: the polar bear and the killer whale (orca). Observations show that the battle between a walrus and a polar bear is very long and exhausting, and usually, polar bears withdraw from the fight after injuring the walrus. However, walruses harm polar bears with their tusks during this battle. In the fight against walruses, killer whales can hunt them successfully, with minimal and even no injuries.

2.1. Population initialization

The WOA is a population-based optimization algorithm that uses randomly initialized walruses (X) as search agents. The search agents are defined as a nxd matrix of candidate solutions as defined in Eq. (1). The WOA uses the problem's constraint of upper bound (UB) and lower bound (LB) to stochastically define the range of values that the population vector can take.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,m} \end{bmatrix}_{N\times m}, \quad (1)$$

where X is the walruses' population, Xi is the ith walrus (candidate solution), xi,j is the value of the jth decision variable suggested by the ith walrus, N is the number of walruses, and m is the number of decision variables. As mentioned, each walrus is a candidate solution to the problem, and based on its suggested values for the decision variables, the objective function of the problem can be evaluated. Te estimated values for the objective function obtained from walruses are specifed in

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1}, \quad (2)$$

where F is the objective function vector and Fi is the value of the objective function evaluated based on the ith walrus. Objective function values are the best measure of the quality of candidate solutions. Te candidate solution that results in the evaluation of the best value for the objective function is known as the best member. On the other hand, the candidate solution that results in the worst value for the objective function is called the worst member. According to the update of the values of the objective function in each iteration, the best and worst members are also updated.

2.2. Exploration

Walruses have a varied diet, feeding on more than sixty species of marine organisms, such as sea cucumbers, tunicates, sof corals, tube worms, shrimp, and various mollusks. However, walrus prefers benthic bivalve mollusks, particularly clams, for which it forages by grazing around the sea foor, seeking and detecting food with its energetic fipper motions and susceptible vibrissae58. In this search process, the strongest walrus with the tallest tusks guides the other walrus in the group to fnd food. Te length of the tusks in the walruses is similar to the quality of the objective function values of the candidate solutions. Terefore, the best candidate solution with the best value for the objective function is considered the strongest walrus in the group. Tis search behavior of the walruses leads to diferent scanning areas of the search space, which improves the exploration power of the WaOA in the global search. Te process of updating the position of walruses is mathematically modeled based on the feeding mechanism under the guidance of the most vital member of the group, using (3) and (4). In this process, a new position for walrus is first generated according to (3). This new position replaces the previous position if it improves the objective function's value; this concept is modeled in (4).

```
I = round(1 + rand(1, 1));
X_P1(i, :) = X(i, :) + rand(1, dim) .* (SW - I .* X(i, :)); Eq(3)
```

$$X_i = \begin{cases} X_i^{P_1}, & F_i^{P_1} < F_i, \\ X_i, & else, \end{cases}$$

2.3. Migration

One of the natural behaviors of walruses is their migration to outcrops or rocky beaches due to the warming of the air in late summer. Tis migration process is employed in the WaOA to guide the walruses in the search space to discover suitable areas in the search space. Tis behavioral mechanism is mathematically modeled using (5) and (6). Tis modeling assumes that each walrus migrates to another walrus (randomly selected) position in another area of the search space. Terefore, the proposed new position is frst generated based on (5). Ten according to (6), if this new position improves the value of the objective function, it replaces the previous position of walrus

$$x_{i,j}^{P_2} = \begin{cases} x_{i,j} + rand_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), F_k < F_i; \\ x_{i,j} + rand_{i,j} \cdot \left(x_{i,j} - x_{k,j}\right), else, \end{cases}$$

$$X_i = \begin{cases} X_i^{P_2}, & F_i^{P_2} < F_i; \\ X_i, & else, \end{cases}$$

2.3. Exploitation

Walruses are always exposed to attacks by the polar bear and the killer whale. Te strategy of escaping and fghting these predators leads to a change in the position of the walruses in the vicinity of the position in which they are located. Simulating this natural behavior of walruses improves the WaOA exploitation power in the local search in problem-solving space around candidate solutions. Since this process occurs near the position of each walrus, it is assumed in the WaOA design that this range of walrus position change occurs in a corresponding walrus-centered neighborhood with a certain radius. Considering that in the initial iterations of the algorithm, priority is given to global search in order to discover the optimal area in the search space, the radius of this neighborhood is considered variable so that it is frst set at the highest value and then becomes smaller during the iterations of the algorithm. For this reason, local lower/upper bounds have been used in this phase of WaOA to create a variable radius with algorithm repetitions. For simulation of this phenomenon in WaOA, a neighborhood is assumed around each walrus, which frst is generated a new position randomly in this neighborhood using (7) and (8). then if the value of the objective function is improved, this new position replaces the previous position according to (9).

$$x_{i,j}^{P_3} = x_{i,j} + \left(lb_{local,j}^{t} + \left(ub_{local,j}^{t} - rand \cdot lb_{local,j}^{t}\right)\right), \tag{7}$$

$$Local\ bounds : \begin{cases} lb_{local,j}^{t} = \frac{lb_j}{t}, \\ ub_{local,j}^{t} = \frac{ub_j}{t}, \end{cases} \tag{8}$$

$$X_i = \begin{cases} X_i^{P_3}, & F_i^{P_3} < F_i ; \\ X_i, & else , \end{cases} \tag{9}$$

3. Limitation of Original WOA

3.1. Limited Exploration:

With a fixed step size, there's a risk of inadequate exploration, especially in the early stages of optimization.

The algorithm might get stuck in local optima or fail to explore diverse regions of the search space thoroughly.

Limited exploration can prevent the algorithm from discovering potentially better solutions, particularly in complex and multimodal landscapes.

3.2.2 Convergence Issues:

Due to the fixed step size, the original WOA might struggle with convergence, especially towards the later stages of optimization.

In scenarios where fine-tuning of solutions is required for convergence to the global optimum, a fixed step size might be too restrictive.

This could result in premature convergence or slow convergence rates, particularly for challenging optimization problems

3.3. Difficulty in Escaping Local Optima:

Without adaptive adjustments, the original WOA might struggle to escape local optima efficiently. Fixed step sizes might limit the algorithm's ability to explore alternative regions of

the search space, particularly when encountering rugged landscapes with multiple local optima.

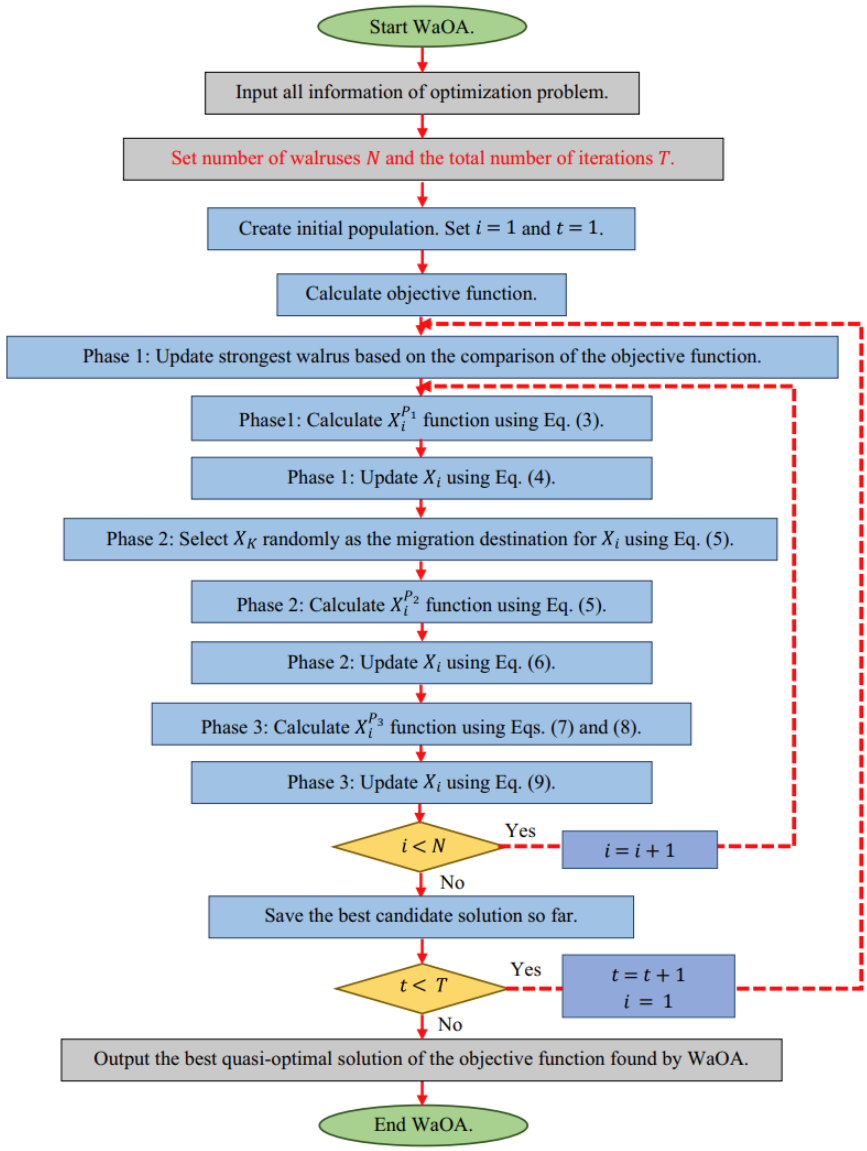


Figure 1: Flowchart of Proposed algorithm

4. Proposed algorithm

To address inherent limitations in the foundational Walrus Optimization Algorithm (WaOA) and to enhance its performance, we propose a modified variant incorporating variable step size within its exploitation phase. This section presents an overview of our proposed algorithm, elucidating the rationale behind the integration of a variable step size into the WaOA framework.The decision to introduce a variable step size stems from its proven efficacy in enhancing the exploration capabilities of optimization algorithms. By augmenting the traditional WaOA with a variable step size during the exploitation phase, we aim to capitalize on the inherent advantages of this stochastic search strategy.

In our modified algorithm, the reliance on randomly generated numbers to govern exploration and exploitation has been replaced with a deterministic strategy based on iteration count.

This step size modification captures how walruses display a remarkable adaptation to navigate the ever-changing landscape. During their journey across the frozen expanse, they employ a dynamic step size strategy that enhances their ability to locate the most bountiful feeding grounds.

In the early stages of their migration, walruses take larger, exploratory strides, allowing them to cover vast distances and investigate new territories with potential for rich resources. This expansive exploration phase ensures they do not become confined to a limited area, missing out on abundant opportunities.

As they near promising regions, the walruses' step sizes gradually diminish, enabling them to exploit the local surroundings more meticulously. Their smaller, refined steps during this exploitation phase allow them to scrutinize every nook and crevice, ensuring they fully capitalize on the available sustenance.

4.1. Phase One: Exploration

Original Code:

In the original code, the exploration phase relied on a fixed step size for all iterations. This means that the algorithm's exploration ability remained constant throughout the optimization process. There was no adaptation of the step size based on the progress of the algorithm.

Modified Code:

In the modified code, an adaptive step size mechanism has been introduced in the exploration phase. The step size decreases linearly with the number of iterations (adaptive_step_size = step_size * (1 - t / Max_iteration)). This allows the algorithm to explore the solution space more extensively in the early iterations when the step size is larger, and gradually shift towards exploitation in the later iterations as the step size decreases. This adaptation enhances the algorithm's capability to balance exploration and exploitation dynamically, potentially leading to faster convergence and improved solution quality.

4.2. Phase Two: Migration

Original Code:

In the original code, the migration phase was static, with no consideration for adaptive adjustments based on the optimization progress. The algorithm randomly selected another individual from the population and performed migration operations based on fixed rules.

Modified Code:

In the modified code, there's no modification specifically addressing the migration phase. However, the introduction of the adaptive step size in the exploration phase indirectly affects the migration phase as well. With a decreasing step size, the algorithm tends to shift towards exploitation in the later iterations, which might affect the behavior of the migration phase. However, direct modifications targeting the migration phase could further enhance the algorithm's performance.

4.3. Phase Three: Exploitation

Original Code:

In the original code, the exploitation phase also lacked adaptive mechanisms. The algorithm applied fixed rules for escaping and fighting against predators, without adjusting these strategies based on the optimization progress.

Modified Code:

In the modified code, similar to the exploration phase, an adaptive step size mechanism has been introduced in the exploitation phase as well. The step size decreases linearly with the

number of iterations, allowing the algorithm to focus more on exploitation in the later stages. This adaptation enhances the algorithm's ability to exploit promising regions of the solution space effectively as the optimization progresses.

Overall Improvement:

The modified code is better than the original code because it introduces adaptive mechanisms, particularly in the exploration and exploitation phases, through the implementation of a variable step size. This adaptation enables the algorithm to dynamically adjust its behavior based on the optimization progress, leading to potentially faster convergence and improved solution quality. Additionally, while not explicitly modified, the introduction of the adaptive step size may indirectly influence the behavior of other phases, such as migration, contributing to overall algorithm performance enhancements..

4.6. Pseudocode for proposed algorithm

Algorithm 1: Modified Walrus Optimisation Algorithm

```
Initialize population X with random positions
Initialize best solution Xbest
Initialize step_size = 2  // Initial step size

For each iteration t = 1 to Max_iterations
   Update the best solution Xbest
   For each walrus i in the population
      // Exploration Phase (Feeding Strategy) with Adaptive Step Size
      adaptive_step_size = step_size * (1 - t / Max_iterations)  // Calculate adaptive step size
      Update position X_P1[i] using adaptive_step_size and Xbest  // Eq(3)
      Enforce search space boundaries for X_P1[i]
      Evaluate fitness F_P1 for X_P1[i]
      If F_P1 is better than current fitness of walrus i
         Update position X[i] = X_P1[i]
         Update fitness of walrus i

      // Migration Phase
      Perform migration operation on X[i] using another walrus position  // Eq(5)
      Enforce search space boundaries
      Evaluate fitness F_P2 for updated position
      If F_P2 is better than current fitness of walrus i
         Update position X[i] with new position
         Update fitness of walrus i

      // Exploitation Phase (Escaping and Fighting Predators)
      Generate new position X_P3[i] based on current iteration  // Eq(7)
      Enforce search space boundaries for X_P3[i]
      Evaluate fitness F_P3 for X_P3[i]
      If F_P3 is better than current fitness of walrus i
         Update position X[i] = X_P3[i]
         Update fitness of walrus i

   End For
   Update best_so_far and average fitness
End For

Return Xbest as the best solution
```

5. Algorithms, test problems and comparison criteria

This section presents the design of experiments conducted to evaluate the performance of WOA. Thirty(30) CEC 2014 test functions, thirty (29) CEC 2017 test functions, and 13 selected optimization design problems in the engineering domain were used to evaluate the WOA. The results obtained were compared with that of the following algorithms:-

5.1. Algorithms and comparison criteria

- Walrus Optimization Algorithm (WOA)
- Modified Walrus Optimization Algorithm (MWOA)
- Sand Cat Optimization Algorithm (SCSO)
- Arithmetic Otimization Algorithm (AOA)
- Fire Hawk Optimizer (FHO)
- Whale Optimization Algorithm (WOA)
- Grey Wolf Optimizer (GWO)
- Crayfish Optimization Algorithm (COA)

The population size and the maximum number of iterations are sensitive parameters and need to be tuned. For this study, the population size is tuned to 50, and the maximum number of iterations is tuned to 100.The stop criterium is the maximum number of iterations.

Typology	No.	Function name	Opt.
Unimodal Functions	1	Shifted and Rotated Bent Cigar	100
	2	Shifted and Rotated Sum of Different Power	200
	3	Shifted and Rotated Zakharov	300
Simple Multimodal Functions	4	Shifted and Rotated Rosenbrock	400
	5	Shifted and Rotated Rastrigin	500
	6	Shifted and Rotated Expanded Schaffer F6	600
	7	Shifted and Rotated Lunacek Bi-Rastrigin	700
	8	Shifted and Rotated Non-Continuous Rastrigin	800
	9	Shifted and Rotated Levy	900
	10	Shifted and Rotated Schwefel	1000
Hybrid Functions	11	Zakharov; Rosenbrock; Rastrigin	1100
	12	High-conditioned Elliptic; Modified Schwefel; Bent Cigar	1200
	13	Bent Cigar; Rosenbrock; Lunacek bi-Rastrigin	1300
	14	High-conditioned Elliptic; Ackley; Schaffer F7; Rastrigin	1400
	15	Bent Cigar; HGBat; Rastrigin; Rosenbrock	1500
	16	Expanded Schaffer F6; HGBat; Rosenbrock; Modified Schwefel	1600
	17	Katsuura; Ackley; Expanded Griewank plus Rosenbrock; Schwefel; Rastrigin	1700
	18	High-conditioned Elliptic; Ackley; Rastrigin; HGBat; Discus	1800
	19	Bent Cigar; Rastrigin; Griewank plus Rosenbrock; Weierstrass; Expanded Schaffer F6	1900
	20	HappyCat; Katsuura; Ackley; Rastrigin; Modified Schwefel; Schaffer F7	2000
Composition Functions	21	Rosenbrock; High-conditioned Elliptic; Rastrigin	2100
	22	Rastrigin; Griewank; Modified Schwefel	2200
	23	Rosenbrock; Ackley; Modified Schwefel; Rastrigin	2300
	24	Ackley; High-conditioned Elliptic; Griewank; Rastrigin	2400
	25	Rastrigin; HappyCat; Ackley; Discus; Rosenbrock	2500
	26	Expanded Schaffer F6; Modified Schwefel; Griewank; Rosenbrock; Rastrigin	2600
	27	HGBat; Rastrigin; Modified Schwefel; Bent Cigar; High-conditioned Elliptic; Expanded Schaffer F6	2700
	28	Ackley; Griewank; Discus; Rosenbrock; HappyCat; Expanded Schaffer F6	2800
	29	$f_{15}; f_{16}; f_{17}$	2900
	30	$f_{15}; f_{18}; f_{19}$	3000

5.2. CEC 2017 and CEC 2014 Benchmark functions

No.	Functions	$f(x^*)$	NO.	Functions	$f(x^*)$
F01	Rotated High Conditioned Elliptic Function	100	F16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
F02	Rotated Bent Cigar Function	200	F17	Hybrid Function 2	1700
F03	Rotated Discus Function	300	F18	Hybrid Function 2	1800
F04	Shifted and Rotated Rosenbrock's Function	400	F19	Hybrid Function 3	1900
F05	Shifted and Rotated Ackley's Function	500	F20	Hybrid Function 4	2000
F06	hifted and Rotated Weierstrass Function	600	F21	Hybrid Function 5	2100
F07	Shifted and Rotated Griewank's Function	700	F22	Hybrid Function 6	2200
F08	Shifted Rastrigin's Function	800	F23	Composition Function 1	2300
F09	Shifted and Rotated Rastrigin's Function	900	F24	Composition Function 2	2400
F10	Shifted Schwefel's Function	1000	F25	Composition Function 3	2500
F11	Shifted and Rotated Schwefel's Function	1100	F26	Composition Function 4	2600
F12	Shifted and Rotated Katsuura Function	1200	F27	Composition Function 5	2700
F13	Shifted and Rotated HappyCat Function	1300	F28	Composition Function 6	2800
F14	Shifted and Rotated HGBat Function	1400	F29	Composition Function 7	2900
F15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500	F30	Composition Function 8	3000

Search Range: $[-100, 100]^D$

Figure 2: CEC 2014 functions

6. Experimental results and discussion

6.1. Performance Comparison of WOA and MWOA on CEC2017

Function	Avg MWOA	Std dev	Avg WOA	Std dev	Diff	p values
1	1258.904625	713571.7679	302084.9722	713571.7679	70.4885124	0.05746
2	0	3349.872341	2935.43087	3349.872341	0	0
3	308.1795689	641.5189832	1167.685232	641.5189832	3.227501224	0.53951
4	421.5785615	21.93610349	427.8844229	21.93610349	-1.18358397	0.033874
5	521.6307906	0.094205868 84	520.4343202	0.094205868 84	-2.626370106	0.85338
6	605.9243715	1.311971459	604.7698091	1.311971459	-1.36441395	0.64142

7	745.4420753	0.3988945803	700.4759287	0.3988945803	-1.673756952	0.34783
8	824.1823225	7.569377434	823.0374833	7.569377434	-0.8770518006	0.20621
9	993.0451244	7.485594043	923.8400181	7.485594043	-4.788029755	0.28378
10	1671.116263	283.1848616	1527.4413	283.1848616	-117.9164601	0.54933
11	1120.027479	224.5693154	1826.722408	224.5693154	-0.4182260481	0.63088
12	19763.7279	0.3291535541	1200.69709	0.3291535541	-88.83452021	0.17613
13	1965.976577	0.1292727987	1300.367848	0.1292727987	22.86831053	0.072446
14	1438.449938	0.2383095822	1400.383613	0.2383095822	1.149262627	0.6204
15	1531.958076	2.088207304	1503.022923	2.088207304	-4.797739943	0.94696
16	1695.443125	0.2817687795	1602.883399	0.2817687795	1.370837625	0.12597
17	1741.197207	822.5642076	2277.619422	822.5642076	-1.241928093	0.49178
18	2238.041109	45.5830282	1887.012313	45.5830282	-74.98358516	0.61001
19	1922.041839	1.052024591	1903.243961	1.052024591	5.507857173	0.76183
20	2078.729811	35.84071765	2053.276988	35.84071765	-8.676960111	0.0069724
21	2200.004483	131.7135072	2301.114341	131.7135072	-0.0002835946161	0.00049818
22	2300.005375	61.6675116	2281.018318	61.6675116	-0.0003541551528	0.66273
23	2497.449717	0	2500	0	-2.550282737	nan
24	2598.003617	33.31283755	2558.504338	33.31283755	-1.996383125	0.25805
25	2700	15.30483038	2690.982404	15.30483038	0	0.78934
26	2800	0.08721015719	2700.243419	0.08721015719	3.999619251	0.33285
27	2900	104.1734321	2873.957301	104.1734321	-4.834795818	0.093204
28	3000	0.00163677767	3000.000232	0.00163677767	0	0.0041035
29	3150.197708	289196.0945	44193.23998	289196.0945	-3.319586189	0.0018575
30	3979.371692	447.5833689	4276.268557	447.5833689	41.09657187	0.023243

6.2 Performance Comparison of WOA and MWOA on CEC2014

Function	Avg WOA	Std dev	Avg WOA	Std dev	Diff	p value
1	107.2123	5.9826	302084.9722	713571.7679	-12923.2542	
2	1217.1392	2061.5459	2935.43087	3349.872341	-346068.786	

3	300.004	0.0038554	1167.685232	641.5189832	-18.9734	
4	402.5318	6.9153	427.8844229	21.93610349	0.2094	
5	519.612	2.8301	520.4343202	0.09420586884	-0.5887	
6	602.22	1.059	604.7698091	1.311971459	1.0974	
7	700.1059	0.064482	700.4759287	0.3988945803	-0.4886	
8	801.7909	1.3181	823.0374833	7.569377434	-2.5312	
9	913.6707	3.989	923.8400181	7.485594043	2.5835	
10	1113.5206	73.8081	1527.4413	283.1848616	47.5965	
11	1468.8332	178.5938	1826.722408	224.5693154	25.7991	
12	1200.026	0.023943	1200.69709	0.3291535541	-0.473	
13	1300.119	0.04331	1300.367848	0.1292727987	-0.0772	
14	1400.1176	0.044149	1400.383613	0.2383095822	-0.0337	
15	1500.9115	0.29243	1503.022923	2.088207304	-0.9324	
16	1602.55	0.3636	1602.883399	0.2817687795	0.0969	
17	1745.432	33.5232	2277.619422	822.5642076	-181.6287	
18	1805.1416	1.8591	1887.012313	45.5830282	-11.2456	
19	1901.3842	0.34758	1903.243961	1.052024591	-0.3508	
20	2002.6999	0.96018	2053.276988	35.84071765	-5.8905	
21	2107.527	8.1948	2301.114341	131.7135072	-71.8829	
22	2211.2926	9.7042	2281.018318	61.6675116	-10.1384	
23	2596.5121	99.8396	2500	0	0.8451	
24	2521.1617	5.9971	2558.504338	33.31283755	5.749	
25	2635.6525	12.085	2690.982404	15.30483038	1.6714	
26	2700.1197	0.042312	2700.243419	0.08721015719	-0.0622	
27	2759.0809	138.9916	2873.957301	104.1734321	47.9375	
28	3169.076	3.7095	3000.000232	0.00163677767	-1.6154	
29	3122.2986	6.6956	44193.23998	289196.0945	-39.7195	
30	3492.4395	28.3786	4276.268557	447.5833689	-46.8146	

6.3. Performance Comparison of MWOA with 7 different algorithms on CEC2014

Function	Value	MWOA	WOA	COA	FHO	GWO	AOA	SCSO	WOA
1	Mean	107.8074804	13493.44937	603036.9379	4676099.821	3966485.731	23943459.62	7849639.35	8349887.952
	Std Dev	59.13333227	24869760.29	1.64762E+12	3.45157E+12	1.80621E+13	2.84263E+14	4.49968E+13	5.68135E+13
	Best	100.3555988	6189.828395	12450.11743	1111603.995	95256.7605	3160645.612	81501.2108	343501.5044
2	Mean	775.0595373	352883.9558	5138.004635	644506156.3	10277637.83	4811010227	7641313.287	2813494.5
	Std Dev	2076451.761	31378615105	14424628.07	2.26497E+16	2.57079E+15	3.35219E+18	1.67578E+15	3.40817E+12
	Best	200.216956	74891.69887	383.8686701	332161114	18106.598	1527492320	1488.679763	176216.4669
3	Mean	300.003652	316.8580486	10972.97783	5709.062332	6106.029139	20490.03304	3617.234177	57003.6403
	Std Dev	1.55E-05	65.82745928	33356781.15	1084467.481	26005079.43	47377954.11	5547667.232	583322616.7
	Best	300.0004122	301.7525261	979.7252113	3781.51585	1238.800374	6466.368782	864.1546712	13428.03325
4	Mean	403.2260087	402.5906036	428.8976291	479.7440154	428.7184786	973.3621052	430.9503734	444.3548502
	Std Dev	90.0475268	3.36829239	258.5269938	169.362155	139.7493621	125157.5499	150.9090674	754.1138207
	Best	400.0000005	400.0170859	400.0412207	430.4841659	400.5839021	468.0230111	401.2000725	402.9601623
5	Mean	520.0121622	519.8552493	520.0605522	520.4212367	520.4403251	520.1080403	519.5566318	520.1867094

	Std Dev	0.000419675	5.718340407	0.689600605	0.006582385	0.007607279	0.001401198	7.086371691	0.013342596
	Best	519.9943725	503.2881178	514.3581925	520.2210576	520.2312413	520.0204208	506.6471056	520.018149
6	Mean	602.377058	601.1133589	604.8494	606.5228997	601.9258861	608.9218011	605.2363727	607.852819
	Std Dev	1.365629825	0.206371155	3.995372186	1.437193973	1.625096768	1.303162377	3.249855379	2.705524545
	Best	600.0258861	600.2655435	600.8389925	604.8465823	600.2968974	605.9961305	601.792165	604.4428248
7	Mean	700.1146689	700.6350653	700.1355475	706.249399	701.0950928	798.1035241	700.9761331	701.1928304
	Std Dev	0.0029054	0.015555189	0.011225818	0.768038311	0.395650699	1077.250275	0.511327481	0.51249862
	Best	700.0172554	700.3578073	700.0319639	704.1648836	700.4293071	728.8045191	700.1036546	700.4969744
8	Mean	801.6516328	804.1484292	809.2574312	833.3163166	809.8927358	834.5171714	831.6292719	838.1280134
	Std Dev	1.11560567	3.801262386	42.04359316	22.27534908	25.92438673	72.45363382	142.5172083	218.5630437
	Best	800.0000001	800.9984005	801.9970414	819.7387846	803.0007011	814.7873349	810.0429364	817.0543474
9	Mean	911.6099171	909.6665639	940.4257214	935.5548958	912.3601464	935.4711709	935.0965281	947.2174392
	Std Dev	16.02900196	8.604308561	143.332661	31.17802482	24.08422833	88.65723393	134.4861967	382.4996121
	Best	901.9899182	904.3025413	912.934458	921.2302877	903.0965751	915.9815389	914.727774	909.593886
10	Mean	1113.988235	1072.840373	1427.222661	2219.604287	1337.934872	1541.684482	1687.342058	1636.821107
	Std Dev	6702.951057	1806.502014	91128.23946	23611.23586	52059.00727	49244.05972	80962.1136	73978.7465
	Best	1006.892834	1005.669723	1027.43954	1877.938348	1022.90655	1007.398838	1054.478058	1073.152605

Function	Value	MWOA	WOA	COA	FHO	GWO	AOA	SCSO	WOA
11	Mean	1113.988235	1468.751495	1931.858591	2395.366502	1585.8276	1979.886165	1890.964768	2197.305422
	Std Dev	1113.988235	29949.95624	109174.6377	29481.7463	79526.30502	58710.52431	90997.47626	78129.62441
	Best	1113.988235	1114.661945	1292.382112	2067.825075	1125.386313	1352.627659	1169.790802	1362.233836
12	Mean	6702.951057	1200.44263	1200.559931	1201.307491	1201.092975	1200.448474	1200.349761	1200.807148
	Std Dev	6702.951057	0.021992336	0.109332289	0.049636878	0.241537015	0.04588997	0.033264698	0.144233858
	Best	6702.951057	1200.177018	1200.046535	1200.896493	1200.075595	1200.078668	1200.061398	1200.236728
13	Mean	1006.892834	1300.180211	1300.24055	1300.623962	1300.206397	1302.188001	1300.360931	1300.452361
	Std Dev	1006.892834	0.002049183	0.012829124	0.009530506	0.005664353	0.791886522	0.026796115	0.035241467
	Best	1006.892834	1300.067015	1300.055309	1300.371192	1300.07228	1300.316863	1300.129134	1300.151459
14	Mean	1400.130249	1400.147076	1400.327467	1400.479281	1400.304618	1417.624792	1400.414051	1400.282624
	Std Dev	0.001730154	0.001050405	0.027699245	0.039985259	0.031841285	61.18451227	0.03765115	0.041407821
	Best	1400.024917	1400.080007	1400.123327	1400.174537	1400.062193	1404.875805	1400.104134	1400.086949
15	Mean	1500.960822	1501.720743	1501.964819	1508.47857	1501.966962	1893.298262	1503.014099	1507.194955
	Std Dev	0.089472056	0.197091075	1.318137308	1.860401063	0.579634578	317269.6877	1.77006196	12.42100268
	Best	1500.475583	1500.77152	1500.47079	1504.09452	1500.597142	1507.62872	1501.259798	1501.636557
16	Mean	1602.619332	1602.433851	1602.873065	1603.288033	1602.617602	1603.543282	1603.115295	1603.506292
	Std Dev	0.196805079	0.135342472	0.225592399	0.040422526	0.160031176	0.074385789	0.139039946	0.150852764
	Best	1601.404564	1601.666174	1601.20979	1602.747658	1601.68427	1602.926022	1602.050161	1602.483133
17	Mean	1749.71572	1912.670413	18366.65908	9502.831253	40039.00582	298080.3908	13356.98902	129944.0828
	Std Dev	2146.318118	3076.962017	236329460.6	9406039.873	10438466256	37923941122	2467421463	72378966102
	Best	1702.21038	1788.93269	3218.093314	4767.089545	2642.481189	4501.217766	2320.636139	3235.130855
18	Mean	1805.177662	1815.565415	5622.179592	6389.822234	9959.629481	12547.15159	10195.21219	14717.21023
	Std Dev	5.999448713	16.21782443	20643069.35	9819221.135	41611760.55	107432800.2	61200985.9	133416299.9
	Best	1800.618947	1807.086807	1894.440008	2245.298847	1931.179908	1892.406553	2019.922284	2037.688894
19	Mean	1901.406078	1901.807808	1902.81655	1908.493774	1902.481056	1918.647176	1903.597846	1905.993936
	Std Dev	0.212690786	0.045158998	1.445323383	0.849222129	0.593062353	229.914929	1.09010079	2.066159198
	Best	1900.30523	1901.427734	1901.167826	1906.260327	1901.562806	1903.972817	1901.584628	1903.543489
20	Mean	2002.390041	2009.22011	4667.982611	3159.570965	7602.626911	8025.455178	6550.412576	8269.964858

	Std Dev	0.97062711	4.733101398	15804094.93	658673.2543	21656848.73	16207040.36	11769616.13	27538705.03
	Best	2000.533459	2003.870319	2130.550167	2146.231413	2083.803118	2110.465558	2074.035901	2123.451809

Function	Value	MWOA	WOA	COA	FHO	GWO	AOA	SCSO	WOA
21	Mean	2107.052031	2176.44272	6894.138224	41735.57284	8314.027915	7079.620327	8726.543374	108261.4579
	Std Dev	61.75224528	1291.431199	49518972.45	1389623773	20546837.36	10270010.05	25463093.57	35551267522
	Best	2100.647656	2117.937532	2373.413844	4808.843644	2752.634272	2909.464567	2659.383236	4987.932601
22	Mean	2209.286531	2219.578949	2231.119642	2251.109807	2264.865043	2370.934299	2291.074808	2295.102069
	Std Dev	92.79607779	31.20290902	132.2252997	127.9127273	2359.141172	8978.250038	3175.271883	6620.380172
	Best	2200.22491	2204.033096	2220.384102	2234.237846	2224.213625	2229.737607	2225.138336	2223.441095
23	Mean	2609.690316	2581.67411	2500	2645.579316	2631.994398	2500	2500	2637.290641
	Std Dev	6246.533108	6773.201659	0	48.17653008	8.503475602	0	0	64.56053926
	Best	2300.003113	2306.422104	2500	2604.324229	2629.462697	2500	2500	2630.095826
24	Mean	2521.686987	2516.238303	2598.783512	2547.883797	2523.25976	2578.139206	2578.8312	2571.601771
	Std Dev	35.10574572	13.89468755	73.99219059	20.00804955	114.7352981	549.8706355	879.0003625	729.8041856
	Best	2510.162492	2507.290724	2539.175584	2534.067461	2510.719242	2535.83415	2515.996155	2520.293975
25	Mean	2638.999833	2634.587385	2696.43423	2653.06587	2691.876921	2698.402337	2698.726718	2697.197726
	Std Dev	169.0475065	155.951071	207.0089464	59.33612275	511.5135322	23.65639822	77.8330475	68.31858726
	Best	2614.592179	2616.573336	2632.290011	2644.171047	2612.548283	2673.885728	2637.603895	2659.792981
26	Mean	2700.117793	2700.181826	2700.206908	2700.451941	2700.167961	2702.645847	2700.270502	2700.425351
	Std Dev	0.001396918	0.001427814	0.009025653	0.005359413	0.001771122	0.41677376	0.006967978	0.046531836
	Best	2700.063351	2700.103889	2700.076157	2700.294721	2700.080655	2700.660964	2700.125191	2700.148373
27	Mean	2735.315585	2710.419164	2888.342547	2766.552185	3005.608276	2890.103861	2880.753978	3056.694591
	Std Dev	11811.63482	2487.424331	2172.544114	17324.06556	18642.53574	1384.455246	3402.57587	37863.37251
	Best	2701.603204	2701.766413	2704.927904	2707.005714	2703.315569	2714.32668	2704.720598	2705.984519
28	Mean	3168.720041	3169.290247	3000	3129.471931	3225.188781	3012.227779	3000	3349.844139
	Std Dev	15.03462437	30.20329569	0	22948.20644	3845.983576	7475.929494	0	16285.05445
	Best	3156.827164	3156.951345	3000	3000.820033	3012.291611	3000	3000	3185.975315
29	Mean	3122.226639	3164.843317	161917.1677	98424.10572	46420.23192	122803.6107	4302.26091	73174.35575
	Std Dev	51.07212259	259.1376123	2.96534E+11	1.69659E+11	90786427530	1.36987E+11	3239456.221	1.16353E+11
	Best	3080.212181	3135.216784	3100	3476.156581	3206.443946	3100	3100	3309.263317
30	Mean	3495.630375	3547.210219	3916.98658	5251.038265	4055.594384	21761.54911	4761.492218	5372.637714

	Std Dev	941.4609167	959.7531893	275463.3386	883337.4616	305497.6896	1615655620	861920.9253	1179945.831
	Best	3455.077325	3489.112165	3200	4080.253591	3492.051473	3200	3532.579729	3807.526628

6.4. Performance Comparison of MWOA with 7 different algorithms on CEC2017

Function	Value	MWOA	WOA	COA	FHO	GWO	AOA	SCSO	WOA
1	Mean	107.8074804	13493.44937	603036.9379	4676099.821	3966485.731	23943459.62	7849639.35	8349887.952
	Std Dev	59.13333227	24869760.29	1.64762E+12	3.45157E+12	1.80621E+13	2.84263E+14	4.49968E+13	5.68135E+13
	Best	100.3555988	6189.828395	12450.11743	1111603.995	95256.7605	3160645.612	81501.2108	343501.5044
2	Mean	775.0595373	352883.9558	5138.004635	644506156.3	10277637.83	4811010227	7641313.287	2813494.5
	Std Dev	2076451.761	31378615105	14424628.07	2.26497E+16	2.57079E+15	3.35219E+18	1.67578E+15	3.40817E+12
	Best	200.216956	74891.69887	383.8686701	332161114	18106.598	1527492320	1488.679763	176216.4669
3	Mean	300.003652	316.8580486	10972.97783	5709.062332	6106.029139	20490.03304	3617.234177	57003.6403
	Std Dev	1.55E-05	65.82745928	33356781.15	1084467.481	26005079.43	47377954.11	5547667.232	583322616.7
	Best	300.0004122	301.7525261	979.7252113	3781.51585	1238.800374	6466.368782	864.1546712	13428.03325
4	Mean	403.2260087	402.5906036	428.8976291	479.7440154	428.7184786	973.3621052	430.9503734	444.3548502
	Std Dev	90.0475268	3.36829239	258.5269938	169.362155	139.7493621	125157.5499	150.9090674	754.1138207
	Best	400.0000005	400.0170859	400.0412207	430.4841659	400.5839021	468.0230111	401.2000725	402.9601623
5	Mean	520.0121622	519.8552493	520.0605522	520.4212367	520.4403251	520.1080403	519.5566318	520.1867094
	Std Dev	0.000419675	5.718340407	0.689600605	0.006582385	0.007607279	0.001401198	7.086371691	0.013342596
	Best	519.9943725	503.2881178	514.3581925	520.2210576	520.2312413	520.0204208	506.6471056	520.018149
6	Mean	602.377058	601.1133589	604.8494	606.5228997	601.9258861	608.9218011	605.2363727	607.852819
	Std Dev	1.365629825	0.206371155	3.995372186	1.437193973	1.625096768	1.303162377	3.249855379	2.705524545
	Best	600.0258861	600.2655435	600.8389925	604.8465823	600.2968974	605.9961305	601.792165	604.4428248
7	Mean	700.1146689	700.6350653	700.1355475	706.249399	701.0950928	798.1035241	700.9761331	701.1928304
	Std Dev	0.0029054	0.015555189	0.011225818	0.768038311	0.395650699	1077.250275	0.511327481	0.51249862
	Best	700.0172554	700.3578073	700.0319639	704.1648836	700.4293071	728.8045191	700.1036546	700.4969744
8	Mean	801.6516328	804.1484292	809.2574312	833.3163166	809.8927358	834.5171714	831.6292719	838.1280134
	Std Dev	1.11560567	3.801262386	42.04359316	22.27534908	25.92438673	72.45363382	142.5172083	218.5630437
	Best	800.0000001	800.9984005	801.9970414	819.7387846	803.0007011	814.7873349	810.0429364	817.0543474
9	Mean	911.6099171	909.6665639	940.4257214	935.5548958	912.3601464	935.4711709	935.0965281	947.2174392
	Std Dev	16.02900196	8.604308561	143.332661	31.17802482	24.08422833	88.65723393	134.4861967	382.4996121
	Best	901.9899182	904.3025413	912.934458	921.2302877	903.0965751	915.9815389	914.727774	909.593886
10	Mean	1113.988235	1072.840373	1427.222661	2219.604287	1337.934872	1541.684482	1687.342058	1636.821107

	Std Dev	6702.951057	1806.502014	91128.23946	23611.23586	52059.00727	49244.05972	80962.1136	73978.7465
	Best	1006.892834	1005.669723	1027.43954	1877.938348	1022.90655	1007.398838	1054.478058	1073.152605

Function	Value	MWOA	WOA	COA	FHO	GWO	AOA	SCSO	WOA
11	Mean	1113.988235	1468.751495	1931.858591	2395.366502	1585.8276	1979.886165	1890.964768	2197.305422
	Std Dev	1113.988235	29949.95624	109174.6377	29481.7463	79526.30502	58710.52431	90997.47626	78129.62441
	Best	1113.988235	1114.661945	1292.382112	2067.825075	1125.386313	1352.627659	1169.790802	1362.233836
12	Mean	6702.951057	1200.44263	1200.559931	1201.307491	1201.092975	1200.448474	1200.349761	1200.807148
	Std Dev	6702.951057	0.021992336	0.109332289	0.049636878	0.241537015	0.04588997	0.033264698	0.144233858
	Best	6702.951057	1200.177018	1200.046535	1200.896493	1200.075595	1200.078668	1200.061398	1200.236728
13	Mean	1006.892834	1300.180211	1300.24055	1300.623962	1300.206397	1302.188001	1300.360931	1300.452361
	Std Dev	1006.892834	0.002049183	0.012829124	0.009530506	0.005664353	0.791886522	0.026796115	0.035241467
	Best	1006.892834	1300.067015	1300.055309	1300.371192	1300.07228	1300.316863	1300.129134	1300.151459
14	Mean	1400.130249	1400.147076	1400.327467	1400.479281	1400.304618	1417.624792	1400.414051	1400.282624
	Std Dev	0.001730154	0.001050405	0.027699245	0.039985259	0.031841285	61.18451227	0.03765115	0.041407821
	Best	1400.024917	1400.080007	1400.123327	1400.174537	1400.062193	1404.875805	1400.104134	1400.086949
15	Mean	1500.960822	1501.720743	1501.964819	1508.47857	1501.966962	1893.298262	1503.014099	1507.194955
	Std Dev	0.089472056	0.197091075	1.318137308	1.860401063	0.579634578	317269.6877	1.77006196	12.42100268
	Best	1500.475583	1500.77152	1500.47079	1504.09452	1500.597142	1507.62872	1501.259798	1501.636557
16	Mean	1602.619332	1602.433851	1602.873065	1603.288033	1602.617602	1603.543282	1603.115295	1603.506292
	Std Dev	0.196805079	0.135342472	0.225592399	0.040422526	0.160031176	0.074385789	0.139039946	0.150852764
	Best	1601.404564	1601.666174	1601.20979	1602.747658	1601.68427	1602.926022	1602.050161	1602.483133
17	Mean	1749.71572	1912.670413	18366.65908	9502.831253	40039.00582	298080.3908	13356.98902	129944.0828
	Std Dev	2146.318118	3076.962017	236329460.6	9406039.873	10438466256	37923941122	2467421463	72378966102
	Best	1702.21038	1788.93269	3218.093314	4767.089545	2642.481189	4501.217766	2320.636139	3235.130855
18	Mean	1805.177662	1815.565415	5622.179592	6389.822234	9959.629481	12547.15159	10195.21219	14717.21023
	Std Dev	5.999448713	16.21782443	20643069.35	9819221.135	41611760.55	107432800.2	61200985.9	133416299.9
	Best	1800.618947	1807.086807	1894.440008	2245.298847	1931.179908	1892.406553	2019.922284	2037.688894
19	Mean	1901.406078	1901.807808	1902.81655	1908.493774	1902.481056	1918.647176	1903.597846	1905.993936
	Std Dev	0.212690786	0.045158998	1.445323383	0.849222129	0.593062353	229.914929	1.09010079	2.066159198
	Best	1900.30523	1901.427734	1901.167826	1906.260327	1901.562806	1903.972817	1901.584628	1903.543489
20	Mean	2002.390041	2009.22011	4667.982611	3159.570965	7602.626911	8025.455178	6550.412576	8269.964858

	Std Dev	0.97062711	4.733101398	15804094.93	658673.2543	21656848.73	16207040.36	11769616.13	27538705.03
	Best	2000.533459	2003.870319	2130.550167	2146.231413	2083.803118	2110.465558	2074.035901	2123.451809

Function	Value	MWOA	WOA	COA	FHO	GWO	AOA	SCSO	WOA
21	Mean	2107.052031	2176.44272	6894.138224	41735.57284	8314.027915	7079.620327	8726.543374	108261.4579
	Std Dev	61.75224528	1291.431199	49518972.45	1389623773	20546837.36	10270010.05	25463093.57	35551267522
	Best	2100.647656	2117.937532	2373.413844	4808.843644	2752.634272	2909.464567	2659.383236	4987.932601
22	Mean	2209.286531	2219.578949	2231.119642	2251.109807	2264.865043	2370.934299	2291.074808	2295.102069
	Std Dev	92.79607779	31.20290902	132.2252997	127.9127273	2359.141172	8978.250038	3175.271883	6620.380172
	Best	2200.22491	2204.033096	2220.384102	2234.237846	2224.213625	2229.737607	2225.138336	2223.441095
23	Mean	2609.690316	2581.67411	2500	2645.579316	2631.994398	2500	2500	2637.290641
	Std Dev	6246.533108	6773.201659	0	48.17653008	8.503475602	0	0	64.56053926
	Best	2300.003113	2306.422104	2500	2604.324229	2629.462697	2500	2500	2630.095826
24	Mean	2521.686987	2516.238303	2598.783512	2547.883797	2523.25976	2578.139206	2578.8312	2571.601771
	Std Dev	35.10574572	13.89468755	73.99219059	20.00804955	114.7352981	549.8706355	879.0003625	729.8041856
	Best	2510.162492	2507.290724	2539.175584	2534.067461	2510.719242	2535.83415	2515.996155	2520.293975
25	Mean	2638.999833	2634.587385	2696.43423	2653.06587	2691.876921	2698.402337	2698.726718	2697.197726
	Std Dev	169.0475065	155.951071	207.0089464	59.33612275	511.5135322	23.65639822	77.8330475	68.31858726
	Best	2614.592179	2616.573336	2632.290011	2644.171047	2612.548283	2673.885728	2637.603895	2659.792981
26	Mean	2700.117793	2700.181826	2700.206908	2700.451941	2700.167961	2702.645847	2700.270502	2700.425351
	Std Dev	0.001396918	0.001427814	0.009025653	0.005359413	0.001771122	0.41677376	0.006967978	0.046531836
	Best	2700.063351	2700.103889	2700.076157	2700.294721	2700.080655	2700.660964	2700.125191	2700.148373
27	Mean	2735.315585	2710.419164	2888.342547	2766.552185	3005.608276	2890.103861	2880.753978	3056.694591
	Std Dev	11811.63482	2487.424331	2172.544114	17324.06556	18642.53574	1384.455246	3402.57587	37863.37251
	Best	2701.603204	2701.766413	2704.927904	2707.005714	2703.315569	2714.32668	2704.720598	2705.984519
28	Mean	3168.720041	3169.290247	3000	3129.471931	3225.188781	3012.227779	3000	3349.844139
	Std Dev	15.03462437	30.20329569	0	22948.20644	3845.983576	7475.929494	0	16285.05445
	Best	3156.827164	3156.951345	3000	3000.820033	3012.291611	3000	3000	3185.975315
29	Mean	3122.226639	3164.843317	161917.1677	98424.10572	46420.23192	122803.6107	4302.26091	73174.35575
	Std Dev	51.07212259	259.1376123	2.96534E+11	1.69659E+11	90786427530	1.36987E+11	3239456.221	1.16353E+11
	Best	3080.212181	3135.216784	3100	3476.156581	3206.443946	3100	3100	3309.263317
30	Mean	3495.630375	3547.210219	3916.98658	5251.038265	4055.594384	21761.54911	4761.492218	5372.637714
	Std Dev	941.4609167	959.7531893	275463.3386	883337.4616	305497.6896	1615655620	861920.9253	1179945.831

	Best	3455.077325	3489.112165	3200	4080.253591	3492.051473	3200	3532.579729	3807.526628

7. Engineering Problems

The goal of applying WOA to engineering design problems is to find the minimum cost and values of the corresponding design parameters. The four (13) problems considered for our experiments are discussed in the next section.As explained previously, the proposed WOA was used to solve four (13) mechanical engineering design problems. The constraint handling was handled by the penalty method (simple scalar penalty functions), where the algorithm is punished when there is a constraint violation. Similarly, the results are presented using best, worst, average, standard deviation (SD), and median performance indicators. The mean, standard deviation, and Wilcoxon signed-rank test were used to compare the different algorithms.

7.1. Speed Reducer

A speed reducer is part of the gear box of mechanical system, and it is used in many other types of applications. The design of the speed reducer is a more challenging benchmark, because it involves seven design variables. the design of the speed reducer is considered with the face width (x1), the module of the teeth (x2), the number of teeth on pinion (x3), the length of the first shaft between bearings (x4), the length of the second shaft between bearings (x5), diameter of the first shaft (x6), and the diameter of the second shaft (x7). The objective is to minimize the total weight of the speed reducer while satisfying eleven constraints. The constraints include the limits on the bending stress of the gear teeth, surface stress, transverse deflections of shafts 1 and 2 due to transmitted force, and stresses in shafts 1 and 2

7.2. Tension/Compression Spring Design Problem

The Tension/Compression Spring Design problem (TCSD) is a continuous constrained problem. The problem is to minimize the volume V of a coil spring under a constant tension/compression load. The problem consists of three design variables. These are: the number of spring's active coils P = x1 [2, 15]; the diameter of the winding D = x2 [0.25, 1.3]; the diameter of the wire d = x3 [0.05, 2].

7.3. Pressure Vessel Design

A pressure vessel design problem involves creating a container to hold fluids or gases at high pressure safely. Key steps include analyzing requirements, selecting appropriate materials, calculating dimensions and thickness, conducting stress analysis, ensuring fatigue resistance, designing critical components like nozzles and flanges, performing welding and fabrication, testing for integrity, and adhering to regulatory standards. Proper operation and maintenance are crucial for safety and reliability throughout the vessel's service life.

7.4. Three-bar truss design problem

A three-bar truss problem involves analyzing the structural behavior of a simple truss composed of three bars connected by joints. The objective is usually to determine the internal forces, such as tension and compression, experienced by each member of the truss under applied loads and constraints. This analysis typically involves applying principles of statics and equilibrium to solve for unknown forces using methods like the method of joints or the method of sections. Three-bar truss problems serve as fundamental examples in structural analysis and are used to illustrate concepts such as equilibrium, force transmission, and load distribution in truss structures.

7.5. Design of gear train

The design of a gear train involves selecting the right gears, arranging them to achieve the desired speed and torque, ensuring smooth meshing, balancing loads, choosing suitable materials, and testing for performance and reliability. It's about efficiently transmitting power from one shaft to another in mechanical systems.

7.6. Cantilever beam

A cantilever beam is a structural element that is supported at only one end, while the other end is free to deflect under load. It is commonly used in civil and mechanical engineering applications to support loads over an empty space or span. The fixed end of the beam provides stability, while the free end can experience deflection and bending due to applied loads. Cantilever beams are often used in structures like bridges, balconies, diving boards, and shelf brackets. They are analyzed using principles of structural mechanics to determine factors such as stress, deflection, and structural stability.

7.7. Minimize I-beam vertical deflection

When a load is applied to an I-beam, which is a common structural element used in construction, it experiences vertical deflection. This deflection occurs due to the bending of the beam under the applied load. The magnitude of vertical deflection depends on several factors, including the material properties of the beam, its dimensions, the magnitude and distribution of the applied load, and the support conditions.

The vertical deflection of an I-beam is typically calculated using principles of structural mechanics, such as Euler-Bernoulli beam theory or finite element analysis. These methods allow engineers to determine the amount of bending that the beam undergoes and the resulting vertical displacement at various points along its length.

7.8. Tubular column design

The design of tubular columns involves determining the appropriate size, material, and shape to safely support vertical loads while considering factors like stability and buckling. Common steps include analyzing loads, selecting materials, choosing a geometric configuration, conducting stability analysis, adhering to design criteria, designing connections, and overseeing fabrication and construction. It requires a thorough understanding of structural engineering principles and collaboration between professionals to achieve safe and efficient structural systems.

7.9. Piston lever

A piston lever is a mechanical device that converts the linear motion of a piston into rotational motion. It consists of a piston, which moves back and forth within a cylinder, and a lever attached to the piston. When fluid (such as air or hydraulic fluid) is introduced into the cylinder, it pushes the piston in one direction, causing the lever to rotate about a pivot point. This rotational motion can be used to perform various tasks, such as lifting heavy loads or actuating other mechanical components.

Piston levers are commonly found in hydraulic systems, where the movement of the piston generates pressure to perform work. They are used in various applications across industries, including automotive, manufacturing, and construction.

7.10. Corrugated bulkhead design

A corrugated bulkhead design refers to the structure used to divide or separate compartments within a container, ship, or other large storage unit. The term "corrugated" refers to the presence of ridges or grooves on the surface of the bulkhead, which provides strength and rigidity to the structure.

Corrugated bulkheads are commonly used in shipping containers and vessels to prevent cargo from shifting during transport. They help maintain the structural integrity of the container or vessel and protect the goods from damage due to movement or impact.

The design of corrugated bulkheads involves considerations such as the material used, the thickness of the bulkhead, the spacing and depth of the corrugations, and the method of attachment to the surrounding structure. These factors are carefully chosen to ensure that the bulkhead can withstand the forces exerted on it during transportation, including changes in pressure, vibration, and impact.

7.11. Car side impact design

Car side impact design focuses on protecting vehicle occupants from injuries during collisions from the side. It involves features like side impact beams, energy-absorbing materials, and side airbags to minimize the force transferred to occupants. The design aims to reinforce the vehicle's structure and create crumple zones that absorb impact energy, enhancing overall safety.

7.12. Design of welded beam design

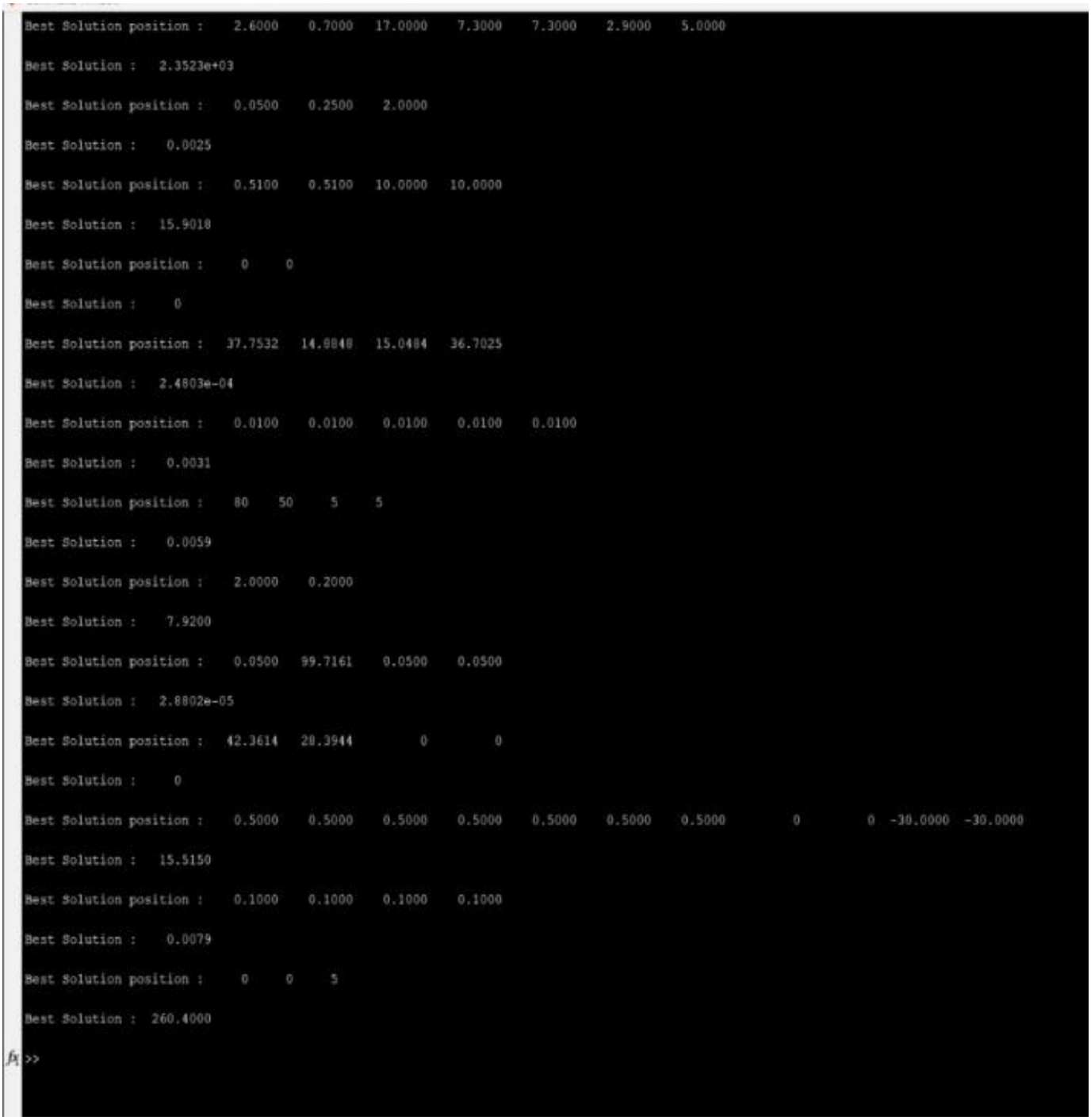
The design of welded beams involves selecting the right steel grade, determining the beam's shape and dimensions, choosing appropriate weld types and sizes, designing connections, and analyzing fatigue resistance. It aims to ensure structural integrity, optimal performance, and cost-effectiveness.

7.13. Reinforced concrete beam design

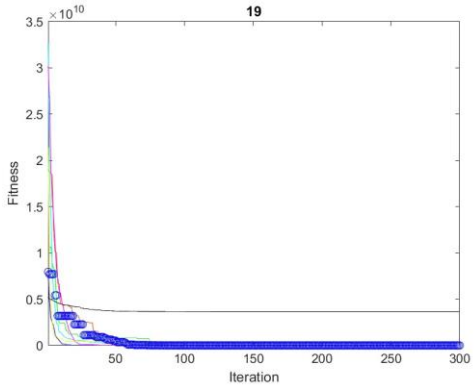
Reinforced concrete beam design involves determining the dimensions of the beam, selecting the reinforcement layout, and calculating the required steel reinforcement to resist bending, shear, and any other applied loads. It also includes considerations for factors like deflection, cracking, and durability. The design process typically follows structural engineering principles and codes to ensure the beam's safety and efficiency in carrying applied loads.

8. Results and Discussion

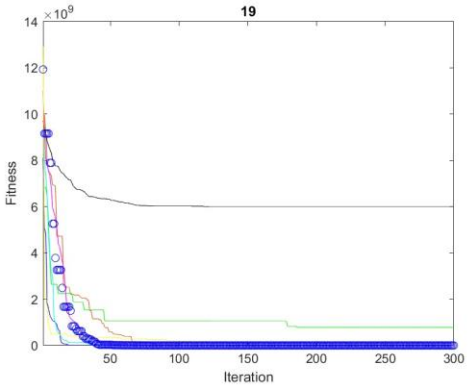
Engineering problems



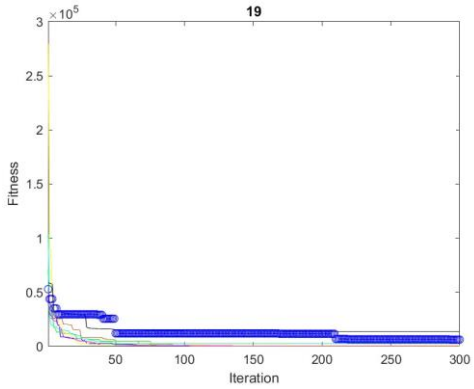
8.0.1. Convergence Curves for CEC2014

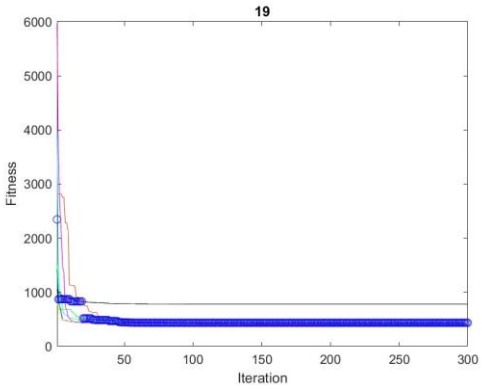


CEC2014-F1

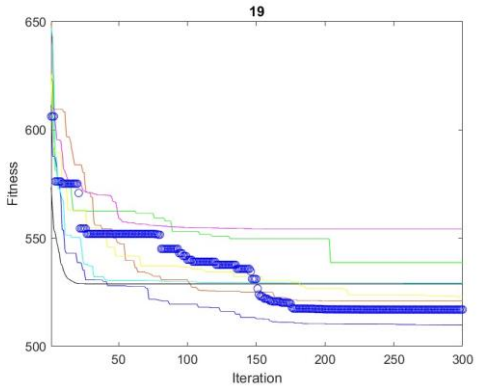


CEC2014-F2

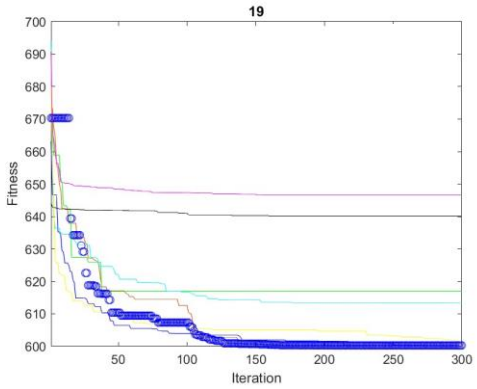




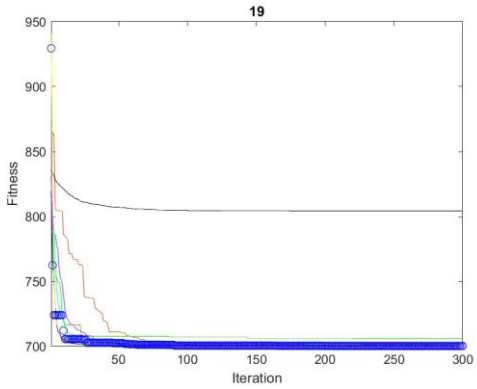
CEC2014-F3



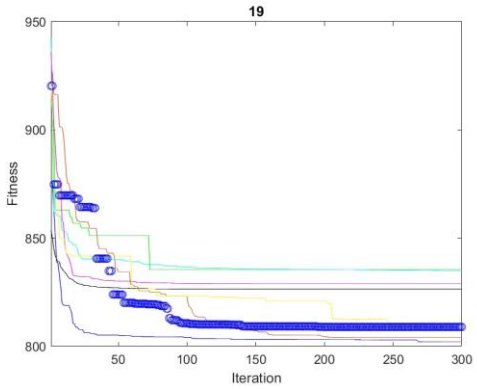
CEC2014-F4



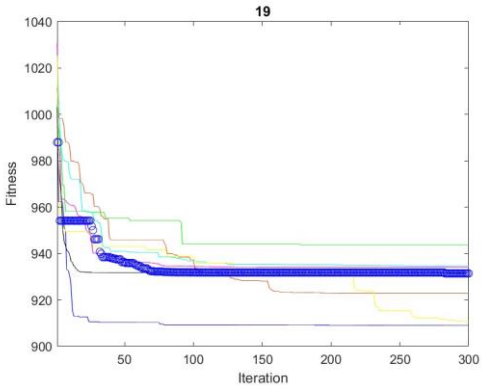
CEC2014-F5



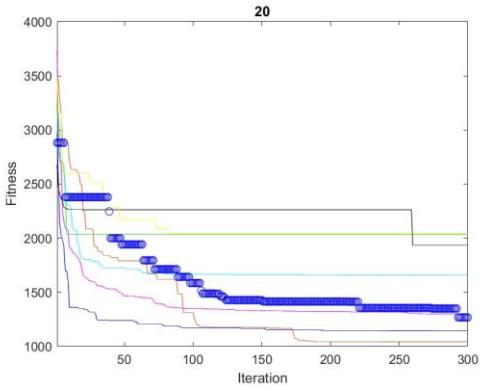
CEC2014-F6



CEC2014-F7

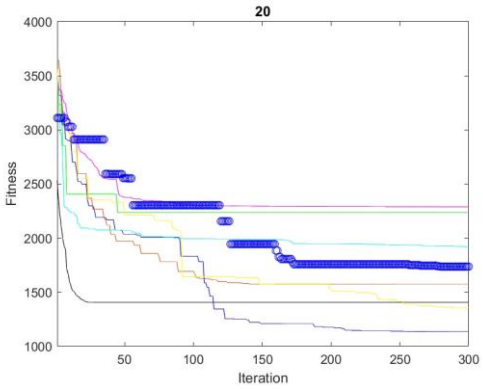


CEC2014-F8

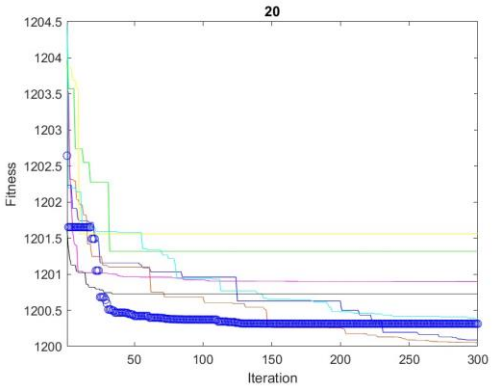


CEC2014-F9

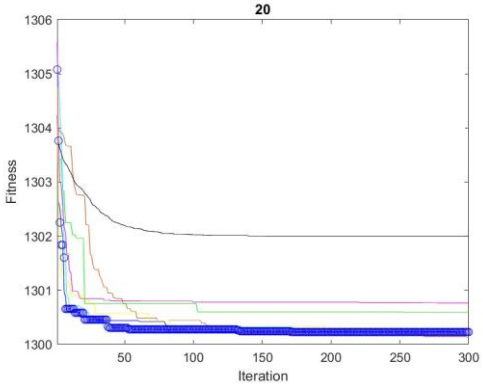
CEC2014-F10



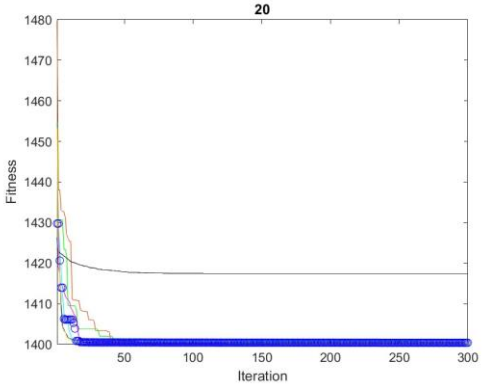
CEC2014-F11



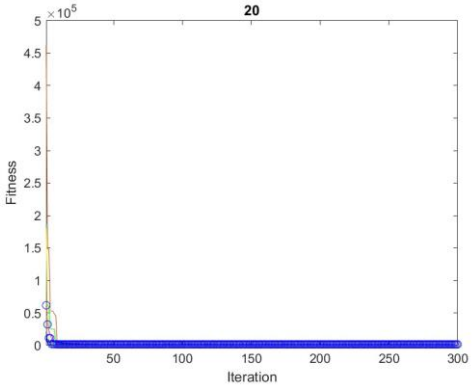
CEC2014-F12



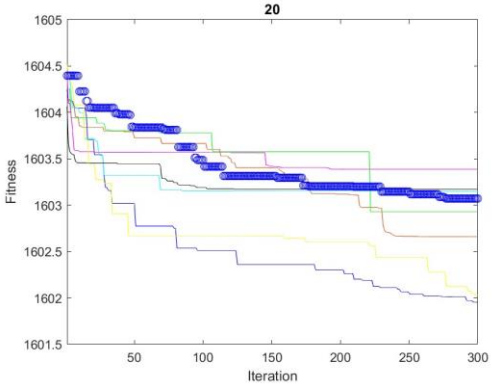
CEC2014-F13



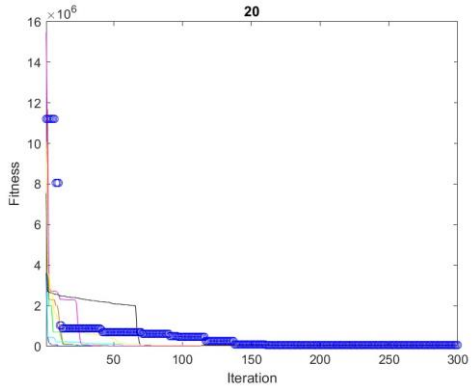
CEC2014-F14



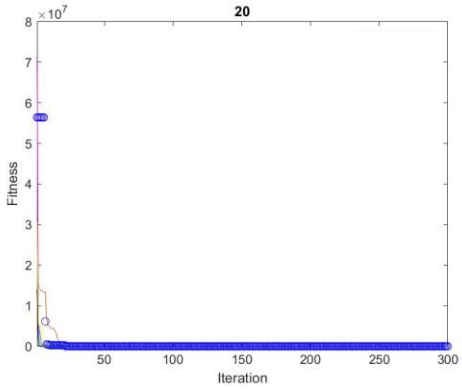
CEC2014-F15



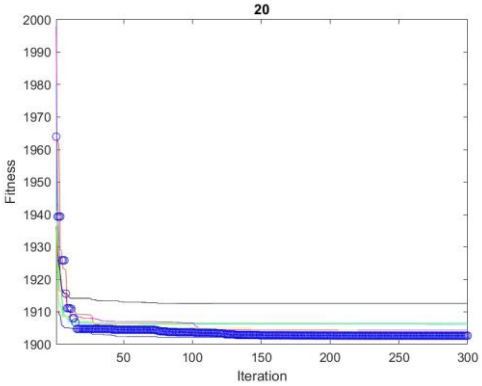
CEC2014-F16



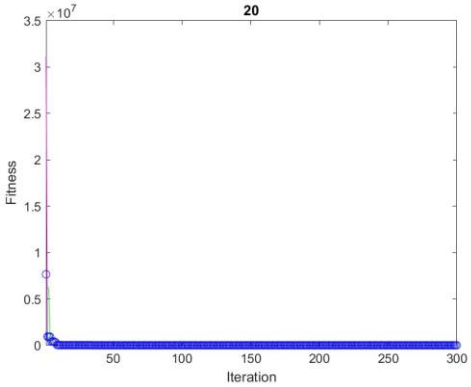
CEC2014-F17



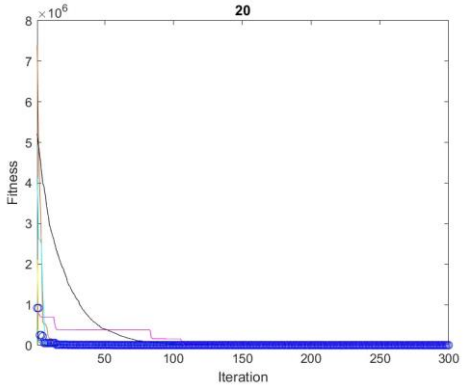
CEC2014-F18



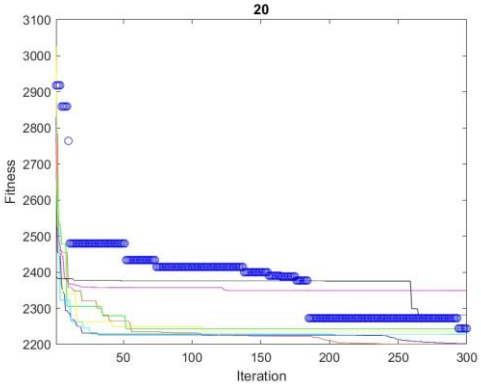
CEC2014-F19



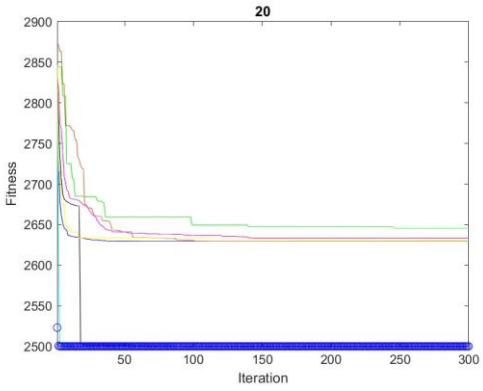
CEC2014-F20



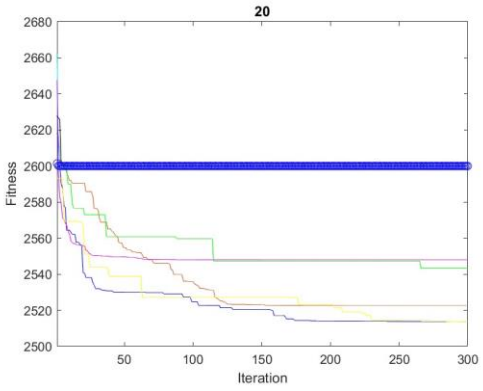
CEC2014-F21



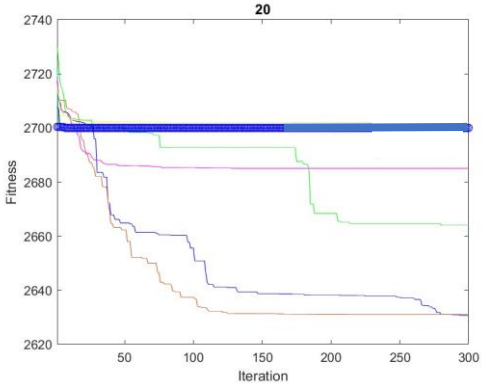
CEC2014-F22



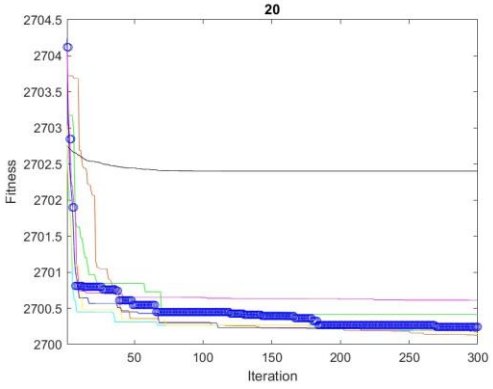
CEC2014-F23



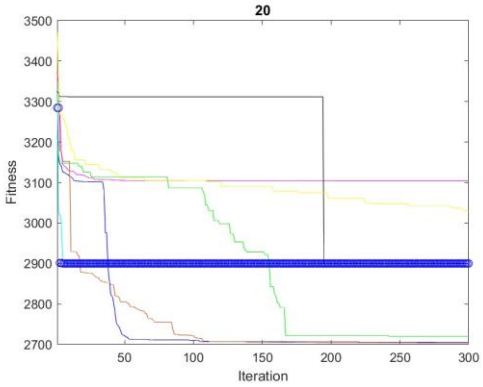
CEC2014-F24



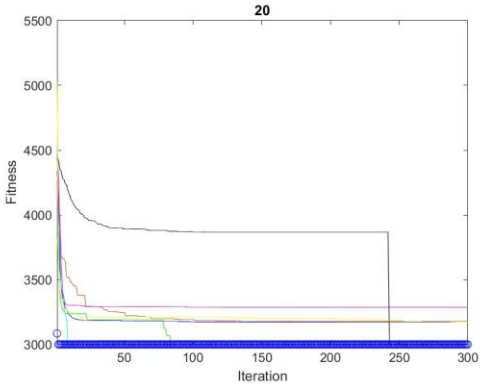
CEC2014-F25



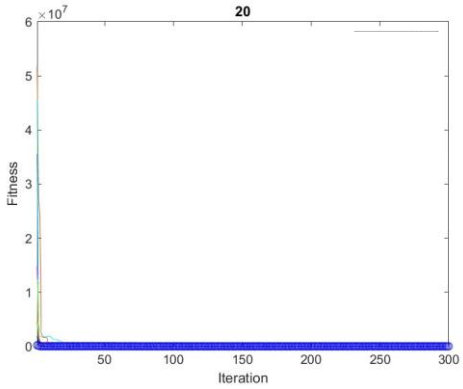
CEC2014-F26



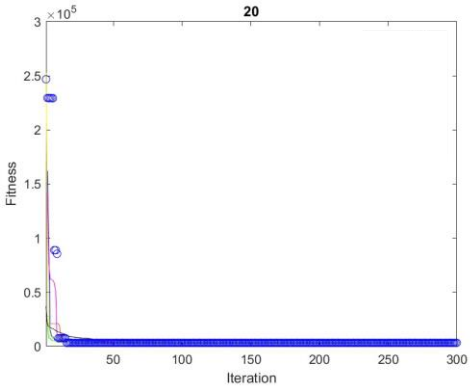
CEC2014-F27



CEC2014-F28

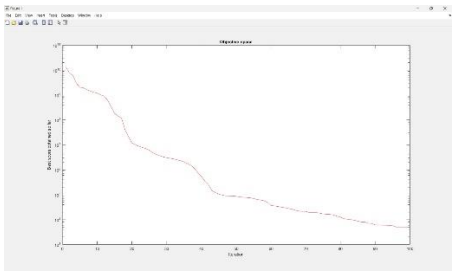


CEC2014-F29

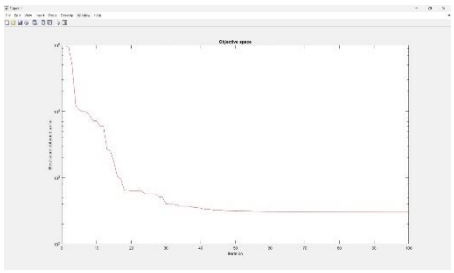


CEC2014-F30

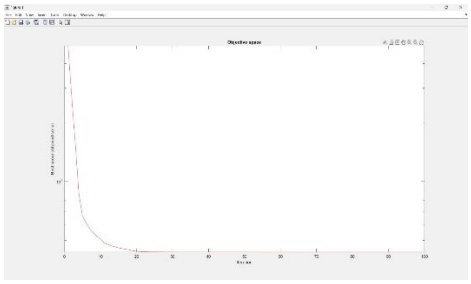
8.0.2. Convergence Curves for CEC2017



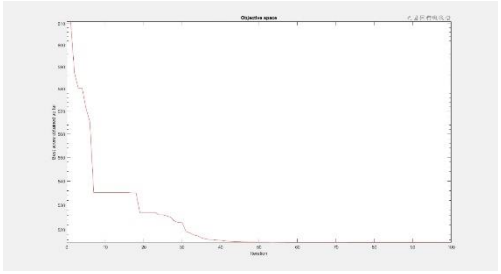
CEC2017-F1



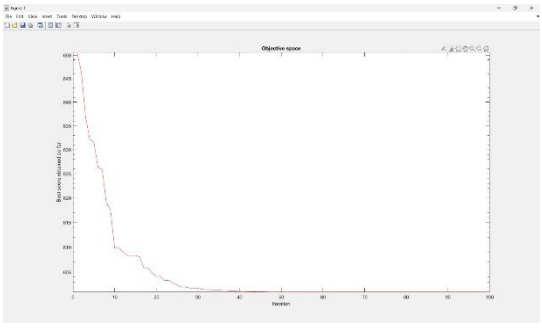
CEC2017-F3



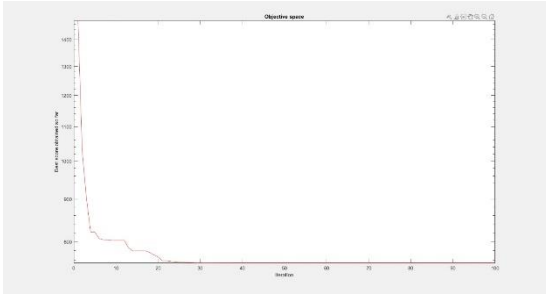
CEC2017-F4



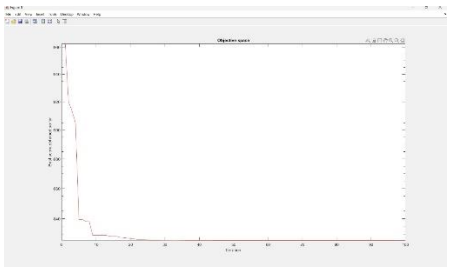
CEC2017-F5



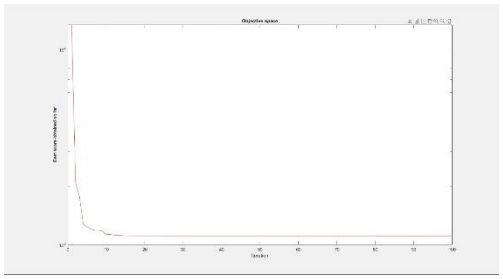
CEC2017-F6



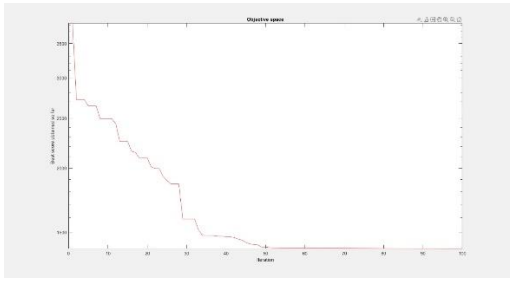
CEC2017-F7



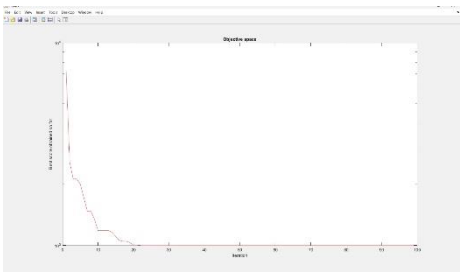
CEC2017-F8



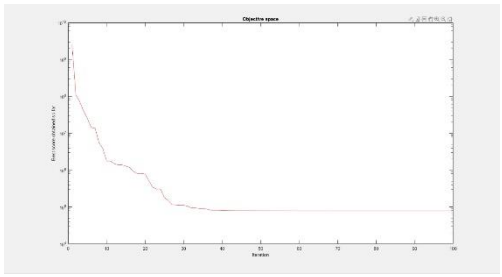
CEC2017-F9



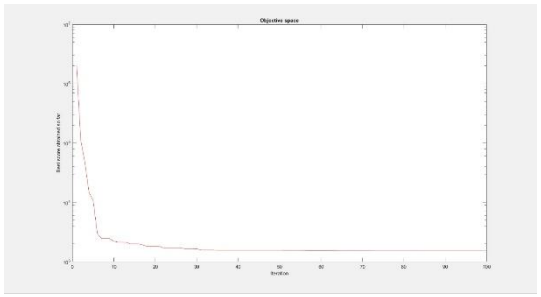
CEC2017-F10



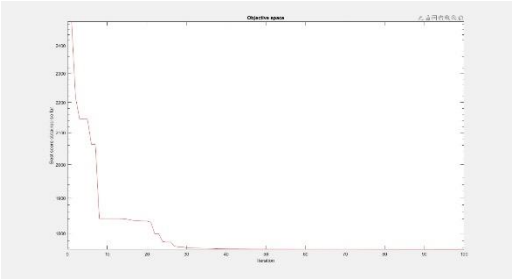
CEC2017-F11



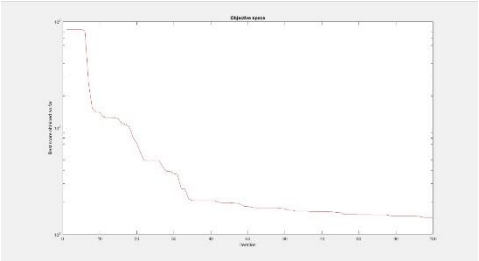
CEC2017-F12



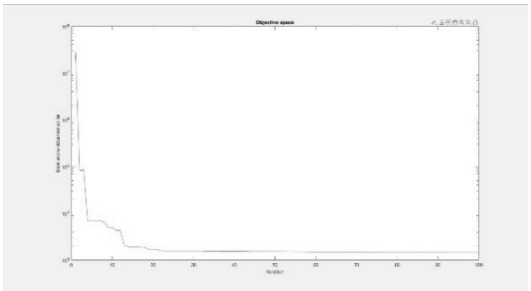
CEC2017-F13



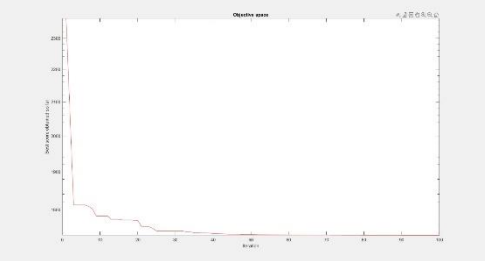
CEC2017-F14



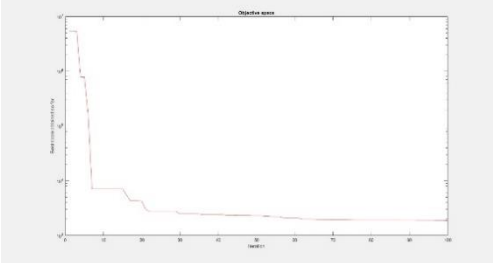
CEC2017-F15



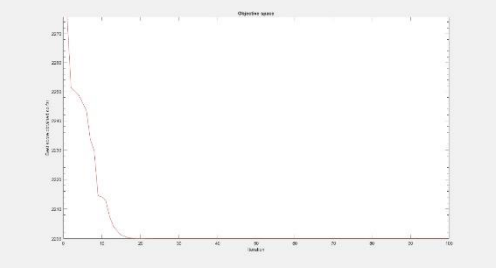
CEC2017-F16



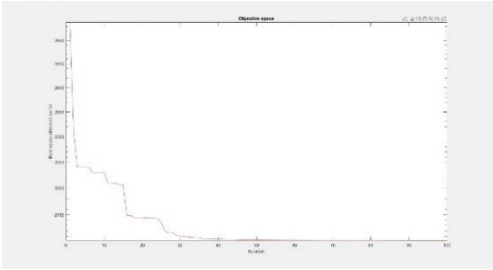
CEC2017-F17



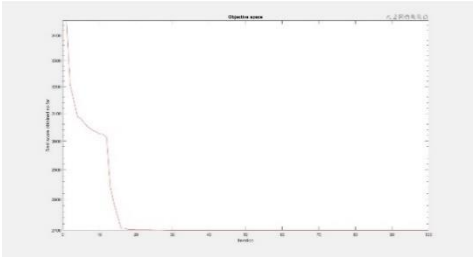
CEC2017-F18



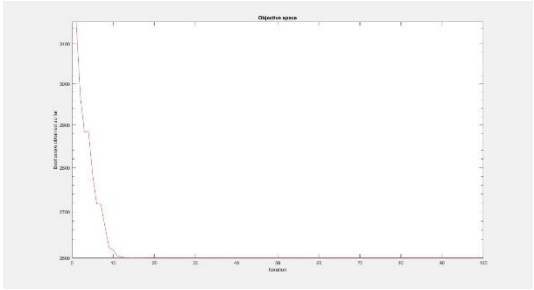
CEC2017-F19



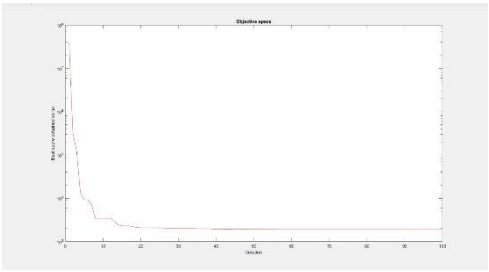
CEC2017-F20



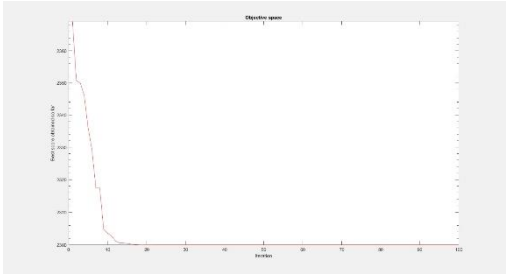
CEC2017-F21



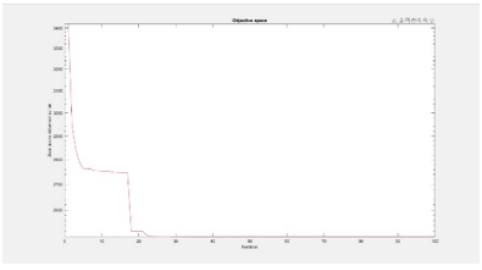
CEC2017-F22



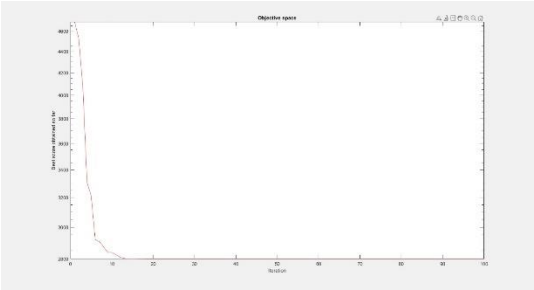
CEC2017-F23



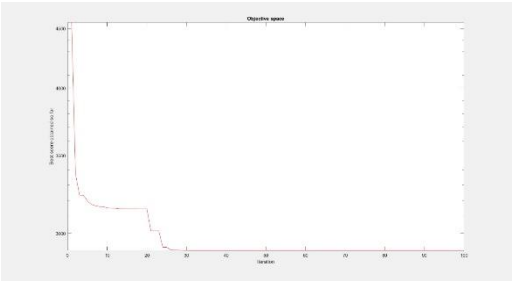
CEC2017-F24



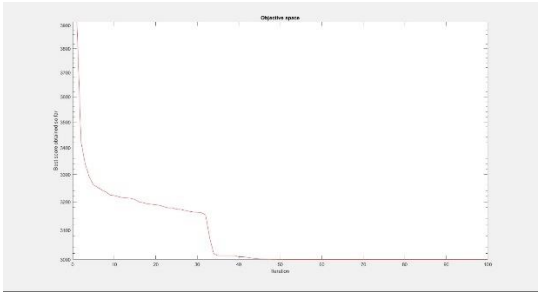
CEC2017-F25



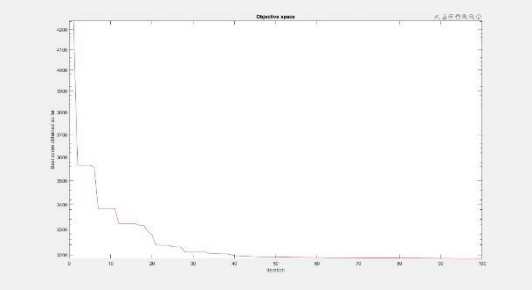
CEC2017-F26



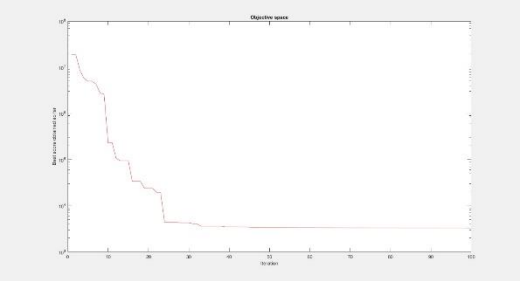
CEC2017-F27



CEC2017-F28



CEC2017-F29



CEC2017-F30

9. Conclusions

This study introduces an enhanced variant of the Walrus Optimization Algorithm (WOA) termed the Modified WOA (MWOA). This iteration addresses inherent limitations of the original algorithm by integrating a suite of methodologies and strategies tailored for exploration and exploitation, as outlined previously. WOA is designed to bolster population diversity and guide exploration, thereby mitigating premature convergence tendencies observed in WOA. The resultant improvements manifest in enhanced convergence speed and solution accuracy.

In a comprehensive comparative analysis encompassing seven metaheuristic algorithms, including WOA, FHO, SASO, AOA, COA, and GWO, MWOA sometimes emerges as a frontrunner in the pursuit of closer-to-optimal solutions. Noteworthy advantages include the larger step size encouraging broader exploration of the search space. This can help the algorithm escape local optima and discover diverse solutions and as the iterations advance (`t` increases), the step size gradually decreases. This finer adjustment supports exploitation by focusing on refining promising solutions and converging towards the global optimum. These findings underscore MWOA's efficacy as a formidable optimization tool.

Future endeavors may focus on further refining MWOA to address residual challenges associated with premature convergence in certain functions, where optimal solutions have yet to be attained. Through continued refinement and enhancement, we anticipate that MWOA will solidify its position as the preeminent optimization tool for addressing a myriad of engineering and real-world challenges.

References

1. Du ZG, Pan JS, Chu SC et al (2020) Quasi-afne transformation evolutionary algorithm with communication schemes for application of rssi in wireless sensor networks. IEEE Access 8:8583–8594.
2. Du ZG, Pan JS, Chu SC et al (2022) Multi-group discrete symbiotic organisms search applied in traveling salesman problems. Soft Comput 26(9):4363–4373.
3. Pan Q, Tang J, Zhan J et al (2023) Bacteria phototaxis optimizer. Neural Comput Appl 35(18):13433–13464.
4. Chu SC, Du ZG, Pan JS (2020) Symbiotic organism search algorithm with multi-group quantumbehavior communication scheme applied in wireless sensor networks. Appl Sci 10(3):930..
5. Du ZG, Pan JS, Chu SC et al (2020) Improved binary symbiotic organism search algorithm with transfer functions for feature selection. IEEE Access 8:225730–225744.
6. . Mirjalili S, Mirjalili S (2019) Genetic algorithm. In: Evolutionary algorithms and neural networks: theory and applications, pp 43–55..
7. . Delahaye D, Chaimatanan S, Mongeau M (2019) Simulated annealing: from basics to applications. In: Handbook of metaheuristics, pp 1–35
8. Wang D, Tan D, Liu L (2018) Particle swarm optimization algorithm: an overview. Soft Comput 22:387–408
9. Chu SC, Du ZG, Peng YJ et al (2021) Fuzzy hierarchical surrogate assists probabilistic particle swarm optimization for expensive high dimensional problem. Knowl Based Syst 220(106):939.
10. Biswas A, Mishra K, Tiwari S, Misra A (2013). Physics-inspired optimization algorithms: a survey. Journal of Optimization, 2013
11. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, Gsa: a gravitational search algorithm, Information sciences 179 (13) (2009) 2232–2248.
12. F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems, Applied intelligence 51 (2021) 1531–1551.
13. Sarzaeim P, Bozorg-Haddad O, Chu X (2018). Teaching-learning-based optimization (TLBO) algorithm. Advanced Optimization by Nature-Inspired Algorithms. Singapore, Asia: Springer, 51–58
14. Moosavi S, Bardsiri V (2019) Poor and rich optimization algorithm: a new human-based and multi populations algorithm. Eng Appl Artif Intel 86:165–181
15. X.-S. Yang, M. Karamanoglu, Nature-inspired computation and swarm intelligence: a stateof-the-art overview, Nature-Inspired Computation and Swarm Intelligence (2020) 3–18.
16. A. Kumar, M. Nadeem, H. Banka, Nature inspired optimization algorithms: a comprehensive overview, Evolving Systems 14 (1) (2023) 141–156.
17. J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95international conference on neural networks, Vol. 4, IEEE, 1995, pp. 1942–1948.
18. M. Azizi, S. Talatahari, A. H. Gandomi, Fire hawk optimizer: A novel metaheuristic algorithm, Artificial Intelli gence Review 56 (1) (2023) 287–363.
19. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in engineering software 69 (2014) 46–61.
20. S. Mirjalili, A. Lewis, The walrus optimization algorithm, Advances in engineering software 95 (2016) 51–67.
21. R. Salgotra, U. Singh, A novel bat flower pollination algorithm for synthesis of linear antenna arrays, Neural Computing and Applications 30 (2018) 2269–2282.
22. M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE computational intelligence magazine 1 (4) (2006) 28–39.
23. Agushaka JO, Ezugwu AE, Abualigah L (2023) Gazelle optimization algorithm: a novel natureinspired metaheuristic optimizer. Neural Computer Appl 35(5):4099–4131.

24. Han, M., Du, Z., Yuen, K. F., Zhu, H., Li, Y., & Yuan, Q. (April 2024). Walrus optimizer: A novel nature-inspired metaheuristic algorithm. Expert Systems with Applications